## lefko3: a gentle introduction

Creating and analyzing matrix projection models in $R$

Richard P. Shefferson

## Contents

Front Matter ..... 7
Copyright ..... 7
Dedication ..... 7
About the author ..... 7
Preface ..... 7
Structure of this book ..... 8
Acknowledgements ..... 8
Book version history ..... 8
1 Introduction to $R$ and lefko3 ..... 11
1.1 An Intro to R and RStudio ..... 12
1.2 Basic mathematics, statistics, and programming operations in R ..... 12
1.3 Memory handling in R and lefko3: sparse matrices ..... 17
1.4 Further memory handling: clearing memory ..... 19
1.5 Control structures ..... 19
1.6 Data input and handling ..... 20
1.7 Using lefko3 ..... 24
1.8 Datasets used in this book ..... 34
1.9 Points to remember ..... 37
2 Preliminaries I: Life History Models ..... 39
2.1 The life cycle graph ..... 39
2.2 Life history model development in lefko3 ..... 43
2.3 Stage classification with multiple size metrics ..... 56
2.4 Automating the creation of large numbers of stages ..... 63
2.5 Advanced stageframe creation ..... 66
2.6 Points to remember ..... 70
3 Preliminaries II: Data Formatting ..... 71
3.1 Kinds of individual state variables that may be included ..... 72
3.2 Formatting demographic data properly for analysis ..... 73
3.3 Function verticalize3() ..... 74
3.4 Function historicalize3() ..... 82
3.5 Handling spatial data and density ..... 87
3.6 Exploring stage-classified standardized datasets ..... 92
3.7 Formatting supplemental data ..... 99
3.8 Handling complex life histories ..... 113
3.9 Points to remember ..... 116
4 Matrix Models I: Intro to MPMs, and Raw (Empirical) MPMs ..... 119
4.1 Developing raw MPMs ..... 120
4.2 Ahistorical vs. historical matrix models ..... 142
4.3 Alternate parameterizations of historical MPMs ..... 148
4.4 Sparse matrix format ..... 150
4.5 Matrix reduction ..... 152
4.6 Arithmetic mean matrices ..... 153
4.7 Points to remember ..... 161
5 Matrix Models II: Function-based MPMs ..... 163
5.1 Developing vital rate models ..... 164
5.2 Assessing vital rates in Cypripedium candidum ..... 168
5.3 Setting up function modelsearch() ..... 173
5.4 Size and fecundity distributions ..... 186
5.5 Using lefkoMod objects to create function-based MPMs ..... 187
5.6 Using stage groups for complex MPMs ..... 208
5.7 Environmental and individual covariates ..... 220
5.8 Creating function-based MPMs without modelsearch() ..... 231
5.9 Points to remember ..... 241
6 Matrix Models III: Age (Leslie), Hybrid Age, and Age-by-Stage MPMs ..... 243
6.1 What are age-based, age-hybrid, and age-by-stage MPM? ..... 243
6.2 Developing function-based age-by-stage MPMs ..... 245
6.3 Developing raw age-by-stage MPMs ..... 278
6.4 Using supplements with age-by-stage MPMs ..... 287
6.5 Age-classified (Leslie) MPMs ..... 290
6.6 Age-hybrid MPMs ..... 299
6.7 Points to remember ..... 307
7 Matrix Models IV: Integral Projection Models ..... 309
7.1 Midpoint method vs. cumulative density function (CDF) ..... 312
7.2 Creating IPMs ..... 314
7.3 Quality control ..... 342
7.4 Points to remember ..... 350
8 Population Projection I: Deterministic Analysis ..... 351
8.1 Population growth rate ..... 355
8.2 Stable stage distribution and reproductive value ..... 357
8.3 Sensitivity analysis ..... 366
8.4 Elasticity analysis ..... 371
8.5 Further analyses ..... 378
8.6 Points to remember ..... 378
9 Population Projection II: Temporal Environmental Stochasticity ..... 379
9.1 Stochastic population growth rate ..... 383
9.2 Long-run average stage distribution and reproductive value ..... 385
9.3 Stochastic sensitivity analysis ..... 389
9.4 Stochastic elasticity analysis ..... 390
9.5 Points to remember ..... 393
10 Population Projection III: Projection Simulations ..... 395
10.1 Density dependence ..... 396
10.2 Projecting existing MPMs ..... 397
10.3 Projecting function-based MPMs ..... 443
10.4 Points to remember ..... 479
11 Special Analyses I: Life Table Response Experiments ..... 481
11.1 The one-way fixed LTRE ..... 481
11.2 The approximate stochastic LTRE ..... 482
11.3 The small-noise approximation LTRE ..... 482
11.4 Example analysis with Anthyllis vulneraria ..... 483
11.5 Points to remember ..... 545
12 Further Issues I: Importing Matrices and MPMs ..... 547
12.1 Creating a new lefkoMat object from imported matrices ..... 547
12.2 Importing matrices from COMPADRE and COMADRE ..... 571
12.3 Points to remember ..... 579
13 Further Issues II: Importing IPMs and Function-based MPMs ..... 581
13.1 Generating standard ahistorical IPMs and function-based MPMs ..... 584
13.2 Importing complex IPMs and fbMPMs ..... 605
13.3 Running projections from imported IPMs and fbMPMs ..... 615
13.4 Points to remember ..... 616
14 Further Issues III: Editing Matrices in MPMs ..... 617
14.1 Subsetting a lefkoMat object ..... 618
14.2 Adding and deleting matrices ..... 620
14.3 Editing elements within matrices ..... 623
14.4 Dealing with age-specific issues ..... 626
14.5 Adding stages to MPMs ..... 632
14.6 Large matrices ..... 640
14.7 Points to remember ..... 644
15 Further Issues IV: Quality Control ..... 645
15.1 Quality control in life history models and vertical datasets ..... 645
15.2 Quality control in vital rate models ..... 662
15.3 Quality control in MPMs and discretized IPMs ..... 670
15.4 Points to remember ..... 671
16 Appendix I: Full Analysis Code Examples ..... 673

# Front Matter 

## Copyright

All content copyright 2022 Richard P. Shefferson

## Dedication

This book is dedicated to the people of Ukraine, who are teaching the world every day that all people have the inherent human right to self-determination.

## About the author

Richard P. Shefferson earned his PhD in Integrative Biology and his MS in Range Management from the University of California at Berkeley. He is currently an associate professor at the University of Tokyo. His research is focused on evolutionary demography, life history evolution, and population dynamics, with particular interests in rare, long-lived plants, eco-evolutionary dynamics, and plantfungal symbiosis. He spends his free time spreading world peace through computational population ecology, and occasional interludes of heavy metal guitar solos.

## Preface

Matrix projection models (MPMs) have been used by ecologists for nearly a century now. Particularly since the 1990s, software packages have occasionally come onto the market offering students of population ecology a means of building or, more often, analyzing specific types of MPMs. However, as far as I am aware, no one has ever attempted to put together all the different forms of MPMs into one main package that would both build and analyze them. Package lefko3 is an attempt to fill that void. In filling that void, lefko3 provides a one-step programming environment for MPM analysis.

Package lefko3 not only builds all kinds of MPMs, including discretized IPMs, but also provides a variety of functions that standardize the process and make it easier. One perennial issue that crops up when population ecologists collaborate, for example, is that everyone has their own way of entering data, and translating that into the proper format can be truly time consuming. lefko3 offers functions that standardize demographic data regardless of what format that they are organized in, and also provides standardized programming objects for the matrix models and further analytical objects. Graduate students and natural areas managers should find lefko3 particularly useful for its simplicity in developing common analyses. Advanced population ecologists should find it useful for its flexibility in creating custom projections, including simulations of density dependent and stochastic runs involving patch dynamics.
lefko3 is a free, open-source package made for the $R$ programming language. $R$ is a free, opensource programming language based on the $\mathrm{S}+$ programming language, and is now the most commonly used programming language for statistical analysis. Packages are available from a variety of sources,

CONTENTS
but the most stable and well-supported packages are available natively through CRAN, which also maintains R itself. Package lefko3 is available through R's native install.packages() context and through CRAN's lefko3 page, with developmental versions also available on R Forge's lefko3 page.

This book was written as intro to MPM analysis using lefko3. It is based roughly on Evan Cooch and Gary White's classic (and continually evolving) Program Mark: A Gentle Introduction (http://www.phidot.org/software/mark/docs/book/). As such, this book can be thought of as a short course on using lefko3, and just as in the program MARK case, this book will keep growing and evolving as we develop more content both for package lefko3 and for this book. Ultimately, this is because although vignettes are provided within lefko3, space limitations on CRAN-based vignettes prevent the incorporation of key details and lengthy explanations, and it is just these details and lengthy explanations that can often help students understand how to use new analytical software. This book attempts to deal with these weaknesses of the vignettes by taking users step by step through the building and analysis of MPMs, even offering some of the theory behind population ecology along the way. Note, however, that this book is not a substitute for a good ecology textbook, but should be used in conjunction with one for the best possible understanding.

As with all such projects, this book is a work in progress. The author encourages readers who find mistakes or issues, or who simply have questions along the way, to contact him.

## Structure of this book

This book is organized into four main parts. Chapters 1,2 , and 3 deal with theory and data preparation required prior to the development of matrices. Chapters 4 through 7 introduce the different matrix projection models and illustrate how they are created. Chapters 8 through 11 are focused on analysis, currently including deterministic, stochastic, cyclical, and density dependent analyses, and custom projections. Chapters 12 through 15 deals with issues and extensions in developing and analyzing MPMs. Chapter 16 is an appendix including full code for basic MPM and construction and analysis, in all of the different MPM types.

We are planning to expand this book with several more chapters. Currently, these include planned chapters on transient dynamics, demographic stochasticity and individual-based simulation, and adaptive dynamics. These chapters and others will be developed as we introduce new functions and procedures to lefko3 itself.

## Acknowledgements

I wish to thank all of the people who have contributed to this work. E. Holton provided discussion and feedback throughout the writing of this book, and also tested code and procedures. J. Nagata provided a sounding board for ideas. The University of Tokyo, Saza Coffee, and Little Bean Coffee provided places to sit and think. The Japan Society for the Promotion of Science provided grant-in-aid 19 H 03298 , which funded the development of this work and of lefko3 itself.

## Book version history

1.0 14 Apr 2022 Original volume with 12 chapters (including front matter). Built with lefko3 v. 5.1.0 and R 4.1.3.
1.1 21 Apr 2022 Equations added to section on substochastic enforcement (chapter 10).
2.0 29 July 2022 New chapters added on LTRE analysis and on importing IPMs. New material added on function actualstage3(). Data object anthyllis added. Updated for lefko3 version 5.4.1.
2.1 14 Sept 2022 Material dealing with function modelsearch() updated to deal with changes to accuracy calculation. Also corrected f_projection3() section to deal with fixe bugs in vital rate density dependence. Updated to lefko3 version 5.5.0.
3.0 3 Apr 2023 Book updated for lefko3 6.0.0. Chapters added on editing matrices and quality control. Importing MPMs from online databases has been included.
3.1 9 May 2023 Book updated for lefko3 6.0.5. Material regarding elasticity and LTRE analyses has been corrected where needed. Material added related to memory use in sparse vs. standard matrices. 3.210 Jul 2023 Book updated for lefko3 6.1.0. Appendix A added with full code for all major MPM types. Function add_stage() added and key areas corrected to reflect bug fixes in lefko3.

## Chapter 1

# Introduction to $R$ and lefko3 

"Computers make excellent and efficient servants, but I have no wish to serve under them."

- Spock, Star Trek, "The Ultimate Computer"

R package lefko3 is devoted to the analysis of demographic data through matrix projection models (MPMs) (Shefferson et al., 2021). It serves as a full working environment for the construction and analysis of all kinds and styles of MPMs, including Lefkovitch (size-classified) MPMs, Leslie (age-based) MPMs, age $\times$ stage MPMs, and discretized integral projection models (IPMs). It can create and analyze both raw (empirical) and function-based forms of these models. It was originally developed to estimate and analyze historical size-classified matrix projection models (hMPMs), which are matrices designed to include the state of an individual in three consecutive times, in contrast to the two consecutive times that characterize most MPMs and IPMs. Such matrices are large, typically having dimensions several orders of magnitude higher than their standard, ahistorical counterparts (the latter will be hereafter referred to as ahistorical MPMs, or ahMPMs, while the acronym MPM will be used to refer to all matrix projection models, whether historical or not). As this package has developed, we have prioritized the development of core algorithms and methods to construct these models and the full suite of possible MPM types quickly, efficiently, and at least relatively painlessly. The result is a package that builds and analyzes MPMs of all types and all sizes quickly and efficiently, with enough flexibility that just about anyone interested in developing MPMs will find it useful.

This package introduces a complete suite of functions covering the MPM workflow, from dataset management to the construction of MPMs to their analysis. Dataset management functions standardize demographic datasets from the dominant formats demographers use into a format that facilitates MPM estimation while accounting for individual identity and other parameters. Demographic vital rate models may be estimated using demographic datasets with this standardized format, and these models take the form of generalized linear or mixed linear models using a variety of response distributions. Matrix estimation functions produce all necessary matrices from a single dataset, including all times, patches, and populations in a single shot, and do so quickly through core binaries engineered for speed and accuracy.

This manual assumes that the user has a very basic knowledge of R . We do not assume that users utilize or are even aware of any other packages, instead focusing on commands within lefko3 and base R (there are a few small exceptions, and these exceptions will include enough details to guide users without familiarity with those packages). We will begin our introduction with an overview of some required knowledge about R , for those who may be lacking even the very basic knowledge.

### 1.1 An Intro to R and RStudio

R is an object-oriented, open access programming language based on the $\mathrm{S}+$ statistical programming language. It is available for free online (www.r-project.org), and operates at the command line. RStudio (www.rstudio.com) is a free development environment for R . It makes using R simpler by offering an organized space to see, write, and save code, view code outputs, track what is currently held in memory, and generally organize analyses. Use of RStudio is not required to use lefko3, but we encourage it, as it simplifies most analyses. This book was written assuming that lefko3 users would program in RStudio.

RStudio also allows the use of R Notebooks (file type ending with the .Rmd extension). This file type provides a means of mixing R code with text and output, as well as html, SQL, Python, and some other programming languages. One of the key initial advantages of using R Notebooks is that $R$ automatically treats the directory in which the current $R$ Notebook file is located in as the default directory for any file operations within the R Notebook. This is an advantage because R's default directory for any code outside of an R Notebook is the R directory itself, and so using R Notebooks allows the user to skip resetting the directory at the outset of each programming session.

To use R Notebooks, first make sure that you have downloaded and installed R. Then download, install, and open RStudio. RStudio will open R command-line within the lower-left panel. The topleft panel will be a place for you to open, read, and write code. The top-right panel shows what is in memory (the Environment panel), as well as a history of commands entered (the History panel), and perhaps a few other odds and ends. The lower-right panel shows files in the current directory (Files panel), plots (Plots panel), installed packages (Packages panel), and help documents (Help panel), among potentially other things.

RStudio is set up initially to handle $R$ analyses and script. However, it cannot yet properly handle R Notebooks. To write and read R Notebooks, we just need to perform one more step - to install the markdown and rmarkdown packages. To do so, make sure that you are connected to the internet, and enter the following commands at the R prompt, and, when R finishes, restart RStudio:

```
install.packages("markdown", dependencies = TRUE)
install.packages("rmarkdown", dependencies = TRUE)
```

Now you're ready to go! To start a new R Notebook, click R Notebook under the New File option in R Studio's File menu. Make sure to save it in an appropriate place on your computer.

### 1.2 Basic mathematics, statistics, and programming operations in $\mathbf{R}$

R allows us to do basic mathematical tasks, such as setting variables and doing basic arithmetic. Here we see such an example. First we ask R what $5+4$ equals.

```
5+4
> [1] 9
```

Next, we create a variable x that we set to be equal to $5+4$, and print the value of x to the screen.

```
x <- 5+4
x
> [1] 9
```

In the output above, $R$ first shows us the answer to the problem $5+4$. It denotes this answer after a [1] because $R$ defaults to treating problems as though they were problems with vectors. Thus, a
one-element vector with the element 5 is added to a one-element vector with a value of 4 , and yields a one-element vector with a value of 9 . In the next set of code, we create a variable named x and assign it the value of $5+4$ using R's main piping operator, <- (another piping operator is the equal $\operatorname{sign},=)$. When we type x and press return, we see a one-element vector with the value of 9 in that element. If we had worked with a long vector, then each wrapped line of values would begin with a bracketed number corresponding to the element number within the vector of the first element in that row, making it a bit easier to see the structure of that vector.

The above example illustrates the use of objects. R is an object-oriented programming language, meaning that using $R$ generally requires creating objects that act to hold data and other values, or to perform different functions. Objects always need names, with a few rules governing what names are possible: 1) object names are case-sensitive, meaning that objectx and object X are different objects; 2) object names may not begin with numbers, as well as a few special characters (such as *); and 3) objects should be named uniquely, to prevent existing functions or other objects being overwritten in memory. We can see what objects are in memory in the Environment panel in the top-right portion of the RStudio screen, where we see the names of the objects and some basic information about these objects.

Use of lefko3 also requires familiarity with certain key data classes. The most important of these are the vector, the matrix, the list, and the data frame. We can begin with the vector, which is probably the most important class and is defined using the $c$ () function. A function is a special operation in a programming language, and in R , the function takes input provided within parentheses, as below. We will use the c() function to create two vectors, the first a numeric vector, and the second a string vector, with the elements of each vector given as input within parentheses.

```
simple_vector <- c(1, 2, 3, 4, 5, 6, 7, 8, 9)
text_vector <- c("1", "2", "3", "4", "5", "6", "7", "8", "9")
simple_vector
> [1] 1 2 3 4 5 6 7 8 9
text_vector
> [1] "1" "2" "3" "4" "5" "6" "7" "8" "9"
```

Vectors are atomic, meaning that all elements of a vector are of the same type. The first is a vector of numeric class, meaning that all elements are assumed to be floating-point decimals. The second is of class character, meaning that they are assumed to be pure text. We can see this by checking the class of each vector, as below.

```
class(simple_vector)
> [1] "numeric"
class(text_vector)
> [1] "character"
```

Vectors are core building blocks for other structures in R. For example, we can use them to build matrices. Below, we use the matrix() function to create a simple numerical matrix. Note that this function takes specific arguments as input, and the names of these arguments come before each equal sign within parentheses. So, for example, the argument nrow refers to the number of rows in the matrix, and our statement nrow $=3$ tells R to make a matrix with three rows.

```
simple_matrix <- matrix(c(1, 2, 3, 4, 5, 6, 7, 8, 9), nrow = 3, ncol = 3)
simple_matrix
> [,1] [,2] [,3]
> [1,] 1 4 4
> [2,] 2 5 8
> [3,] 3 6 9
```

This matrix was built by taking a vector and filling the matrix by column using the matrix() function. Filling the matrix by column is standard practice in computing, and reflects how other key programming languages treat vectors. However, users can also use the byrow = TRUE option to fill the matrix by row preferentially, as they might in Matlab. We can also use a previously defined vector to fill the matrix, and all of these procedures work provided that the vector is of length equal to the number of elements in the matrix to be filled.

```
simple_matrix_byrow <- matrix(simple_vector, nrow = 3, ncol = 3, byrow = TRUE)
simple_matrix_byrow
> [,1] [,2] [,3]
> [1,] 1 2 3
> [2,] 4 5 6
> [3,] 7 8 9
```

Matrices are also atomic, and so we can also build a text matrix, as below. If an attempt is made to combine numeric values and text values in a vector or matrix, then the resulting vector or matrix will treat all elements as text by default.

```
text_matrix <- matrix(text_vector, nrow = 3, ncol = 3)
text_matrix
> [,1] [,2] [,3]
> [1,] "1" "4" "7"
> [2,] "2" "5" "8"
> [3,] "3" "6" "9"
```

Importantly, although the elements of these matrices are of either type numeric or character, the class() function does not allow us to differentiate here, instead telling us only that the object is a matrix and an array (an array is simply a multi-dimensional object that can be propagated with a vector, and a matrix is a two-dimensional array). To differentiate, we can use the class() function on the elements of the matrix.

```
class(simple_matrix)
> [1] "matrix" "array"
class(text_matrix)
> [1] "matrix" "array"
class(simple_matrix[1,1])
> [1] "numeric"
class(text_matrix[1,1])
> [1] "character"
```

Elements of matrices are denoted in square brackets, with the row on the left and the column on the right of the comma. However, because matrices are filled with vectors, they can also be accessed via a single number, in which case they correspond to the place of the element in the corresponding vector used to fill the matrix (note that this number always refers to the element number in the associated column vector, even if the matrix was filled by row). Thus, we can access the eighth element in simple_matrix in two ways.

```
simple_matrix[2,3]
> [1] 8
simple_matrix[8]
> [1] 8
```

R's use of the vector as the default handling method for mathematical analysis means that even arithmetic operations handled by R are really done as problems in linear algebra. Thus, note that adding the scalar 3 to a numeric vector ends up adding 3 to each element in the vector, and adding 3 to a matrix adds 3 to each element in the matrix, as below.

```
simple_vector + 3
> [1] 4 4 5 [lllllllll
simple_matrix + 3
> [,1] [,2] [,3]
> [1,] 4 % 7 10
> [2,] 5 8 11
> [3,] 6 9 12
```

Preventing vectors from including multiple types of entries allows R to allocate memory efficiently. It also makes R's vectors consistent with vector definitions in the major programming languages, making R vectors computationally passable to other languages. However, sometimes we wish to build a vector composed of multiple types of objects. In these cases, we can build an object of class list. Lists are powerful and flexible objects, where each element can be any kind of element. In fact, not only can these elements be numeric, character, integer (i.e. whole numbers), or logical (i.e. true of false values only), but they can also be entire vectors, matrices, or even other lists. Here, we will create a new list including some of the objects that we have created so far.

```
first_list <- list(my_fave_vector= simple_vector, my_fave_matrix= simple_matrix,
    my_least_fave_matrix = text_matrix)
first_list
> $my_fave_vector
> [1] 1 2 3 4 5 6 7 8 9
>
> $my_fave_matrix
> [,1] [,2] [,3]
> [1,] 1 4 7
> [2,] 2 5 8
> [3,] 3 6 9
>
> $my_least_fave_matrix
> [,1] [,2] [,3]
> [1,] "1" "4" "7"
> [2,] "2" "5" "8"
> [3,] "3" "6" "9"
```

Here we see that our list has three objects, each of a different class. Each object has a name, and can be accessed using the $\$$ operator or via the double square bracket, as below.

```
first_list$my_fave_matrix
> [,1] [,2] [,3]
> [1,] 1 4 7
> [2,] 2 5 8
> [3,] 3 6 9
first_list[[2]]
> [,1] [,2] [,3]
> [1,] 1 [ 4 % 7
> [2,] 2 5 8
> [3,] 3 6 9
```

Finally, we come to the data frame. A data frame is essentially a dataset that meets R's formatting requirements. Thus, columns are variables, and rows are data points. The variables that are part of a data frame are vectors of equal length, but do not need to be of the same class. Technically, a data frame is actually a list object in which each element of the list is a vector of the same length, and so the variables are accessible in the same way that list elements are. Here is an example using the cars data frame, which comes packaged with base R. Note that before we can look at the data frame, we need to load it into our working memory using the data() function.

| > | speed | dist |
| :---: | :---: | :---: |
| $>1$ | 4 | 2 |
| $>2$ | 4 | 10 |
| $>3$ | 7 | 4 |
| $>4$ | 7 | 22 |
| $>5$ | 8 | 16 |
| $>6$ | 9 | 10 |
| $>7$ | 10 | 18 |
| $>8$ | 10 | 26 |
| $>9$ | 10 | 34 |
| > 10 | 11 | 17 |
| > 11 | 11 | 28 |
| > 12 | 12 | 14 |
| > 13 | 12 | 20 |
| > 14 | 12 | 24 |
| > 15 | 12 | 28 |
| > 16 | 13 | 26 |
| > 17 | 13 | 34 |
| > 18 | 13 | 34 |
| > 19 | 13 | 46 |
| > 20 | 14 | 26 |
| > 21 | 14 | 36 |
| > 22 | 14 | 60 |
| > 23 | 14 | 80 |
| > 24 | 15 | 20 |
| > 25 | 15 | 26 |
| > 26 | 15 | 54 |
| > 27 | 16 | 32 |
| > 28 | 16 | 40 |
| > 29 | 17 | 32 |
| > 30 | 17 | 40 |
| > 31 | 17 | 50 |
| > 32 | 18 | 42 |
| > 33 | 18 | 56 |
| > 34 | 18 | 76 |
| > 35 | 18 | 84 |
| > 36 | 19 | 36 |
| > 37 | 19 | 46 |
| > 38 | 19 | 68 |
| > 39 | 20 | 32 |
| > 40 | 20 | 48 |
| > 41 | 20 | 52 |


| $>42$ | 20 | 56 |
| ---: | ---: | ---: |
| $>43$ | 20 | 64 |
| $>44$ | 22 | 66 |
| $>45$ | 23 | 54 |
| $>46$ | 24 | 70 |
| $>47$ | 24 | 92 |
| $>48$ | 24 | 93 |
| $>49$ | 24 | 120 |
| $>50$ | 25 | 85 |

We can see that we have two variables in this data frame. Suppose we wished to access the sixth data point's value for the speed variable. We can do so in the following way.

```
cars$speed[6]
> [1] 9
```

Alternatively, we can see all of the values of the sixth data point by calling the row using single square brackets, and leaving the column blank, as below.

```
cars[6,]
> speed dist
> 6 9 10
```


### 1.3 Memory handling in R and lefko3: sparse matrices

Beginning with version 6.0.0, lefko3 includes sparse matrix capability. Let's first introduce the sparse matrix, and then see how $R$ and lefko3 handle these objects.

Mathematically, a sparse matrix is defined as a matrix in which at least $50 \%$ of the elements are 0 values. Such matrices have a number of interesting properties, but the most important from the computational standpoint relates their ability to save memory. Let's consider with an example.

Let's imagine that we are building a single matrix for a historical matrix projection model. Imagine that we have 100 stages in the life history of the organism, and because we are using a historicallyformatted MPM, we need to create a matrix that is 10,000 rows by 10,000 columns large. In memory, a matrix is represented by a vector composed of the number of elements in the matrix, each taking up the amount of memory required by a single value of the specific value type, plus a tiny bit extra for metadata relating to the number of rows and columns, and the type of variable. Since a typical floating-point decimal (technically type double) takes up 64 bits of data, and we have 10,000 by 10,000 $=100,000,000$ elements in our matrix, then our matrix takes up $6,400,000,000$ bits (equivalent to 800 MB), plus a tiny bit extra for meta-data. Since a typical MPM contains decomposed survival-transition and fecundity matrices in addition to the core matrices, this means that our MPM is already at least 2.4 GB.

Computationally, sparse matrix format can reduce this dramatically, provided that the matrix is truly sparse. In computing, sparse matrix formats generally use some variant of two or three vectors to store the data in the matrix, plus the aforementioned meta-data: one matrix composed of the non-zero values in the matrix, and a second and potentially third vector composed of the integer locations of those values within the matrix. For a mathematically dense matrix, this means that sparse matrix format actually takes up more memory than standard matrix format. However, for a mathematically sparse matrix, there can be serious memory savings. In our historical MPM case, for example, if we assume that all possible estimable elements are non-zero, then only $1 \%$ of the elements in that matrix will be non-zero, yielding a much smaller allocation to the matrix in memory (how we obtain this number is explained in detail in chapter 4). A typical sparse matrix format memory allocation for such
a matrix would be about 12 MB ( 64 bits for the value and 32 bits for the location integer for 1 million total values).

Package lefko3 uses the dgCMatrix format in package Matrix, which should be installed automatically when you install lefko3 (if it is not installed, then try running install.packages("Matrix")). To see an example of such a matrix, try the following:

```
sparse_example_vector <- c(0, 0, 0, 0, 0, 5, 0, 0, 0, 70, 3, 0, 0, 0, 0, 0)
sparse_example_sp_matrix <- Matrix::Matrix(sparse_example_vector, nrow = 4,
    sparse = TRUE)
sparse_example_sp_matrix
> 4 x 4 sparse Matrix of class "dtCMatrix"
>
> [1,] . . . .
> [2,] . 5 70 .
> [3,] . . 3 .
> [4,] . . . .
```

The above example does not really showcase the advantages and disadvantages of using sparse matrices very well. Let's try the following, instead. We will build two large matrices in standard format, the first a mathematically sparse matrix and the second a mathematically dense matrix. These are both 2000 rows by 2000 columns.

```
mat_example_1 <- matrix(0, 2000, 2000)
mat_example_1[1, 1] <- 0.5
mat_example_1[100, 45] <- 13.2
mat_example_1[1004, 1999] <- 2.4
mat_example_2 <- matrix(c(1:4000000), 2000, 2000)
```

Now let's build sparse format versions of these matrices, as below. Please note that you need to have the Matrix package installed to compute these sparse matrices.

```
spmat_example_1 <- Matrix::Matrix(mat_example_1, sparse = TRUE)
spmat_example_2 <- Matrix::Matrix(mat_example_2, sparse = TRUE)
```

Now that we have our matrices, let's see how much memory we have used. We will use the object_size() function in package pryr to do this.

```
# Comparison of first matrix in standard vs sparse format
pryr::object_size(mat_example_1)
> 32.00 MB
pryr::object_size(spmat_example_1)
> 9.55 kB
# Comparison of second matrix in standard vs sparse format
pryr::object_size(mat_example_2)
> 16.00 MB
pryr::object_size(spmat_example_2)
> 48.01 MB
```

The above output shows the advantages and disadvantages of sparse matrix use. The first matrix that we entered was very sparse, and using sparse matrix format cut down the size from 32 MB to
roughly 9.6 kB . The second matrix was mathematically completely dense, without a single 0 , and sparse matrix format increased the size of the object in memory from 16 MB to roughly 48 MB . Use of sparse matrices has many computational advantages, but only when the matrices are both spare and large. On the latter characteristic, the primary issue is that because a sparse matrix is technically two or three vectors, computations can take more time than with standard matrices. This results in the primary advantages to the use of sparse matrices only accruing once matrices are are at least several tens of rows and columns large. We will use sparse matrices when necessary in this manual to keep memory from becoming overwhelmed.

### 1.4 Further memory handling: clearing memory

Proper memory handling is an important art that will impact every scientific programmer at some point or other. Memory issues arise from two sources: 1) stray memory allocations that, due to errors in computation, affect structures in memory, and 2) memory cluttering caused by the creation of many large objects. We advise users to develop good habits in memory management, and offer these tips.

First, always begin your R session by clearing the memory. At the start of the session, this means using the rm() function to remove all objects, as follows.

```
rm(list=ls(all=TRUE))
```

This command can be used at other times, as well. Running it as is will remove all R objects from memory, while using it can be more focused to remove specific objects. For example, in the example below, we remove two objects form memory, the cars dataset and our sparse matrix.

```
rm("cars", "sparse_example_sp_matrix")
```

The Environment window in RStudio shows how much memory is being used. When large objects are created, memory starts to clutter, and it is possible to run out of available memory through the creation of object after object. Note, however, that removing objects does not necessarily lower the amount of memory actually used. This occurs because removing the object from memory does not necessarily clear the allocation itself - the object's details are simply removed from the memory's allocation table. To fully clear the memory associated wiht removed objects, then following the rm() function with the $\mathrm{gc}($ ) function ('gc' is an acronym for 'garbage collection'), as below.

```
gc()
> used (Mb) gc trigger (Mb) limit (Mb) max used (Mb)
> Ncells 2151522 115.0 4104654 219.3 NA 3354998 179.2
> Vcells 15757297 120.3 25101243 191.6 16384 20717345 158.1
```

No arguments are necessary for the function. The text output merely summarizes the kind of memory that was cleared. We will not explain it here - users interested in pursuing this further should search for further details on the internet.

### 1.5 Control structures

As in other programming languages, R uses control structures to define the flow of a program. These structures are of invaluable use in developing code for analysis. These structures can be categorized as either selection structures, or repetition structures. The most commonly used selection structure is the if statement. This statement works by testing a condition, and then executing a set of instructions if the condition is true. Let's see an example of this.

```
if (cars$speed[6] == 0) {
    writeLines("This car is not moving!")
} else if (cars$speed[6] < 0) {
    writeLines("This car is moving backward!")
} else {
    writeLines("This car is moving forward!")
}
> This car is moving forward!
```

Here, $R$ first sees if the $6^{\text {th }}$ car's speed in the cars dataset is zero. It tests this condition with the double equal sign, because the equal sign by itself is not used to test conditions but instead to assign values to variables and other objects. The else portion tells R what to do if the condition is not true. There are two elsestatements here, and the first tests another condition. The final else statement tells R what to do if none of the conditions tested is correct.

The most common repetition statement is the for loop. Here we see an example of such a loop.

```
total_dist <- 0
for (i in c(1:length(cars$speed))) {
    total_dist <- total_dist + cars$dist[i]
}
total_dist
> [1] 2149
```

In the chunk of code above, we first define a variable total_dist, and give it a starting value of 0 . We then create a for loop that initializes an integer variable $i$. The initial value of $i$ is 1 , and the for loop will then add each value of dist (the stopping distance in the cars dataset) to total_dist. The loop runs with i incrementing to each successive value in the $c()$ statement ( $c(1: l e n g t h(c a r s \$ d i s t))$ creates a vector of integers from 1 to the length of the speed variable in the cars dataset). When i increments to the last value, the loop ends.

Other structures also exist, including the while statement, which runs a loop as long as a condition is met. R-specific alternatives to loops exist, such as the apply series of functions, which can speed up complicated computation normally requiring loops.

### 1.6 Data input and handling

R deals primarily with statistical analysis and modeling. And, of course, statistical methods ultimately use data. There are numerous ways to get data into $R$, including importing spreadsheets or databases in various formats through a variety of functions. We can also create our own data frame, as below. Here, we create a series of atomic vectors of the same length using a version of the cbind() function which binds columns into a data frame (the cbind() function binds vectors of the same class together as columns within a matrix, while cbind.data.frame() binds vectors of any class together as variables within a data frame). Then, we use as.factor() to set one of these variables, gender, to be a categorical variable rather than a numeric variable.

```
gender <- c(1,1,1,2,2,2)
height <- c(1.5,1.4,1.6,1.5,1.6,1.5)
our.data <- cbind.data.frame(gender, height)
our.data$gender <- as.factor(our.data$gender)
our.data
```

| $>$ | gender | height |
| :--- | ---: | ---: |
| $>1$ | 1 | 1.5 |
| $>2$ | 1 | 1.4 |
| $>3$ | 1 | 1.6 |
| $>4$ | 2 | 1.5 |
| $>5$ | 2 | 1.6 |
| $>6$ | 2 | 1.5 |

Now let's suppose that we would like to understand more about our data, for example the various measures of central tendency and variability. We can use the summary () function to get at some of this, most importantly at the mean, median, and the quartiles, as below. Note that because gender is a categorical variable, we only see counts of the categories rather than any measures of central tendency and variability.

```
summary(our.data)
> gender height
> 1:3 Min. :1.400
> 2:3 1st Qu.:1.500
> Median :1.500
> Mean :1.517
> 3rd Qu.:1.575
> Max. :1.600
```

However, we would most likely want more than this. So, here are a few more useful functions. Note the use of the seq() function, which creates a sequence of values from the first number input to the second number input at increments of the third number input. This function is used to tell R the specific quantiles that we would like displayed.

```
mean(our.data$height)
> [1] 1.516667
median(our.data$height)
> [1] 1.5
sd(our.data$height)
> [1] 0.07527727
var(our.data$height)
> [1] 0.005666667
range(our.data$height)
> [1] 1.4 1.6
quantile(our.data$height, probs = seq(0, 1, 0.25))
> 0% 25% 50% 75% 100%
> 1.400 1.500 1.500 1.575 1.600
```

We might also wish to plot these data somehow. One of the most powerful functions to use to produce plots is plot(), which defaults to different styles of plot given different classes of input variables. Let's use this function to explore the relationship between distance traveled as a function of speed in the cars dataset (figure 1.1). We will add the option bty $=$ " n " in order to prevent R from framing the plot in a box (it's just a personal dislike of mine...).
plot(dist ~ speed, cars, bty = "n")

In the above plot, we utilized R's standard linear notation to tell it the relationship between variables. Thus, dist ~ speed can be interpreted statistically as dist is a function of speed. Note


Figure 1.1: 1st plot of distance against speed in cars dataset
also the reference to the cars dataset in the call - without this call, we would need to write cars $\$$ dist ~ cars\$speed. We can alter this plot in many ways, including the kinds of points, the labels, the axes, etc. Here is an example (figure 1.2).

```
plot(dist ~ speed, cars, pch = 2, col = "red", xlab = "Speed of car",
    ylab = "Distance until stop", xlim = c(0,40), ylim = c(0, 150), bty = "n")
```



Figure 1.2: 2nd plot of distance against speed in cars dataset
We can also try using the hist () function to create a histogram, for example to look at height by gender in our own dataset (figure 1.3).
hist(our.data\$height)
The hist () function can also be modified in a variety of important ways, using similar inputs to plot().

## Histogram of our.data\$height



Figure 1.3: Histogram of sample height data

Let's now see what the default plot style is for our gender-height data (figure 1.4).

```
plot(height ~ gender, our.data, bty = "n")
```



Figure 1.4: Box-whisker plot of sample height data
We can see that the default here is a box-and-whisker plot, which is standard when we incorporate an x term that is categorical and a y term that is quantitative and continuous.

Finally, now that we've explored a simple method of getting data into R, what about importing spreadsheets of demographic data? Although other methods certainly exist, we are particularly partial to the read.csv() function approach. This function allows us to import our data as a comma-separated value (CSV) text file. To utilize this approach, first open your preferred spreadsheet program and make sure that the data file meets the following conformational characteristics:

1. The variables in your dataset are arranged by column, and the data points themselves are arranged by row.
2. The top row in your spreadsheet contains ONLY the names of your variables, and these variable names conform to R object naming rules (i.e. each variable name unique, each variable name starts with a letter or underscore only, each variable name includes no punctuation).
3. The data sheet itself does not contain ANY commas anywhere.
4. All columns intended to be numeric variables contain ONLY numeric entries, or blanks (even a single text entry in such a variable will cause the variable to be imported as a string variable).
5. The dataset must start with the very top-left cell in the spreadsheet. Starting everything lower than the first row will cause improper importing and formatting issues.
6. The dataset does not include any characters other than pure ASCII characters. While R can handle UTF-8 and some other kinds of text encoding, some methods produce unexpected behavior when handling values or objects named with non-ASCII encoding.

Once your dataset conforms to these parameters in the spreadsheet program, export your file to CSV format (we generally encourage MS-DOS CSV format, if such an option is provided, as it is the simplest version). Then, in R , use the following command to import your dataset, replacing the myfilenamehere portion with the real file name:

```
mydata <- read.csv("myfilenamehere.csv", header = TRUE)
```

Once done, you can try using the summary () function to see what everything looks like and whether the variables were imported properly.

Let's now move on to using lefko3 itself.

### 1.7 Using lefko3

Users of lefko3 will first need to install the package itself. The following line of code will do that. The user should agree if prompted to install a number of other packages stated as dependencies. The specific dependencies currently include BH, glmmTMB, lme4, Matrix, MuMIn, pscl, Rcpp, RcppArmadillo, stringr, and VGAM, and they should be installed similarly if not prompted and if not installed. Note that lefko3 is also meant to work with the newest version of $R$. It has been tested with versions of $R$ as far back as 3.5.0 and so should work with those, but it is possible that dependency upgrades will affect certain functions (most importantly modelsearch(), which is dependent on virtually all of the packages noted above).

```
install.packages("lefko3", dependencies = TRUE)
```

Alternatively, development versions of lefko3 are available on R Forge, where updates are more frequent than on CRAN. If the user has R set up to compile $\mathrm{C}++$ and Fortran (this is not the default situation on Mac and Windows machines, so only advanced users are encouraged to do so), then the following line can also be used to install the latest development version of lefko3. Note that the CRAN version is stable and requires no compilation, and the R Forge version generally does not include updated documentation. So, DO NOT INSTALL THE R FORGE VERSION IF YOU ARE UNSURE OF WHAT COMPILING IS OR WHAT R NEEDS TO DO DURING INSTALLATION!
install.packages("lefko3", repos="http://R-Forge.R-project.org")
After that, use of lefko3 requires loading it into the working environment, which can be done with the following line.

```
library(lefko3)
```

Package lefko3 includes functions to handle the entire workflow of MPM construction and analysis. The functions themselves can be divided into at least 8 major categories, as below:

1. Data transformation and handling functions
a. add_1M(): Adds new matrices to existing MPMs.
b. delete_lM(): Deletes matrices from MPMs.
c. historicalize3(): Formats demographic datasets in vertical but not historical format.
d. subset_1M(): Creates MPMs from matrices in other MPMs.
e. verticalize3(): Formats demographic datasets in horizontal format.
2. Functions setting or determining population characteristics for analysis
a. density_input(): Summarizes density dependence inputs for population projection.
b. density_vr(): Summarizes density dependence inputs in vital rates for function-based population projection.
c. $h f v_{-} \mathrm{qc}()$ : Tests assumptions of various response distributions in key vital rate variables in demographic datasets.
d. overwrite(): A less powerful version of function supplemental().
e. sf_create(): Creates a data frame summarizing the life history model.
f. sf_distrib(): Tests for overdispersion and zero-inflation in count-based size and fecundity.
g. sf_skeleton(): Creates a stageframe skeleton.
h. start_input(): Creates a start vector for population projection.
i. summary_hfv(): Summarizes key information about properly formatted demographic datasets.
j. supplemental(): Summarizes extra inputs for MPM creation not found within the demographic dataset.
3. Vital rate model building and selection
a. create_pm(): Creates a skeleton data frame indexing variables used in vital rate modeling.
b. modelsearch(): Creates best-fit vital rate models for function-based MPMs, and to test the influences of individual history and other factors.
c. vrm_import(): Imports coefficients from linear vital rate models for use in developing discretized IPMs and function-based MPMs.
4. Matrix / integral projection model creation and import functions
a. mpm_create(): Constructs all forms of MPM.
b. create_1M(): Creates MPMs from matrices provided by the user.
c. aflefko2(): Constructs function-based ahistorical age-by-stage MPMs.
d. arlefko2(): Constructs raw ahistorical age-by-stage MPMs.
e. flefko2(): Constructs ahistorical function-based MPMs.
f. rlefko2(): Constrcuts ahistorical raw MPMs.
g. flefko3(): Constructs historical function-based MPMs.
h. rlefko3(): Constructs historical raw MPMs.
i. fleslie(): Constructs function-based Leslie MPMs.
j. rleslie(): Constructs raw Leslie MPMs.
k. hist_null(): Constructs historically-formatted MPM assuming no individual history.
l. Imean(): Develops element-wise arithmetic mean MPMs.
5. Matrix / integral projection model editing functions
a. edit_1M(): Edits matrices by element.
b. add_stage(): Adds a new stage to an existing MPM.
6. Population dynamics analysis functions
a. actualstage3(): Calculates actual population stage frequencies and proportions.
b. elasticity3(): Estimates the deterministic or stochastic elasticities of population growth rate to matrix elements.
c. f_projection3(): Core population projection function for time-by-time projection of function-based MPMs and IPMs from vital rate models.
d. lambda3(): Estimates deterministic population growth rate.
e. ltre3(): Conducts deterministic, stochastic, and small noise approximation life table response experiments.
f. projection3(): Core population projection function for already constructed MPMs and IPMs.
g. repvalue3(): Estimates the deterministic stage reproductive values or stochastic long-run reproductive values.
h. sensitivity3(): Estimates the deterministic or stochastic sensitivities of population growth rate to matrix elements.
i. slambda3(): Estimates stochastic population growth rate.
j. stablestage3(): Estimates the deterministic stable stage equilibrium or stochastic longrun stage distribution.
7. Functions describing, summarizing, comparing, or visualizing MPMs and derived structures.
a. cond_hmpm(): Extracts conditional ahistorical matrices from historical MPMs.
b. cond_diff(): Develops conditional ahistorical difference matrices, given two sets of historical or historically-formatted MPMs as input.
c. diff_1M(): Creates difference matrices between two MPMs wioth matrices of equal dimension.
d. image3(): Creates images of MPMs, sensitivity matrices, elasticity matrices, and LTRE matrices.
e. summary (): Summarizes a variety of different objects, including MPMs and analyses.
8. Other useful functions.
a. beverton3(): The two-parameter Beverton-Holt function for scalars.
b. logistic3(): The two-parameter logistic function for scalars.
c. ricker3(): The two-parameter Ricker function for scalars.
d. usher3(): The two-parameter Usher function for scalars.
e. plot.lefkoProj(): Displays plots of population projections resulting from functions projection3() and f_projection3().
f. summary (): Various versions of the summary () function have been created to provide useful summaries of output from many lefko3 functions.

In addition, lefko3 includes five datasets that can be used for educational purposes. These include anthyllis, cypdata, cypvert, lathyrus, and pyrola.

### 1.7.1 The lefkoMat object: organized MPMs and IPMs

Package lefko3 produces a number of different classes of S3 objects, which can be thought of as standardized lists used as input and output for its functions. The most fundamental of these is the
lefkoMat object, which holds the MPM and associated metadata. This list object is made of the following elements.

1. A: A list of full population projection matrices, in order of population, patch, and year. Each matrix is either of $R$ class matrix or Matrix class dgCMatrix (the latter is sparse matrix format).
2. U: A list of matrices showing only survival-transition elements, in the same order as A.
3. F: A list of matrices showing only fecundity elements, in the same order as A.
4. hstages: A data frame showing the order of paired stages (given if matrices are historical, otherwise NA).
5. agestages: A data frame showing the order of age-stages (if an age-by-stage MPM has been created, otherwise NA).
6. ahstages: A data frame detailing the characteristics of the life history model used for construction of the MPM.
7. labels: A data frame showing the order of matrices, according to population, patch, and year.
8. matrixqc: A vector used in summary statements to describe the overall quality of each matrix.
9. dataqc: A vector used in summary statements to describe key sampling aspects of the dataset (in raw MPMs).
10. modelqc: A vector used in summary statements to describe the vital rate models (in functionbased MPMs).

Objects within these lists may be called as in figure 1.5, below.


Figure 1.5: Structure of a lefkoMat object

We will detail the creation and meaning of these objects in later chapters, but for now let us construct a simple MPM as a lefkoMat object. For this, we will input the first six matrices of the MPM analyzed in Davison et al. (2010). This example will focus on just two populations of Anthyllis vulneraria, an herbaceous plant studied in south-western Belgium. First we will create the stageframe,
which summarizes the life history model of the plant (stageframes and life history models will be described in Chapter 2).

```
sizevector <- c(1, 1, 2, 3) # These sizes are not from the original paper
stagevector <- c("Sdl", "Veg", "SmFlo", "LFlo")
repvector <- c(0, 0, 1, 1)
obsvector <- c(1, 1, 1, 1)
matvector <- c(0, 1, 1, 1)
immvector <- c(1, 0, 0, 0)
propvector <- c(0, 0, 0, 0)
indataset <- c(1, 1, 1, 1)
binvec <- c(0.5, 0.5, 0.5, 0.5)
comments <- c("Seedling", "Vegetative adult", "Small flowering",
    "Large flowering")
anthframe <- sf_create(sizes = sizevector, stagenames = stagevector,
    repstatus = repvector, obsstatus = obsvector, matstatus = matvector,
    immstatus = immvector, indataset = indataset, binhalfwidth = binvec,
    propstatus = propvector, comments = comments)
```

class (anthframe)
> [1] "data.frame" "stageframe"
anthframe

| > 1 | Sdl | 1 | NA | NA | NA | NA | 0 |  | 1 |  |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| $>2$ | Veg | 1 | NA | NA | NA | NA | 0 |  | 1 |  |
| > 3 | SmFlo | 2 | NA | NA | NA | NA | 1 |  | 1 |  |
| > 4 | LFlo | 3 | NA | NA | NA | NA | 1 |  | 1 | 0 |
| > | immstatus matstatus indataset binhalfwidth_raw sizebin_min sizebin_max |  |  |  |  |  |  |  |  |  |
| > 1 |  | 1 | 0 |  |  |  |  | 0.5 |  | . 5 |
| $>2$ |  | 0 | 1 |  |  |  |  | 0.5 |  | . 5 |
| $>3$ |  | 0 | 1 |  |  |  |  | 1.5 |  | . 5 |
| > 4 |  | 0 | 1 |  |  |  |  | 2.5 |  | 3.5 |


| $>$ | sizebin_center | sizebin_width | binhalfwidthb_raw | sizebinb_min | sizebinb_max |
| :--- | ---: | ---: | ---: | ---: | ---: |
| $>$ | 1 | 1 | 1 | NA | NA |
| $\gg$ | 1 | 1 | NA | NA | NA |
| $>$ | 2 | 1 | NA | NA | NA |
| $>4$ | 3 | 1 | NA | NA | NA |

    sizebinb_center sizebinb_width binhalfwidthc_raw sizebinc_min sizebinc_max
    1 NA NA NA NA
    2 NA NA NA NA
3 NA NA NA NA
4 NA NA NA NA NA
> sizebinc_center sizebinc_width group comments
> 1 NA NA 0 Seedling
$>2$ NA NA O Vegetative adult
> 3 NA NA 0 Small flowering
> 4 NA NA 0 Large flowering

This object, called anthframe, is a data frame in $R$, and checking its class reveals that it is also a member of the class stageframe, which is a kind of data frame unique to lefko3 having the particular structure seen here. This object was created by developing a number of vectors describing the life
history stages that this plant lives through, and using these as inputs into the stageframe creation function, sf_create().

Now let's input three matrices each for two populations, both of which are part of a metapopulation. The three matrices show transitions across pairs of consecutive years, 2003-2004, 2004-2005, and 20052006. We input these matrices by row because the code supplied in Davison et al. (2010) was written in Matlab, which defaults to input by row.

```
# POPN C 2003-2004
XC3 <- matrix(c(0, 0, 1.74, 1.74,
    0.208333333, 0, 0, 0.057142857,
    0.041666667, 0.076923077, 0, 0,
    0.083333333, 0.076923077, 0.066666667, 0.028571429), 4, 4, byrow = TRUE)
# POPN C 2004-2005
XC4 <- matrix(c(0, 0, 0.3, 0.6,
0.32183908, 0.142857143, 0, 0,
0.16091954, 0.285714286, 0, 0,
0.252873563, 0.285714286, 0.5, 0.6), 4, 4, byrow = TRUE)
# POPN C 2005-2006
XC5 <- matrix(c(0, 0, 0.50625, 0.675,
0, 0, 0, 0.035714286,
0.1, 0.068965517, 0.0625, 0.107142857,
0.3, 0.137931034, 0, 0.071428571), 4, 4, byrow = TRUE)
# POPN E 2003-2004
XE3 <- matrix(c(0, 0, 2.44, 6.569230769,
0.196428571, 0, 0, 0,
0.125, 0.5, 0, 0,
0.160714286, 0.5, 0.133333333, 0.076923077), 4, 4, byrow = TRUE)
# POPN E 2004-2005
XE4 <- matrix(c(0, 0, 0.45, 0.646153846,
0.06557377, 0.090909091, 0.125, 0,
0.032786885, 0, 0.125, 0.076923077,
0.049180328, 0, 0.125, 0.230769231), 4, 4, byrow = TRUE)
# POPN E 2005-2006
XE5 <- matrix(c(0, 0, 2.85, 3.99,
0.083333333, 0, 0, 0,
0, 0, 0, 0,
0.416666667, 0.1, 0, 0.1), 4, 4, byrow = TRUE)
```

Let's take a quick look at one of these matrices.

```
XC3
> [,1] [,2] [,3] [,4]
> [1,] 0.00000000 0.00000000 1.74000000 1.74000000
> [2,] 0.20833333 0.00000000 0.00000000 0.05714286
> [3,] 0.04166667 0.07692308 0.00000000 0.00000000
> [4,] 0.08333333 0.07692308 0.06666667 0.02857143
```

Note that the structure is very conventional - the fecundity rates are up in the top-right corner,
while the rest of the elements showing transition probabilities and so are bound between 0 and 1 . All MPM matrices should contain only non-negative values, so we recommend inspecting your matrices when you construct them.

Next we will make a list of our matrices, and then use the create_lM() function to create our lefkoMat object. We will also supply vectors showing the order of patches in the list of matrices, and the order of year at time $t$ in the list of matrices (ahistorical matrices show transition rates from time $t$ to time $t+1$, so the matrix covering 2004-2005 is referred to by year 2004). Because lefko3 includes special methods to build and analyze historical MPMs, we will tell R that these matrices are not historical (historical = FALSE).

```
mats_list <- list(XC3, XC4, XC5, XE3, XE4, XE5)
anth_lefkoMat <- create_lM(mats_list, anthframe, historical = FALSE,
    patchorder = c(1, 1, 1, 2, 2, 2),
    yearorder = c(2003, 2004, 2005, 2003, 2004, 2005))
anth_lefkoMat
    $A
    $A[[1]]
> [,1] [,2] [,3] [,4]
> [1,] 0.00000000 0.00000000 1.74000000 1.74000000
> [2,] 0.20833333 0.00000000 0.00000000 0.05714286
> [3,] 0.04166667 0.07692308 0.00000000 0.00000000
> [4,] 0.08333333 0.07692308 0.06666667 0.02857143
>
    $A[[2]]
> [,1] [,2] [,3] [,4]
    [1,] 0.0000000 0.0000000 0.3 0.6
    [2,] 0.3218391 0.1428571 0.0
    [3,] 0.1609195 0.2857143 0.0 0.0
    [4,] 0.2528736 0.2857143 0.5 0.6
    $A[[3]]
            [,1] [,2] [,3] [,4]
    [1,] 0.0 0.00000000 0.50625 0.67500000
    [2,] 0.0 0.00000000 0.00000 0.03571429
    [3,] 0.1 0.06896552 0.06250 0.10714286
    [4,] 0.3 0.13793103 0.00000 0.07142857
    $A[[4]]
    [,1] [,2] [,3] [,4]
    [1,] 0.0000000 0.0 2.4400000 6.56923077
    [2,] 0.1964286 0.0 0.0000000 0.00000000
    [3,] 0.1250000 0.5 0.0000000 0.00000000
    [4,] 0.1607143 0.5 0.1333333 0.07692308
    $A[[5]]
    [,1] [,2] [,3] [,4]
    [1,] 0.00000000 0.00000000 0.450 0.64615385
    [2,] 0.06557377 0.09090909 0.125 0.00000000
    [3,] 0.03278689 0.00000000 0.125 0.07692308
    [4,] 0.04918033 0.00000000 0.125 0.23076923
```

```
>
> $A[[6]]
> [,1] [,2] [,3] [,4]
> [1,] 0.00000000 0.0 2.85 3.99
> [2,] 0.08333333 0.0 0.00 0.00
> [3,] 0.00000000 0.0 0.00 0.00
> [4,] 0.41666667 0.1 0.00 0.10
>
>
> $U
> $U[[1]]
> [,1] [,2] [,3] [,4]
> [1,] 0.00000000 0.00000000 0.00000000 0.00000000
> [2,] 0.20833333 0.00000000 0.00000000 0.05714286
> [3,] 0.04166667 0.07692308 0.00000000 0.00000000
> [4,] 0.08333333 0.07692308 0.06666667 0.02857143
>
> $U[[2]]
> [,1] [,2] [,3] [,4]
> [1,] 0.0000000 0.0000000 0.0 0.0
> [2,] 0.3218391
> [3,] 0.1609195 0.2857143 0.0 0.0
```



```
>
> $U[[3]]
> [,1] [,2] [,3] [,4]
> [1,] 0.0 0.00000000 0.0000 0.00000000
> [2,] 0.0 0.00000000 0.0000 0.03571429
> [3,] 0.1 0.06896552 0.0625 0.10714286
> [4,] 0.3 0.13793103 0.0000 0.07142857
>
> $U[[4]]
> [,1] [,2] [,3] [,4]
> [1,] 0.0000000 0.0 0.0000000 0.00000000
> [2,] 0.1964286 0.0 0.0000000 0.00000000
> [3,] 0.1250000 0.5 0.0000000 0.00000000
> [4,] 0.1607143 0.5 0.1333333 0.07692308
>
> $U[[5]]
> [,1] [,2] [,3] [,4]
> [1,] 0.00000000 0.00000000 0.000 0.00000000
> [2,] 0.06557377 0.09090909 0.125 0.00000000
> [3,] 0.03278689 0.00000000 0.125 0.07692308
> [4,] 0.04918033 0.00000000 0.125 0.23076923
>
> $U[[6]]
> [,1] [,2] [,3] [,4]
> [1,] 0.00000000 0.0 0 0.0
> [2,] 0.08333333 0.0 0 0.0
> [3,] 0.00000000 0.0 0 0.0
> [4,] 0.41666667 0.1 0}00.
```

```
>
>
> $F
$F[[1]]
            [,1] [,2] [,3] [,4]
> [1,] 0 0 1.74 1.74
> [2,] 0 0 0.00 0.00
> [3,] 0 0 0.00 0.00
> [4,] 0 0 0.00 0.00
>
> $F[[2]]
> [,1] [,2] [,3] [,4]
> [1,] 0 0 0.3 0.6
> [2,] 0 0}00.
> [3,] 0
> [4,] 0 0}00.
>
> $F[[3]]
> [,1] [,2] [,3] [,4]
> [1,] 0 0 0.50625 0.675
> [2,] 0 0 0.00000 0.000
> [3,] 0 0 0.00000 0.000
> [4,] 0 0 0.00000 0.000
>
> $F[[4]]
> [,1] [,2] [,3] [,4]
> [1,] 0 0 2.44 6.569231
> [2,] 0 0 0.00 0.000000
> [3,] 0 0 0.00 0.000000
> [4,] 0 0 0.00 0.000000
>
> $F[[5]]
> [,1] [,2] [,3] [,4]
> [1,] 0 0 0.45 0.6461538
> [2,] 0 0 0.00 0.0000000
> [3,] 0 0 0.00 0.0000000
> [4,] 0 0 0.00 0.0000000
>
> $F[[6]]
            [,1] [,2] [,3] [,4]
> [1,] 0 0 2.85 3.99
> [2,] 0 0 0.00 0.00
> [3,] 0 0 0.00 0.00
> [4,] 0 0 0.00 0.00
>
>
> $hstages
> [1] NA
>
> $agestages
> [1] NA
```

```
>
> $ahstages
> stage_id stage_id stage original_size original_size_b original_size_c min_age
> 1 1 1 Slllllll
> 2 
> 3 <rrrrlon
> max_age repstatus obsstatus propstatus immstatus matstatus entrystage
> 1 NA 
> 2 NA 0
> 3 N NA 
> indataset binhalfwidth_raw sizebin_min sizebin_max sizebin_center
> 1 1 0
> 2 1 0.5 0.5 0.5 1.5 l
> 3 1 0.5 0.5 1.5 2.5
\begin{tabular}{llllll}
\(>\) & 1 & 0.5 & 2.5 & 3.5 & 3
\end{tabular}
    sizebin_width binhalfwidthb_raw sizebinb_min sizebinb_max sizebinb_center
> 1 1 NA NA NA
> 2 1 NA NA NA
> 3 1 NA NA NA
>4 1 NA NA NA
> sizebinb_width binhalfwidthc_raw sizebinc_min sizebinc_max sizebinc_center
\begin{tabular}{llllll} 
& NA & NA & NA & NA
\end{tabular}
> NA NA NA NA
> NA NA NA NA
> 4 NA NA NA NA
sizebinc_width group comments alive almostborn
1 NA \(0 \quad\) Seedling 110
N NA 0 Vegetative adult 1 0
N NA 0 Small flowering 1 0
4 NA 0 Large flowering 1 0
$labels
pop patch year2
1 1 1 2003
2 1 1 2004
> 3 1 1 2005
> 4 1 2 2003
> 5 1 2 2004
> 6 1 2 2005
>
$matrixqc
[1] 44 12 6
$dataqc
[1] NA NA
>
> attr(,"class")
> [1] "lefkoMat"
```

Voilà! We are now free to use this MPM in our analyses. But before we do that we might wish
to explore this structure a bit more. The first element of this lefkoMat object, named A, is a list of the actual projection matrices. The create_1M() function uses the stageframe input that we provided to ascertain which transition rates are survival probabilities and which correspond to fecundity rates. The element named $U$ is a list including a survival-transition matrix for each $A$ matrix, meaning that fecundity is removed. The F element is a list including a fecundity matrix for each A matrix, meaning that transition probabilities have been removed. The element ahstages is our stageframe, though slightly edited and reordered. Here, element hstages is just NA, but would include the order of historical stage pairs if this MPM was historical. Element agestages is also NA, but would show us the order of age-stage combinations if this were an age-by-stage MPM or a Leslie MPM. The final element, matrixqc includes some basic quality control data in vector format. All of this is summarized using the summary () function, as below.

```
summary(anth_lefkoMat)
>
> This ahistorical lefkoMat object contains 6 matrices.
>
> Each matrix is square with 4 rows and columns, and a total of 16 elements.
> A total of 44 survival transitions were estimated, with 7.333 per matrix.
> A total of 12 fecundity transitions were estimated, with 2 per matrix.
> This lefkoMat object covers 1 population, 2 patches, and 3 time steps.
>
> This lefkoMat object appears to have been imported. Number of unique individuals and transitions nc
>
> Survival probability sum check (each matrix represented by column in order):
> [,1] [,2] [,3] [,4] [,5] [,6]
> Min. 0.0667 0.500 0.0625 0.0769 0.0909 0.000
> 1st Qu. 0.0810 0.575 0.1708 0.1192 0.1334 0.075
> Median 0.1198 0.657 0.2106 0.3077 0.2276 0.100
> Mean 0.1599 0.637 0.2209 0.4231 0.2303 0.175
> 3rd Qu. 0.1987 0.720 0.2607 0.6116 0.3245 0.200
> Max. 0.3333 0.736 0.4000 1.0000 0.3750 0.500
```

Summaries such as this are quite useful. Here we can see how many matrices we have (corresponding to A matrices), how big each matrix is, how many elements were estimated on average per matrix and across the board (the numbers of estimated elements shown actually shows us the number of nonzero elements, and so it is possible that the number of estimated elements is actually larger if some elements were estimated as 0 ), and how many populations, patches / subpopulations, and time steps are covered. This is followed by a summary of the column sums of the corresponding $U$ matrices this is an important quality control check because all numbers within the U matrices should conform to $0 \leq a_{i j} \leq 1$, where $a_{i j}$ is the element at row $i$ and column $j$, and all column sums within these matrices should also conform to $0 \leq \sum_{i} a_{i j} \leq 1$, where $\sum_{i} a_{i j}$ is the sum of all elements in column $j$ and corresponds to the survival probability of stage $j$.

### 1.8 Datasets used in this book

This book will utilize four main datasets. The first dataset is a data frame holding individual-level monitoring data on a population of the North American orchid species Cypripedium candidum, also known as the white lady's slipper. The second dataset is a data frame holding similar data for a population of the European perennial Lathyrus vernus. The third dataset is a lefkoMat object holding projection matrices from nine populations of from the European perennial Anthyllis vulneraria. The fourth dataset is a data frame holding demographic data on a population each of Pyrola japonica and Pyrola subaphylla growing in Fukushima Prefecture, Japan.

### 1.8.1 Cypripedium candidum data

This dataset is available in two formats, as data frames called cypdata and cypvert. These datasets contain the exact same information and are in different formats only. They can be called with the following code.

```
data(cypdata)
data(cypvert)
```

The white lady's slipper, Cypripedium candidum, is a North American perennial herb in the family Orchidaceae. It is long-lived and of conservation concern. This plant begins life by germinating from a dust seed, and then develops into a protocorm, which is a special subterranean life stage found in orchids and pyroloids. During this stage, the plant is non-photosynthetic and completely parasitic on its mycorrhizal fungi. It spends several years as a protocorm, and previous studies suggest that it typically spends 3 years before becoming a seedling. As a seedling, it may or may not produce aboveground sprouts, often remaining entirely subterranean and continuing its parasitic lifestyle. It may persist this way for many years before attaining adult size, at which point it may sprout with or without flowers, or may remain underground in a condition referred to as vegetative dormancy. The latter condition may continue for many years, with over a decade of continuous dormancy documented in the literature (Shefferson et al., 2018).


Figure 1.6: Field work in the Cypripedium candidum habitat yielding the demographic data included with package lefko3. Photo courtesy of R. Shefferson

The population from which the dataset originates is located within a wet meadow in a state nature preserve located in northeastern Illinois, USA (Figure 1.6). The population was monitored annually from 2004 to 2009 , with two monitoring sessions per year. Monitoring sessions took roughly 2 weeks each, and included complete censuses of the population divided into sections referred to as patches. Each monitoring session consisted of searches for previously recorded individuals, which were located according to coordinates relative to fixed stakes at the site, followed by a search for new individuals. Data recorded per individual included: the location, the number of non-flowering sprouts, the number of flowering sprouts, the number of flowers per flowering sprout, and the number of fruit pods per sprout (only in the second monitoring session per year, once fruiting had occurred). Location was used to infer individual identity. More information about this population and its characteristics is given in Shefferson et al. (2001) and Shefferson et al. (2017).

### 1.8.2 Lathyrus vernus data

This dataset is available as a data frame called lathyrus, and can be called with the following code.

```
data(lathyrus)
```

Lathyrus vernus (family Fabaceae) is a long-lived forest herb, native to Europe and large parts of northern Asia. Individuals increase slowly in size and usually flower only after 10-15 years of vegetative growth. Flowering individuals have an average conditional lifespan of 44.3 years (Ehrlén and Lehtila, 2002). L. vernus lacks organs for vegetative spread and individuals are well delimited (Ehrlén, 2002). One or several erect shoots of up to 40 cm height emerge from a subterranean rhizome in March-April. Flowering occurs about four weeks after shoot emergence. Shoot growth is determinate, and the number of flowers is determined in the previous year (Ehrlén and Van Groenendael, 2001). Individuals may not produce aboveground structures every year, but can remain dormant in one or more seasons. $L$. vernus is self-compatible but requires visits from bumble-bees to produce seeds. Individuals produce few, large seeds and establishment from seeds is relatively frequent (Ehrlén and Eriksson, 1996). The pre-dispersal seed predator Bruchus atomarius often consumes a large fraction of developing seeds, and roe deer (Capreolus capreolus) sometimes consume the shoots (Ehrlén and Munzbergova, 2009).

Data for this study were collected from six permanent plots in a population of L. vernus located in a deciduous forest in the Tullgarn area, SE Sweden (58.9496 N, 17.6097 E), during 1988-1991 (Ehrlén, 1995). The six plots were similar with regard to soil type, elevation, slope, and canopy cover. Within each plot, all individuals were marked with numbered tags that remained over the study period, and their locations were carefully mapped. New individuals were included in the study in each year. Individuals were recorded at least three times every growth season. At the time of shoot emergence, we recorded whether individuals were alive and produced above-ground shoots, and if shoots had been grazed. During flowering, we recorded flower number and the height and diameter of all shoots. At fruit maturation, we counted the number of intact and damaged seeds. To derive a measure of above-ground size for each individual, we calculated the volume of each shoot as $\pi \times\left(\frac{1}{2} \text { diameter }\right)^{2} \times h e i g h t$, and summed the volumes of all shoots. This measure is closely correlated with the dry mass of aboveground tissues $\left(R^{2}=0.924, P<0.001, n=50\right.$, log-transformed values; Ehrlén 1995). Size of individuals that had been grazed was estimated based on measures of shoot diameter in grazed shoots, and the relationship between shoot diameter and shoot height in non-grazed individuals. Only individuals with an aboveground volume of more than 230 mm 3 flowered and produced fruits during this study. Individuals that lacked aboveground structures in one season but reappeared in the following year were considered dormant. Individuals that lacked aboveground structures in two subsequent seasons were considered dead from the year in which they first lacked aboveground structures. Probabilities of seeds surviving to the next year, and of being present as seedlings or seeds in the soil seed bank, were derived from separate yearly sowing experiments in separate plots adjacent to each subplot (Ehrlén and Eriksson, 1996).

### 1.8.3 Anthyllis vulneraria data

Davison et al. (2010) reported stochastic contributions made by differences in vital rate means and variances among nine natural populations of Anthyllis vulneraria, also known as kidney vetch. This plant occurs in calcareous grasslands in the Viroin Valley of south-western Belgium. A. vulneraria is a grassland specialist and the unique host plant of the Red-listed blue butterfly (Cupido minimus). It is a short-lived, rosette-forming legume with a complex life cycle but no seedbank.

Nine populations $(\mathrm{N}=27-50,000)$ growing in distinct grassland fragments were surveyed between 2003 and 2006, yielding three $(4 \times 4)$ annual transition matrices for each population. The populations occurred within grassland fragments, and were mostly managed as nature reserves through rotational sheep grazing. These surveys coincided with a summer heat wave (2003), followed by a spring drought (2005) and an even more extreme heat wave (2006). These populations have been subject to detailed study for aspects of their genetics and demography, and further details on the sites can be obtained
through the resulting publications (Krauss et al., 2004; Honnay et al., 2006; Piessens et al., 2009). We use the matrices published in Davison et al. (2010) to illustrate some of the features of lefko3, and so provide these matrices in a lefkoMat object (section 1.7.1).

### 1.8.4 Pyrola data

This section is currently being developed.

### 1.9 Points to remember

1. $R$ is an object-oriented language using the atomic vector as its basic unit of analysis.
2. R Studio provides an easy-to-use, convenient working environment for R analyses, including analyses using lefko3.
3. Package lefko3 provides $R$ with convenient, standardized objects organizing all aspects of the MPM creation and analysis process. It also provides powerful functions that develop and analyze MPMs using lightning fast binaries.
4. Package lefko3 incorporates four key datasets for use in all examples, vignettes, and the chapters of this book.
5. Package lefko3 does not currently work with tidyverse structures. Please use base R and Matrix structures when using this package.

## Chapter 2

## Preliminaries I: Life History Models

"You know," said Arthur, "it's at times like this, when I'm trapped in a Vogon airlock with a man from Betelgeuse, and about to die of asphyxiation in deep space that I really wish I'd listened to what my mother told me when I was young."

- Douglas Adams, The Hitchhiker's Guide to the Galaxy

All organisms grow and develop. Development from birth proceeds through stages of life, some of which are easily defined and recognizable, while others are less so. For example, the transition from pupa to adult in many insects involves a distinct metamorphosis, making it easy to identify both the pupa and the adult. In contrast, childhood in humans is more difficult to define and its recognized, official end varies from culture to culture. Demography concerns itself with documenting and explaining these transitions, particularly the rates at which they occur.

Demographic analysis generally requires a model of development for the organism being studied, because vital rates generally differ across developmental stages. We call such a model a life history model. At the very least, this model needs to include the major, biologically notable stages of life, where a stage is defined as a portion of the life cycle defined by a unique set of biological, developmental, or demographic characteristics. The decision of what stages to use is somewhat subjective, and so the stages that a life history model includes may change given the circumstances of the research.

Developing a proper life history model is among the most difficult things to accomplish in demography. Indeed, although life history models underpin population ecology generally, and matrix projection model (MPM) analyses specifically, some analyses suggest that the majority of demographic analyses of MPMs utilize incorrect life history models (Kendall et al., 2019). These errors lead to biases that can misinform biological inference, with profound consequences for conservation management, evolutionary prediction, or whatever the goal of the study happens to be.

In this chapter, we consider how to develop a reasonable life history model for use in matrix population modeling using lefko3. We begin with a discussion of what is required of a good life history model for MPM analysis, including some of the more common errors in model development. We will then move on to create some life history models for use in examples later in this book. Our discussion will focus on the most commonly utilize life history model, the life cycle graph.

### 2.1 The life cycle graph

Matrix projection models (MPMs) are representations of the dynamics of a population across all life history stages deemed relevant, across the most relevant time interval (typically one year, but sometimes different, and assumed to be consistent within each analysis). Each MPM requires a complete model of the organism's life history prior to construction, and this model must explicitly show all life stages and all life history transitions that are biologically, realistically possible. Each stage is mutually
exclusive across time, meaning that an individual can only be in a single stage at a given time, and each transition takes exactly one full time step. The time step itself needs to be defined consistently and should be equivalent to the approximate amount of time between consecutive monitoring occasions. Each stage is represented in each matrix by a single column and a single row, and each transition by a single element. Matrix elements $\left(a_{k j}\right)$ show either the probability of transition for an individual in stage $j$ at occasion $t$ (along the columns), to stage $k$ at occasion $t+1$ (along the rows), or the mean rate of production of new recruits into stage $k$ at occasion $t+1$ (along the rows) by individuals in reproductive stage $j$ in occasion $t$ (along the columns). Conceptually, each individual is in a particular stage in the instance of monitoring or observation, and then either dies or completes a transition in the interval between that occasion's observation and the next occasion's observation. Death is not an explicit life stage and so is not modeled as such, instead becoming a potential endpoint of each transition.

Here we utilize a life cycle graph approach to build a life history model. A life cycle graph shows the life history of an organism as a series of nodes and arrows, using the simplifying assumptions that time is discrete and stages represent the state of the individual at each discrete point in time. The nodes, are typically shown as circles in the graph, and represent unique stages that the individual may go through during development. Individuals are noted as occurring in these stages during monitoring sessions conducted over a roughly or strictly regular frequency, such as at the start of the growing season every year, or over specific dates each year. Arrows represent transitions from stage to stage between consecutive occasions. These may be interpreted either as probabilities of survival-transition when reflecting the individual's stage across time, or as fecundity rates when reflecting the production of offspring by individuals in reproductive stages. A good life cycle graph, and in general a good life history model, must explicitly include representations of all stages possible for the organism to enter, and all biologically plausible transitions between these stages. Figure 2.1 (a) shows an example of a life cycle graph.


| b) |  | time $t$ |  |  |  |  |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: |
|  |  | From DS |  | From V F |  | From D From F |
|  | To DS | $\mathrm{S}_{5 s}$ | 0 | 0 | 0 | $\mathrm{F}_{\mathrm{SF}}$ |
|  | ToJ | $\mathrm{S}_{15}$ | $\mathrm{S}_{\mu}$ | 0 | 0 | $\mathrm{F}_{\mathrm{JF}}$ |
|  | To V | 0 | $\mathrm{S}_{\mathrm{vj}}$ | $\mathrm{S}_{\mathrm{vv}}$ | $S_{\text {vo }}$ | $S_{\text {VF }}$ |
|  | To D | 0 | $S_{\text {DJ }}$ | $\mathrm{S}_{\mathrm{Dv}}$ | $S_{\text {DD }}$ | $\mathrm{S}_{\mathrm{DF}}$ |
|  | To F | 0 | 0 | $\mathrm{S}_{\mathrm{FV}}$ | $\mathrm{S}_{\mathrm{FD}}$ | $\mathrm{S}_{\mathrm{FF}}$ |

Figure 2.1: Simple life history model (a) and ahistorical MPM (b) for Cypripedium candidum, a North American herbaceous plant species

Figure 2.1 is a simple example of a stage-based life history model viewed as a life cycle graph, and its associated Lefkovitch matrix for a terrestrial orchid species, Cypripedium candidum (Shefferson et al., 2001). Each stage is shown as a node, and each transition is shown as an uni-directional arrow (a). The rates and probabilities are shown as mathematical symbols in (b), with $S_{k j}$ denoting survivaltransition probability from stage $j$ in occasion $t$ to stage $k$ in occasion $t+1$, and $F_{k j}$ denoting the fecundity of reproductive stage $j$ in occasion $t$ into recruit stage $k$ in occasion $t+1$ in this life history.

Stages are generally defined under the assumption that they are unique moments in an organism's life. This uniqueness is assumed to extend to the demography of the organism during its time within that stage, meaning that the stage must have a mutually exclusive set of life history characteristics. In some cases, stages are defined as developmental stages, as happens with insect instars. In other cases, stages are defined almost purely on the basis of size, as occurs with perennial herbs that may exhibit a different number of stems, leaves, and flowers in each growing season. In still other cases, stages may
be defined via other characteristics, for example via age in a Leslie MPM. As another example, the vegetative dormancy stage of some perennial herbs is defined by a lack of aboveground size, and so is characteristically not observable and is defined by its lack of observability (Shefferson et al., 2001).

### 2.1.1 Stage duration

Stages always have durations, and these durations can influence the life history model chosen. A life history model used for MPM analysis needs to employ a consistent time interval for transitions, and so stages can only be defined if they occur at roughly the time resolution of the life history model. To think about stage duration, users may wish to consider a simple model that relates the life history model to the timing of monitoring. In this model, individuals exist in some particular stage at the instance of monitoring. Monitoring happens at regular intervals, and stages are defined in ways so that their durations correspond at least to the interval between consecutive monitoring occasions. Stage transitions can then be thought of conceptually as occurring either right before the monitoring occasion (the pre-breeding model), or immediately after (the post-breeding model). To simplify analysis, we also assume that death happens at the time of transition.

Erroneous accounting of stage duration is commonplace in MPM analyses. The most common problem is that one or more stages within a life history model are characterized by different durations than other stages. Often, a stage's actual duration is longer than acknowledged. As an example, we might wish to create a seedling stage that yields a small adult. We might initially create a simple three-stage life history in which a seed stage is followed by the seedling, and this is followed by a small adult stage, which may be capable of reproduction. Imagine, however, that the seedling stage is almost never shorter than two years, and that we have a dataset based on an annual census (i.e. our MPM will consist of annual matrices). In this circumstance, the seedling stage must be split into at least two seedling stages to force the analysis to include the correct minimum number of years to keep the organism a seedling. Failure to do so would create a model that assumes a shorter juvenile period than actually occurs, and would likely lead to biased estimates of population dynamics parameters, such as the asymptotic population growth rate.

Stage duration mismatch may also occur when actual stage durations are shorter than assumed by the model. For example, if I wish to study an insect that lives potentially up to ten years, and I will employ a matrix projection model in which matrices represent one year of transition, then I cannot divide the insect instars into multiple stages if they only last several days or weeks each in succession. Instead, I would need to lump those stages together in a way that incorporates a realistic amount of instar development over the course of a single year, and use that conglomerate stage within the life history model. Alternatively, I can deal with this issue by changing the set time interval used in the MPM, making a model with a much finer time resolution (although this would require monitoring many more times than just once a year, and might require redefining other stages).

### 2.1.2 The process and timing of fecundity

The timing of monitoring relative to the reproductive season greatly impacts the structure of life history models. Life history models are typically categorized as either pre-breeding or post-breeding. Here, breeding refers to the production of offspring, and so a pre-breeding model assumes that monitoring is conducted immediately before the new recruits are born, while a post-breeding model assumes that the monitoring is conducted just after new recruits are born (in both cases, breeding is assumed to be seasonal). In a pre-breeding model, no organism is monitored prior to an age of approximately one year, or one of whatever time unit is being used in the study. Further, in pre-breeding models fecundity equals the production of newborns multiplied by the survival of newborns to age 1 , since fecundity must take place over a full time step and the timing of the census misses both the birth event itself and the time in which the organism is a newborn (Kendall et al., 2019). In a post-breeding model, organisms are first monitored at age 0 , and fecundity equals the survival of the parent from
the stage/age preceding reproduction to reproduction itself, multiplied by the number of newborns produced (Kendall et al., 2019).

Population ecologists often mistakenly forget important components of fecundity because of the inherent complexity of life cycle models. Often, ecologists will estimate fecundity in pre-breeding studies erroneously as only the production of newborns, without multiplying by the survival of newborns to age 1 (Kendall et al., 2019). Ecologists working with post-breeding models often erroneously omit the survival of the parent from the stage immediately preceding reproduction, since including this term generally requires creating reproductive versions of all possible preceding stages even if, counterintuitively, they are not considered reproductive stages (Kendall et al., 2019). Both of these omissions result in biased analytical results, of which the most common problem is that population growth rate is estimated to be too high.

A comparison of the same system developed in pre-breeding vs. post-breeding format might help illustrate these issues. Figure 2.2 shows the same model as in figure 2.1, but given in both pre-breeding and post-breeding format. The flowering stage is the only reproductive stage in the pre-breeding model, and fecundity there is the expected number of seeds produced multiplied by either the probability of survival as a dormant seed (in production of dormant seed), or the probability of germination (in the production of juveniles). In the post-breeding model, the probability of survival from the stage preceding flowering is multiplied by the expected number of seeds, leading to a single offspring stage instead of two, and three forms of the flowering stage to deal with different expected fecundities in each case.


Figure 2.2: Simple life history pre-breeding model (a) and post-breeding model (b) for Cypripedium candidum, a North American herbaceous plant species. Stages include: S, seed; DS, dormant seed; J, juvenile (including protocorm and seedling); D, vegetatively dormant adult; V, sprouting but not flowering adult; F, flowering adult

In monitoring studies of plant populations, the typical strategy taken is that of a pre-breeding model, in which fecundity is estimated as the production of seeds in a given year multiplied by their over-winter survival probability as seeds and their germination probability in the following year. If the life history includes seed dormancy, then it can be modeled by multiplying seed production by seed survival (which is given as the probability of maintaining seed viability from one year to the next). In more complex situations, different seed survival probabilities may be set depending on whether the seed is freshly produced or dormant within the seedbank. Added complexity can arise if there are multiple fecundity pathways, for example when clonal reproduction is also possible, or if multiple propagule
stages exist, or if recruitment can yield seeds that immediately germinate as well as seeds that are dormant for one or more years. We urge users to be careful with this step, as properly defining the life history model has important implications for all analyses of population dynamics (Kendall et al., 2019).

### 2.1.3 Dataset and analytical influences on life history model design

Life history models can be influenced by factors seemingly external to the life history of the organism. Some of the most important considerations outside of the species' biology include the size of the dataset, and the decision of whether to create a raw MPM or a function-based MPM. All else being equal, a larger dataset and a function-based MPM will allow more stages to be used in the life history. Assuming a dataset of fixed size, raw MPMs will include more and more zeros as the number of stages in the life history model increases. This happens because the chance of having any individuals actually move through a specific transition decreases as the number of possible transitions increases. These increasingly common zeros can cause problems in analysis, because they suggest that some transitions are impossible when, in fact, individuals moving through them were missing just by chance. Even a transition with extremely high probability may be missing in some years just by chance. One impact of this would be the reduction of the mean transition within an element-wise mean matrix to an unrealistic level, causing both the survival probability of the associated stage and the population growth rate to be estimated too low.

Function-based MPMs are better able to handle large numbers of matrix elements because overparameterization is prevented by parsimonious model selection when vital rate models are determined. However, the size of the dataset influences the statistical power of these vital rate models, with smaller datasets yielding larger error associated with matrix elements. Loss of statistical power in vital rate models can lead to the loss of process variance, and can make population dynamics appear more static than they really are. In addition, adding more stages than necessary in the function-based case can create conditions in which stages themselves are estimated as having much higher survival probability than they actually do (this happens when stages that do not occur are modeled, as they are typically estimated as having trivial but non-zero probabilities of survival). The survival of a stage within a matrix is the column sum of for that stage in a survival-transition (i.e. a projection matrix without any fecundity terms). Vital rate functions generally produce non-zero values no matter what input is provided, even if sizes and conditions are entered that do not occur in the dataset, or that occur only rarely. Such situations may result in the artificial inflation of survival, and in extreme cases can result in predicted survival probability above 1.0. So, even in the function-based case, great effort needs to be taken to prevent the inclusion of too many stages, and of stages that do not actually occur in the dataset.

Chapter 6 in Morris and Doak (2002) provides a good description of techniques that may be used to define stages and to determine the exact number to use given the context imposed by a given dataset of a given size. Ellner and Rees (2006) provides some discussion of this issue within the context of integral projection model development.

Now let's move on to create a life history model and define it in R using lefko3.

### 2.2 Life history model development in lefko3

The basic workflow to analysis with package lefko3 starts with the development of a life history model that encapsulates all of the appropriate life stages relevant to population dynamics. We will illustrate life history model development with the Cypripedium candidum dataset described in section 1.8.1. This plant begins life by germinating from a dust seed, and then develops into a protocorm, which is a special subterranean life stage found in orchids and pyroloids. During this stage, the plant is non-photosynthetic and completely parasitic on its mycorrhizal fungi. It spends several years as a protocorm, and previous studies suggest that it typically spends three years before becoming a seedling.

As a seedling, it may or may not produce aboveground sprouts, often remaining entirely subterranean and continuing its parasitic lifestyle. It may persist this way for many years before attaining adult size, at which point it may sprout with or without flowers, or may remain underground in a condition referred to as vegetative dormancy. The latter condition may continue for many years, with over a decade of continuous dormancy documented in the literature (Shefferson et al., 2018).

We will now load lefko3 and the dataset, called cypdata, after clearing memory.

```
rm(list=ls(all=TRUE))
library(lefko3)
data(cypdata)
```

The first key decision in developing a life cycle graph is to decide on which life history stages to include. This is not a simple task. We might at first believe that any clearly defined developmental stage should be its own life history stage. However, life history stages need to be defined relative to the time interval used in the MPM. They also need to be defined uniquely, and to have transitions from other stages, leading either to further life history stages or back to themselves. Finally, they also need to be defined with consideration given to the size of the dataset, since smaller datasets will allow fewer stages.

It might be easiest to begin our discussion with consideration of the function-based MPM, and what sort of life history model works best if we choose this approach. With a function-based MPM (which includes the integral projection model, or IPM), there are really only two considerations that we need to be concerned about. First, what stages should be included based on a purely biological understanding of the organism's life history, and our decision on the time interval? Second, what is the range of stages and/or sizes that actually occur in the dataset and are typically encountered in nature? Let's discuss these points in turn.

The first point - taking the biology of the organism into account - seems intuitive, but it sometimes breaks down when we consider the time interval of the study. Key issues have already been mentioned the timing of reproduction relative to the monitoring occasion means that fecundity itself must include either newborn survival or the survival of new parents from the preceding stage, but not both. This means that the same life history might lead to a single reproductive stage if a pre-breeding model is used, or several if a post-breeding model is used (since reproduction would happen after each possible preceding stage). If newborns are propagules, such as seeds, then the same choices determine whether the first stage in the life cycle graph is a recruited juvenile (as in the pre-breeding case), or a propagule (as in the post-breeding case). Users should take great care with this step.

The second point - identifying stages that actually occur in the dataset and that should occur typically in nature - is also important. Some stages may never be monitored, and so these will have to be included in the model but dealt with via proxy rates later. In our Cypripedium case, for example, we never monitored germinated seeds, protocorms, and seedlings, because it is essentially impossible to do so. We will include these stages in our life history model, but we will use proxy rates for their survival transitions, since we cannot use our dataset to estimate them. Conversely, there may be some stages that appear to be outliers, appearing only once and representing unusually large sizes or other unusual conditions. Such stages should not be included, because their inclusion may yield odd impacts on analysis, such as the artificial inflation of stage survival.

The choice of observable life history stages is really focused on determining which survival and fecundity transitions we have the ability to estimate given our dataset. We should develop our life cycle graph with a reasonable range of sizes and stages in mind, since we do not wish to create stages in our life cycle model that go beyond the limits of what was actually observed (models that make predictions outside of the range of data they were parameterized with have a tendency to produce erroneous and at times egregiously strange predictions). So let's explore the range of sizes actually occurring in our dataset. There is no one variable that encapsulates the entire size of the individual in our dataset, so we will create a series of vectors that sums the numbers of sprouts that are single-
flowered flowering, double-flowered flowering, and non-flowering, and use these sums as plant sizes. If we use the horizontal dataset for this purpose, then we can assess size as follows.

```
size. }04<- cypdata$Inf2.04 + cypdata$Inf. 04 + cypdata$Veg. 04
size. }05\mathrm{ <- cypdata$Inf2.05 + cypdata$Inf. }05\mathrm{ + cypdata$Veg. }0
size.06 <- cypdata$Inf2.06 + cypdata$Inf.06 + cypdata$Veg. 06
size.07 <- cypdata$Inf2.07 + cypdata$Inf.07 + cypdata$Veg. }0
size. }08<- cypdata$Inf2.08 + cypdata$Inf. 08 + cypdata$Veg. 08
size.09 <- cypdata$Inf2.09 + cypdata$Inf. }09\mathrm{ + cypdata$Veg. }0
summary(c(size.04, size.05, size.06, size.07, size.08, size.09))
> Min. 1st Qu. Median Mean 3rd Qu. Max. NA's
> 1.000 1.000 2.000 3.644 5.000 24.000 156
```

The summary shows that the smallest recorded size is a single sprout, and the largest is 24 sprouts. The many NAs are a combination of vegetative dormancy and instances of individuals being dead or simply not yet born. Given this, we might utilize the following life history model for our function-based MPMs (figure 2.3).


Figure 2.3: Life history model of Cypripedium candidum for use in function-based MPMs

We can see a variety of transitions within this figure. The propagule and juvenile stages are the five stages in the top-left corner and have fairly simple transitions. New recruits may enter the population directly from the germination of a seed produced the previous year (Protocorm 1 stage), or they may begin as dormant seed (Dormant seed stage). Dormant seed may remain dormant, die, or germinate into the Protocorm 1 stage. Protocorms exist for up to three years without transitioning further, and individuals cannot have fewer than three years as protocorms, yielding the Protocorm 1, Protocorm 2, and Protocorm 3 stages. Protocorm 3 leads to the Seedling stage, in which the plant may persist for at least one year and potentially many years before becoming mature. Here,
maturity does not really refer to reproduction per se, but rather to a morphology indistinguishable from a reproductive plant except for the lack of a flower. The first mature stage is usually either vegetative dormancy (Dorm), during which time the plant does not sprout, or a small, non-flowering adult ( $\mathbf{1 V}$ ). Once in this mature portion of the life history (designated in the hashed blue box), the plant may transition among 49 mature stages, including vegetative dormancy, 1-24 shoots without flowers, or 1-24 shoots with at least one flower.

### 2.2.1 Building the stageframe

Now that we have our life history model, we will need to describe the life history characterizing the dataset, matching it to our analyses properly with a stageframe for our Cypripedium candidum dataset. This is a vitally important step, and most instances of errors occurring in the use of lefko3 originate from an inappropriate stageframe being used in an analysis. Since this analysis will be function-based, we will include all possible size classes here. If constructing raw matrices, all sizes that occur in the dataset need to be accounted for in a way that is both natural and parsimonious with respect to the numbers of individuals moving through actual transitions. If constructing functionbased matrices, such as IPMs, then representative sizes at systematic increments will be satisfactory. Since size is count-based in the Cypripedium candidum case, we will use all numbers of stems that might occur from zero to the maximum in the dataset, representing the life history diagram shown in the beginning of this chapter.

To start, we will create two vectors, one showing the names of all stages in our life history model, and one showing whether the stage is included in the dataset. Both vectors utilize the same order of stages.

```
stagevector <- c("SD", "P1", "P2", "P3", "SL", "D", "V1", "V2", "V3", "V4",
    "V5", "V6", "V7", "V8", "V9", "V10", "V11", "V12", "V13", "V14", "V15", "V16",
    "V17", "V18", "V19", "V20", "V21", "V22", "V23", "V24", "F1", "F2", "F3",
    "F4", "F5", "F6", "F7", "F8", "F9", "F10", "F11", "F12", "F13", "F14", "F15",
    "F16", "F17", "F18", "F19", "F20", "F21", "F22", "F23", "F24")
indataset <- c(0, 0, 0, 0, 0, rep(1, 49))
```

We included all stages in this step, and the indataset vector allows us to tell R that the first five stages ("SD", "P1", "P2", "P3", "SL") do not actually exist in the dataset. Next, we will create a few more vectors to describe the life history fully. The vectors will include the representative size (typically though not necessarily the mean or median size), reproductive status, observation status (i.e. whether it is technically possible to observe the stage, or whether the stage is essentially invisible and must be inferred to be present), and maturity status of each of these stages.

```
sizevector <- c(0, 0, 0, 0, 0, seq(from = 0, t = 24), seq(from = 1, to = 24))
repvector <- c(0, 0, 0, 0, 0, rep(0, 25), rep(1, 24))
obsvector <- c(0, 0, 0, 0, 0, 0, rep(1, 48))
matvector <- c(0, 0, 0, 0, 0, rep(1, 49))
```

Now we will create just three more vectors. The first will be a vector for status as an immature stage. While this vector might seem to be simply the logical inverse and so the mathematical complement to matvector, in truth there are cases where a stage may be both mature and immature, depending on the circumstance, and cases where a stage may actually be neither. One stage that is neither is $\mathbf{S D}$, the dormant seed stage - that is marked as 0 in both cases. The second vector is a vector showing status as a propagule stage. The final vector is a comment vector, providing text descriptions of the stages.

```
immvector <- c(0, 1, 1, 1, 1, rep(0, 49))
propvector <- c(1, rep(0, 53))
comments <- c("Dormant seed", "Yr1 protocorm", "Yr2 protocorm", "Yr3 protocorm",
    "Seedling", "Veg dorm", "Veg adult 1 stem", "Veg adult 2 stems",
    "Veg adult 3 stems", "Veg adult 4 stems", "Veg adult 5 stems",
    "Veg adult 6 stems", "Veg adult 7 stems", "Veg adult 8 stems",
    "Veg adult 9 stems", "Veg adult 10 stems", "Veg adult 11 stems",
    "Veg adult 12 stems", "Veg adult 13 stems", "Veg adult 14 stems",
    "Veg adult 15 stems", "Veg adult 16 stems", "Veg adult 17 stems",
    "Veg adult 18 stems", "Veg adult 19 stems", "Veg adult 20 stems",
    "Veg adult 21 stems", "Veg adult 22 stems", "Veg adult 23 stems",
    "Veg adult 24 stems", "Flo adult 1 stem", "Flo adult 2 stems",
    "Flo adult 3 stems", "Flo adult 4 stems", "Flo adult 5 stems",
    "Flo adult 6 stems", "Flo adult 7 stems", "Flo adult 8 stems",
    "Flo adult }9\mathrm{ stems", "Flo adult }10\mathrm{ stems", "Flo adult 11 stems",
    "Flo adult 12 stems", "Flo adult 13 stems", "Flo adult 14 stems",
    "Flo adult 15 stems", "Flo adult 16 stems", "Flo adult 17 stems",
    "Flo adult 18 stems", "Flo adult 19 stems", "Flo adult 20 stems",
    "Flo adult 21 stems", "Flo adult 22 stems", "Flo adult 23 stems",
    "Flo adult 24 stems")
```

Next we will create the stageframe using the sf_create() function.

| cypframe_fb |  |  |  |  |  |  |  |  |  |  |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
|  | stage | ize | size_b | size | min_age | nax_age | repstatus | obsstatus |  | opstatus |
| $>1$ | SD | 0 | NA | NA | NA | NA | 0 | 0 |  | 1 |
| $>2$ | P1 | 0 | NA | NA | NA | NA | 0 | 0 |  | 0 |
| $>3$ | P2 | 0 | NA | NA | NA | NA | 0 | 0 |  | 0 |
| $>4$ | P3 | 0 | NA | NA | NA | NA | 0 | 0 |  | 0 |
| $>5$ | SL | 0 | NA | NA | NA | NA | 0 | 0 |  | 0 |
| $>6$ | D | 0 | NA | NA | NA | NA | 0 | 0 |  | 0 |
| $>7$ | V1 | 1 | NA | NA | NA | NA | 0 | 1 | 1 | 0 |
| $>8$ | V2 | 2 | NA | NA | NA | NA | 0 | 1 | 1 | 0 |
| $>9$ | V3 | 3 | NA | NA | NA | NA | 0 | 1 | 1 | 0 |
| > 10 | V4 | 4 | NA | NA | NA | NA | 0 | 1 | 1 | 0 |
| > 11 | V5 | 5 | NA | NA | NA | NA | 0 | 1 | 1 | 0 |
| > 12 | V6 | 6 | NA | NA | NA | NA | 0 | 1 | 1 | 0 |
| > 13 | V7 | 7 | NA | NA | NA | NA | 0 | 1 | 1 | 0 |
| > 14 | V8 | 8 | NA | NA | NA | NA | 0 | 1 | 1 | 0 |
| > 15 | V9 | 9 | NA | NA | NA | NA | 0 | 1 | 1 | 0 |
| > 16 | V10 | 10 | NA | NA | NA | NA | 0 | 1 | 1 | 0 |
| > 17 | V11 | 11 | NA | NA | NA | NA | 0 | 1 | 1 | 0 |
| > 18 | V12 | 12 | NA | NA | NA | NA | 0 | 1 | 1 | 0 |
| > 19 | V13 | 13 | NA | NA | NA | NA | 0 | 1 | 1 | 0 |
| > 20 | V14 | 14 | NA | NA | NA | NA | 0 | 1 | 1 | 0 |
| > 21 | V15 | 15 | NA | NA | NA | NA | 0 | 1 | 1 | 0 |
| > 22 | V16 | 16 | NA | NA | NA | NA | 0 | 1 |  | 0 |
| > 23 | V17 | 17 | NA | NA | NA | NA | 0 | 1 | 1 | 0 |



| > 5 | 0 | 1 | NA | NA | NA |
| :---: | :---: | :---: | :---: | :---: | :---: |
| $>6$ | 0 | 1 | NA | NA | NA |
| $>7$ | 1 | 1 | NA | NA | NA |
| $>8$ | 2 | 1 | NA | NA | NA |
| $>9$ | 3 | 1 | NA | NA | NA |
| > 10 | 4 | 1 | NA | NA | NA |
| > 11 | 5 | 1 | NA | NA | NA |
| > 12 | 6 | 1 | NA | NA | NA |
| $>13$ | 7 | 1 | NA | NA | NA |
| > 14 | 8 | 1 | NA | NA | NA |
| > 15 | 9 | 1 | NA | NA | NA |
| $>16$ | 10 | 1 | NA | NA | NA |
| $>17$ | 11 | 1 | NA | NA | NA |
| $>18$ | 12 | 1 | NA | NA | NA |
| > 19 | 13 | 1 | NA | NA | NA |
| > 20 | 14 | 1 | NA | NA | NA |
| > 21 | 15 | 1 | NA | NA | NA |
| > 22 | 16 | 1 | NA | NA | NA |
| $>23$ | 17 | 1 | NA | NA | NA |
| $>24$ | 18 | 1 | NA | NA | NA |
| > 25 | 19 | 1 | NA | NA | NA |
| > 26 | 20 | 1 | NA | NA | NA |
| > 27 | 21 | 1 | NA | NA | NA |
| > 28 | 22 | 1 | NA | NA | NA |
| > 29 | 23 | 1 | NA | NA | NA |
| > 30 | 24 | 1 | NA | NA | NA |
| > 31 | 1 | 1 | NA | NA | NA |
| > 32 | 2 | 1 | NA | NA | NA |
| > 33 | 3 | 1 | NA | NA | NA |
| > 34 | 4 | 1 | NA | NA | NA |
| sizebinb_center sizebinb_width binhalfwidthc_raw sizebinc_min sizebinc_max |  |  |  |  |  |
| > 1 | NA | NA | NA | NA | NA |
| $>2$ | NA | NA | NA | NA | NA |
| $>3$ | NA | NA | NA | NA | NA |
| $>4$ | NA | NA | NA | NA | NA |
| $>5$ | NA | NA | NA | NA | NA |
| $>6$ | NA | NA | NA | NA | NA |
| > 7 | NA | NA | NA | NA | NA |
| $>8$ | NA | NA | NA | NA | NA |
| > 9 | NA | NA | NA | NA | NA |
| > 10 | NA | NA | NA | NA | NA |
| $>11$ | NA | NA | NA | NA | NA |
| > 12 | NA | NA | NA | NA | NA |
| $>13$ | NA | NA | NA | NA | NA |
| > 14 | NA | NA | NA | NA | NA |
| > 15 | NA | NA | NA | NA | NA |
| > 16 | NA | NA | NA | NA | NA |
| $>17$ | NA | NA | NA | NA | NA |
| $>18$ | NA | NA | NA | NA | NA |
| > 19 | NA | NA | NA | NA | NA |
| > 20 | NA | NA | NA | NA | NA |


| > 21 | NA | NA | NA | NA | NA |
| :---: | :---: | :---: | :---: | :---: | :---: |
| > 22 | NA | NA | NA | NA | NA |
| > 23 | NA | NA | NA | NA | NA |
| > 24 | NA | NA | NA | NA | NA |
| > 25 | NA | NA | NA | NA | NA |
| > 26 | NA | NA | NA | NA | NA |
| > 27 | NA | NA | NA | NA | NA |
| > 28 | NA | NA | NA | NA | NA |
| > 29 | NA | NA | NA | NA | NA |
| > 30 | NA | NA | NA | NA | NA |
| > 31 | NA | NA | NA | NA | NA |
| > 32 | NA | NA | NA | NA | NA |
| > 33 | NA | NA | NA | NA | NA |
| > 34 | NA | NA | NA | NA | NA |
|  | sizebinc_center | sizebinc_width | group comments |  |  |
| $>1$ | NA | NA | 0 Dormant seed |  |  |
| $>2$ | NA | NA | 0 Yr1 protocorm |  |  |
| $>3$ | NA | NA | 0 Yr2 protocorm |  |  |
| $>4$ | NA | NA | 0 Yr3 protocorm |  |  |
| $>5$ | NA | NA | 0 Seedling |  |  |
| $>6$ | NA | NA | 0 Veg dorm |  |  |
| $>7$ | NA | NA | 0 Veg adult 1 stem |  |  |
| $>8$ | NA | NA | 0 Veg adult 2 stems |  |  |
| > 9 | NA | NA | 0 Veg adult 3 stems |  |  |
| > 10 | NA | NA | 0 Veg adult 4 stems |  |  |
| > 11 | NA | NA | 0 Veg adult 5 stems |  |  |
| > 12 | NA | NA | 0 Veg adult 6 stems |  |  |
| > 13 | NA | NA | 0 Veg adult 7 stems |  |  |
| > 14 | NA | NA | 0 Veg adult 8 stems |  |  |
| > 15 | NA | NA | 0 Veg adult 9 stems |  |  |
| > 16 | NA | NA | 0 Veg adult 10 stems |  |  |
| > 17 | NA | NA | 0 Veg adult 11 stems |  |  |
| > 18 | NA | NA | 0 Veg adult 12 stems |  |  |
| > 19 | NA | NA | 0 Veg adult 13 stems |  |  |
| > 20 | NA | NA | 0 Veg adult 14 stems |  |  |
| > 21 | NA | NA | 0 Veg adult 15 stems |  |  |
| > 22 | NA | NA | 0 Veg adult 16 stems |  |  |
| > 23 | NA | NA | 0 Veg adult 17 stems |  |  |
| > 24 | NA | NA | 0 Veg adult 18 stems |  |  |
| > 25 | NA | NA | 0 Veg adult 19 stems |  |  |
| > 26 | NA | NA | 0 Veg adult 20 stems |  |  |
| > 27 | NA | NA | 0 Veg adult 21 stems |  |  |
| > 28 | NA | NA | 0 Veg adult 22 stems |  |  |
| > 29 | NA | NA | 0 Veg adult 23 stems |  |  |
| > 30 | NA | NA | 0 Veg adult 24 stems |  |  |
| > 31 | NA | NA | 0 Flo adult 1 stem |  |  |
| > 32 | NA | NA | 0 Flo adult 2 stems |  |  |
| > 33 | NA | NA | 0 Flo adult 3 stems |  |  |
| > 34 | NA | NA | 0 Flo adult 4 stems |  |  |
| > [ | reached 'max' / | getOption("max | print") -- omitted 20 row | ] |  |

A close look at the resulting object, cypframe, shows a data frame that includes the following
information in order for each stage: the stage's name, the associated size (in terms of up to three size metrics), its minimum and maximum age of occurrence (only used if creating an age-by-stage MPM), its reproductive status, its status as an observable stage, its status as a propagule stage, its status as an immature stage, its status as a mature stage, whether it occurs in the dataset, the half-width of its size class bin, the minimum and maximum of its size class bin, the centroid of its size class bin (currently the arithmetic mean), its full size class bin width (these last three variables are given for up to three size metrics in order), the stage group that the stage belongs to, and a comments field describing the stage. Stage names and combinations of characteristics for all stages occurring in the dataset must be unique to prevent estimation errors, and the comments field may be edited to include any information deemed pertinent. Note that we did not include information on secondary and tertiary size, stage group, and the ages at which stages occur, but these may be supplied as well. We will cover these topics later in this book.

At this stage, we can build our function-based MPM. However, we might ask: How does the life history model differ if we wish to develop a raw MPM? The key difference is that we need to consider how large our dataset is, and to create only as many stages as can be routinely transitioned to and from based on the data. The life history model above in Figure 2.3, for example, is not usable for a raw MPM because we have cut the size bins too finely - it is likely that in a typical year, only some of these stages will have individuals actually transitioning between them. The impact of this is that we will end up with many zeros for transitions that, in a sufficiently large population, should not equal zero, and this fact will force our estimates of the population growth rate lower than necessary.

To deal with this problem, we need to explore the dataset to determine a reasonable number of life history stages and where the breaks should occur between these stages. A number of means exist to do this, and users should see Caswell (2001) and Kendall et al. (2019) for good discussions of the topic. Here, we suggest plotting a size distribution and assessing a series of numbers of natural breaks using the Jenks natural breaks algorithm (Jenks, 1967), or another such algorithm. Let's take a look at a distribution plot of size to help us with this process (figure 2.4).
plot(density(c(size.04, size.05, size.06, size.07, size.08, size.09),
na.rm = TRUE), main = "", xlab = "Size (\# of sprouts)", bty = "n")


Figure 2.4: Distribution of size in Cypripedium candidum
We can see that most individuals are small, and so our size data is densest around 1-2 sprouts or so. Large individuals are rare, so we will need to make size bins wider for large plants than for small plants. Let's try finding some natural breaks with the Jenks algorithm, separating into three, four, five, and six stages. To separate the size data into these numbers of stages, we need to identify a total of four, five, six, and seven breaks including the minimum and maximum. First, let's install the package BAMMtools, which includes the getJenksBreaks() function.

```
install.packages("BAMMtools", dependencies = TRUE)
```

Now let's see where we might fit natural breaks, and how many there might need to be.

```
BAMMtools::getJenksBreaks(c(size.04, size.05, size.06, size.07, size.08,
    size.09), k = 4)
> [1] 1 1 4 12 24
BAMMtools::getJenksBreaks(c(size.04, size.05, size.06, size.07, size.08,
    size.09), k = 5)
> [1] 1 3 3 7 14 24
BAMMtools::getJenksBreaks(c(size.04, size.05, size.06, size.07, size.08,
    size.09), k = 6)
> [1] 1 1 2 4 4 7 7 14 24
BAMMtools::getJenksBreaks(c(size.04, size.05, size.06, size.07, size.08,
    size.09), k = 7)
> [1]
```

The Jenks method gives us the borders of the size classes under different numbers of stages. The first line shows a three stage model, yielding four breaks including the minimum and maximum. This model has the first stage include 1-3 sprouts, the second stage include $4-11$ sprouts, and the 3 rd stage include 12-24 sprouts. The fourth line is a six stage model with seven breaks shown. Given what we know about the size distribution, we will try to separate the data into five stages (one sprout, 2-4 sprouts, $5-7$ sprouts, $8-14$ sprouts, and $15-24$ sprouts), which will result in 11 total life history stages in our life history (one dormant seed, three protocorm stages of different age, one seedling stage, one vegetatively dormant stage, and five size-classified adult stages). Here is our new life history model (figure 2.5). Note that the hashed blue box here refers to the mature stages that are actually observable in our population.


Figure 2.5: Life history model of Cypripedium candidum for use in raw MPMs

Let's now build a stageframe using these breaks. We will build vectors designating the same kinds of information as before, plus a new vector designating the half-width of the size bins associated with each stage (the binvec vector). The default half-width is 0.5 , and while we used the default for the function-based stageframe, we cannot assume the default here.

```
sizevector <- c(0, 0, 0, 0, 0, 0, 1, 3, 6, 11, 19.5)
stagevector <- c("SD", "P1", "P2", "P3", "SL", "D", "XSm", "Sm", "Md", "Lg",
    "XLg")
repvector <- c(0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1)
obsvector <- c(0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1)
matvector <- c(0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1)
immvector <- c(0, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0)
propvector <- c(1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
indataset <- c(0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1)
binvec <- c(0, 0, 0, 0, 0, 0.5, 0.5, 1.5, 1.5, 3.5, 5)
comments <- c("Dormant seed", "1st yr protocorm", "2nd yr protocorm",
    "3rd yr protocorm", "Seedling", "Dormant adult",
    "Extra small adult (1 shoot)", "Small adult (2-4 shoots)",
    "Medium adult (5-7 shoots)", "Large adult (8-14 shoots)",
    "Extra large adult (>14 shoots)")
cypframe_raw <- sf_create(sizes = sizevector, stagenames = stagevector,
    repstatus = repvector, obsstatus = obsvector, matstatus = matvector,
    propstatus = propvector, immstatus = immvector, indataset = indataset,
    binhalfwidth = binvec, comments = comments)
```




The above method used bin-halfwidths together with bin midpoints to define the boundaries of each bin. This generally requires some mental mathematics, and also the acceptance of the bin midpoint as the representative size. However, in reality the bin midpoint is just a representative size for the class and is not actually used in calculations. So, users may stipulate different representative sizes, as long as they supply the size minima and maxima. In the code below, we develop an alternative stageframe with the same bins used for size, but different representative sizes. Note the use of the sizemin and sizemax options. The resulting stageframe will result in the exact same MPM, whether applied to raw or function-based cases.

```
sizevector_alt <- c(0, 0, 0, 0, 0, 0, 1, 4, 7, 14, 24)
sizeminvec <- c(0, 0, 0, 0, 0, -0.5, 0.5, 2.5, 4.5, 7.5, 14.5)
sizemaxvec <- c(0, 0, 0, 0, 0, 0.5, 1.5, 4.5, 7.5, 14.5, 24.5)
cypframe_raw_alt <- sf_create(sizes = sizevector_alt, stagenames = stagevector,
    repstatus = repvector, obsstatus = obsvector, matstatus = matvector,
```

```
propstatus = propvector, immstatus = immvector, indataset = indataset,
sizemin = sizeminvec, sizemax = sizemaxvec, comments = comments)
```

cypframe_raw_alt

> sizebin_center sizebin_width binhalfwidthb_raw sizebinb_min sizebinb_max

| $>1$ | 0.0 | 0 | NA | NA | NA |
| :--- | :--- | :--- | :--- | :--- | :--- |
| $>2$ | 0.0 | 0 | NA | NA | NA |
| $>3$ | 0.0 | 0 | NA | NA | NA |
| $>4$ | 0.0 | 0 | NA | NA | NA |
| $>5$ | 0.0 | 0 | NA | NA | NA |
| $>6$ | 0.0 | 1 | NA | NA | NA |
| $>7$ | 1.0 | 1 | NA | NA | NA |
| $>8$ | 3.5 | 2 | NA | NA | NA |
| $>9$ | 6.0 | 3 | NA | NA | NA |
| $>10$ | 11.0 | 7 | NA | NA | NA |
| $>11$ | 19.5 | 10 | NA | NA |  |


| $>$ | sizebinb_center sizebinb_width binhalfwidthc_raw sizebinc_min sizebinc_max |  |  |  |  |
| :--- | ---: | ---: | ---: | ---: | ---: |
| $>1$ | NA | NA | NA | NA | NA |
| $>2$ | NA | NA | NA | NA | NA |
| $>3$ | NA | NA | NA | NA | NA |
| $>4$ | NA | NA | NA | NA |  |
| $>5$ | NA | NA | NA | NA | NA |
| $>6$ | NA | NA | NA | NA |  |
| $>7$ | NA | NA | NA | NA | NA |
| $>8$ | NA | NA | NA | NA | NA |
| $>9$ | NA | NA | NA | NA | NA |
| $>10$ | NA | NA | NA | NA | NA |



### 2.3 Stage classification with multiple size metrics

So far, we have seen the simplest case of stage classification with size, where each stage is defined by a mutually exclusive size bin of a single size variable. However, more complicated cases certainly exist, in which stages should be defined on the basis of multiple size variables. Let's rethink the Cypripedium candidum example with this in mind.

In the previous two stageframes, we defined stages on the basis of the number of sprouts. However, many ecologists would argue that a flowering sprout is biologically quite different from a non-flowering sprout. Certainly, the amount of resources needed to produce a flowering sprout should be greater than that required for a non-flowering sprout. Of course, flowering sprouts actually produce offspring as well, making them demographically different from non-flowering sprouts. Further, the population actually includes an occasional, rare two-flowered sprout, as well. Can we redefine our life history model to use the numbers of non-flowering, one-flowered, and two-flowered sprouts to define stages?

To conduct this exercise, we first need to explore our data to see what combinations of different numbers of these three kinds of sprouts actually exist. We will do that using the verticalize3() function to standardize and format our demographic data first. We will describe this function in detail in the next chapter and so will not do so here. However, we will just point out that the three kinds of sprouts are input as sizeacol, sizebcol, and sizeccol in this function call, with Veg. 04 designating the number of non-flowering sprouts in 2004, Inf. 04 designating the number of singleflowered sprouts in 2004, and Inf2.04 designating the number of double-flowered sprouts in 2004 (other years are incorporated via the blocksize option, but more on that in the next chapter).

```
vert.data.check <- verticalize3(cypdata, noyears = 6, firstyear = 2004,
    patchidcol = "patch", individcol = "plantid", blocksize = 4,
    sizeacol = "Veg.04", sizebcol = "Inf.04", sizeccol = "Inf2.04",
    repstracol = "Inf.04", repstrbcol = "Inf2.04", fecacol = "Pod.04",
    censorcol = "censor", censorkeep = 1, censorRepeat = FALSE,
    NAasO = TRUE, age_offset = 4, censor = FALSE)
summary(vert.data.check)
> rowid popid patchid individ year2
> Min. : 1.00 Length:320 A: 93 Min. : 164.0 Min. :200
> 1st Qu.:21.00 Class :character B:154 1st Qu.: 391.0 1st Qu.:2005
> Median :37.50 Mode :character C: 73 Median : 453.0 Median :2006
> Mean :38.45 Mean : 651.5 Mean :2006
> 3rd Qu.:56.00 3rd Qu.: 476.0 3rd Qu.:2007
> Max. :77.00 Max. :1560.0 Max. :2008
```




The resulting object is a data frame, and we need to see what sorts of occurrences of each size actually occur within this data frame. For this purpose, we will use the xtabs() function in R's stats package to create contingency tables of the different sizes at time $t$.

```
xtabs(~ sizea2 + sizeb2 + sizec2, vert.data.check)
```

xtabs(~ sizea2 + sizeb2 + sizec2, vert.data.check)
>, , sizec2 = 0
>, , sizec2 = 0
>
>
> sizeb2
> sizeb2
> sizea2
> sizea2
>
>
>
>
>
>
>
>
>
>
>
>
>
>
>
>
>

```
>
```

| > | 9 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| > | 10 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| > | 11 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| > | 12 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| > | 13 | 0 | 2 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| $\gg$ |  |  |  |  |  |  |  |  |  |  |  |  |  |
| >, , sizec2 = 1 |  |  |  |  |  |  |  |  |  |  |  |  |  |
| $>$ |  |  |  |  |  |  |  |  |  |  |  |  |  |
| > |  | ize |  |  |  |  |  |  |  |  |  |  |  |
| > | sizea2 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 10 | 11 | 18 |
| > | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| > | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| > | 2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 2 | 0 | 0 | 0 |
| > | 3 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| > | 4 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| > | 5 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| > | 6 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| > | 7 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| > | 8 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| > | 9 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| > | 10 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| > | 11 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| > | 12 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| > | 13 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Clearly, not all size combinations are possible, and many combinations never occur. If we set up all combinations of sizes, including even a stage for 13 non-flowering sprouts and 18 single-flowered sprouts, then that stage will represent something that never actually happened in our dataset and may never happen in nature. The resulting matrices would likely include unrealistic statistics for these stages, potentially including artificially high survival. So, we will only build stages for combinations that seem likely. We might view these stages as occurring above a diagonal line running across the top contingency table, perhaps from around 11 or 12 non-flowering sprouts with no flowering sprouts, to 10 or 11 single-flowered sprouts with no non-flowering sprouts. Only the smallest two-flowered combinations will be included, as well.

```
sizevector.f <- c(0, 0, 0, 0, 0, 0, seq(1, 12, by = 1), seq(0, 9, by = 1),
    seq(0, 8, by = 1), seq(0, 7, by = 1), seq(0, 6, by = 1), seq(0, 5, by = 1),
    seq(0, 4, by = 1), seq(0, 3, by = 1), 0, 1, 2, 0, 1, 0,
    0, 0, 1,0)
sizebvector.f <- c(0, 0, 0, 0, 0, 0, rep(0, 12), rep(1, 10), rep(2, 9),
    rep(3, 8), rep(4, 7), rep(5, 6), rep(6, 5), rep(7, 4), rep(8, 3), 9, 9, 10,
    0, 1, 1, 2)
sizecvector.f <- c(0, 0, 0, 0, 0, 0, rep(0, 12), rep(0, 10), rep(0, 9),
    rep(0, 8), rep(0, 7), rep(0, 6), rep(0, 5), rep(0, 4), 0, 0, 0, 0, 0, 0,
    1, 1, 1, 1)
stagevector.f <- c("DS", "P1", "P2", "P3", "Sdl", "Dorm", "V1 IO D0",
    "V2 IO DO", "V3 IO DO", "V4 IO D0", "V5 IO DO", "V6 IO DO", "V7 IO DO",
    "V8 IO D0", "V9 IO D0", "V10 IO D0", "V11 IO D0", "V12 IO DO", "VO I1 DO",
    "V1 I1 D0", "V2 I1 D0", "V3 I1 D0", "V4 I1 D0", "V5 I1 D0", "V6 I1 D0",
    "V7 I1 D0", "V8 I1 D0", "V9 I1 D0", "V0 I2 D0", "V1 I2 D0", "V2 I2 D0",
    "V3 I2 D0", "V4 I2 D0", "V5 I2 D0", "V6 I2 D0", "V7 I2 D0", "V8 I2 DO",
    "V0 I3 D0", "V1 I3 D0", "V2 I3 D0", "V3 I3 D0", "V4 I3 D0", "V5 I3 D0",
```

```
    "V6 I3 D0", "V7 I3 D0", "V0 I4 D0", "V1 I4 D0", "V2 I4 D0", "V3 I4 DO",
    "V4 I4 D0", "V5 I4 D0", "V6 I4 D0", "V0 I5 D0", "V1 I5 D0", "V2 I5 DO",
    "V3 I5 D0", "V4 I5 D0", "V5 I5 D0", "V0 I6 D0", "V1 I6 D0", "V2 I6 DO",
    "V3 I6 D0", "V4 I6 D0", "V0 I7 D0", "V1 I7 D0", "V2 I7 D0", "V3 I7 D0",
    "V0 I8 D0", "V1 I8 D0", "V2 I8 D0", "V0 I9 DO", "V1 I9 DO", "V0 I10 D0",
    "V0 IO D1", "V0 I1 D1", "V1 I1 D1", "V0 I2 D1")
repvector.f <- c(0, 0, 0, 0, 0, rep(0, 13), rep(1, 59))
obsvector.f <- c(0, 0, 0, 0, 0, 0, rep(1, 71))
matvector.f <- c(0, 0, 0, 0, 0, rep(1, 72))
immvector.f <- c(0, 1, 1, 1, 1, rep(0, 72))
propvector.f <- c(1, rep(0, 76))
indataset.f <- c(0, 0, 0, 0, 0, rep(1, 72))
binvec.f <- c(0, 0, 0, 0, 0, rep(0.5, 72))
binbvec.f <- c(0, 0, 0, 0, 0, rep(0.5, 72))
bincvec.f <- c(0, 0, 0, 0, 0, rep(0.5, 72))
```

```
vertframe.3f <- sf_create(sizes = sizevector.f, sizesb = sizebvector.f,
```

vertframe.3f <- sf_create(sizes = sizevector.f, sizesb = sizebvector.f,
sizesc = sizecvector.f, stagenames = stagevector.f, repstatus = repvector.f,
sizesc = sizecvector.f, stagenames = stagevector.f, repstatus = repvector.f,
obsstatus = obsvector.f, propstatus = propvector.f, immstatus = immvector.f,
obsstatus = obsvector.f, propstatus = propvector.f, immstatus = immvector.f,
matstatus = matvector.f, indataset = indataset.f, binhalfwidth = binvec.f,
matstatus = matvector.f, indataset = indataset.f, binhalfwidth = binvec.f,
binhalfwidthb = binbvec.f, binhalfwidthc = bincvec.f)

```
    binhalfwidthb = binbvec.f, binhalfwidthc = bincvec.f)
```

| > | stage | size | size_b | size_c | min_age | max_age | repstatus | obsstatus | propstatus |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| > 1 | DS | - | 0 | 0 | NA | NA | 0 | 0 | 1 |
| $>2$ | P1 | 0 | 0 | 0 | NA | NA | 0 | 0 | 0 |
| $>3$ | P2 | 0 | 0 | 0 | NA | NA | 0 | 0 | 0 |
| $>4$ | P3 | 0 | 0 | 0 | NA | NA | 0 | 0 | 0 |
| $>5$ | Sdl | 0 | 0 | 0 | NA | NA | 0 | 0 | 0 |
| $>6$ | Dorm | 0 | 0 | 0 | NA | NA | 0 | 0 | 0 |
| > 7 | V1 IO D0 | 1 | 0 | 0 | NA | NA | 0 | 1 | 0 |
| $>8$ | V2 IO D0 | 2 | 0 | 0 | NA | NA | 0 | 1 | 0 |
| > 9 | V3 IO D0 | 3 | 0 | 0 | NA | NA | 0 | 1 | 0 |
| > 10 | V4 I0 D0 | 4 | 0 | 0 | NA | NA | 0 | 1 | 0 |
| > 11 | V5 I0 D0 | 5 | 0 | 0 | NA | NA | 0 | 1 | 0 |
| > 12 | V6 I0 D0 | 6 | 0 | 0 | NA | NA | 0 | 1 | 0 |
| > 13 | V7 I0 D0 | 7 | 0 | 0 | NA | NA | 0 | 1 | 0 |
| > 14 | V8 I0 D0 | 8 | 0 | 0 | NA | NA | 0 | 1 | 0 |
| > 15 | V9 IO D0 | 9 | 0 | 0 | NA | NA | 0 | 1 | 0 |
| > 16 | V10 IO D0 | 10 | 0 | 0 | NA | NA | 0 | 1 | 0 |
| > 17 | V11 IO DO | 11 | 0 | 0 | NA | NA | 0 | 1 | 0 |
| > 18 | V12 IO D0 | 12 | 0 | 0 | NA | NA | 0 | 1 | 0 |
| > 19 | VO I1 DO | 0 | 1 | 0 | NA | NA | 1 | 1 | 0 |
| > 20 | V1 I1 D0 | 1 | 1 | 0 | NA | NA | 1 | 1 | 0 |
| > 21 | V2 I1 D0 | 2 | 1 | 0 | NA | NA | 1 | 1 | 0 |
| > 22 | V3 I1 D0 | 3 | 1 | 0 | NA | NA | 1 | 1 | 0 |
| > 23 | V4 I1 D0 | 4 | 1 | 0 | NA | NA | 1 | 1 | 0 |
| > 24 | V5 I1 D0 | 5 | 1 | 0 | NA | NA | 1 | 1 | 0 |
| > 25 | V6 I1 D0 | 6 | 1 | 0 | NA | NA | 1 | 1 | 0 |
| > 26 | V7 I1 D0 | 7 | 1 | 0 | NA | NA | 1 | 1 | 0 |
| > 27 | V8 I1 D0 | 8 | 1 | 0 | NA | NA | 1 | 1 | 0 |



| $>9$ | 3 | 1 | 0.5 | -0.5 | 0.5 |
| :---: | :---: | :---: | :---: | :---: | :---: |
| > 10 | 4 | 1 | 0.5 | -0.5 | 0.5 |
| > 11 | 5 | 1 | 0.5 | -0.5 | 0.5 |
| > 12 | 6 | 1 | 0.5 | -0.5 | 0.5 |
| $>13$ | 7 | 1 | 0.5 | -0.5 | 0.5 |
| > 14 | 8 | 1 | 0.5 | -0.5 | 0.5 |
| > 15 | 9 | 1 | 0.5 | -0.5 | 0.5 |
| > 16 | 10 | 1 | 0.5 | -0.5 | 0.5 |
| > 17 | 11 | 1 | 0.5 | -0.5 | 0.5 |
| > 18 | 12 | 1 | 0.5 | -0.5 | 0.5 |
| > 19 | 0 | 1 | 0.5 | 0.5 | 1.5 |
| > 20 | 1 | 1 | 0.5 | 0.5 | 1.5 |
| > 21 | 2 | 1 | 0.5 | 0.5 | 1.5 |
| > 22 | 3 | 1 | 0.5 | 0.5 | 1.5 |
| > 23 | 4 | 1 | 0.5 | 0.5 | 1.5 |
| > 24 | 5 | 1 | 0.5 | 0.5 | 1.5 |
| > 25 | 6 | 1 | 0.5 | 0.5 | 1.5 |
| > 26 | 7 | 1 | 0.5 | 0.5 | 1.5 |
| > 27 | 8 | 1 | 0.5 | 0.5 | 1.5 |
| > 28 | 9 | 1 | 0.5 | 0.5 | 1.5 |
| > 29 | 0 | 1 | 0.5 | 1.5 | 2.5 |
| > 30 | 1 | 1 | 0.5 | 1.5 | 2.5 |
| > 31 | 2 | 1 | 0.5 | 1.5 | 2.5 |
| > 32 | 3 | 1 | 0.5 | 1.5 | 2.5 |
| > 33 | 4 | 1 | 0.5 | 1.5 | 2.5 |
| > 34 | 5 | 1 | 0.5 | 1.5 | 2.5 |
| > | sizebinb_center | sizebinb_width | binhalfwidthc_raw | sizebinc_min | sizebinc_max |
| $>1$ | 0 | 0 | 0.0 | 0.0 | 0.0 |
| $>2$ | 0 | 0 | 0.0 | 0.0 | 0.0 |
| $>3$ | 0 | 0 | 0.0 | 0.0 | 0.0 |
| $>4$ | 0 | 0 | 0.0 | 0.0 | 0.0 |
| $>5$ | 0 | 0 | 0.0 | 0.0 | 0.0 |
| $>6$ | 0 | 1 | 0.5 | -0.5 | 0.5 |
| $>7$ | 0 | 1 | 0.5 | -0.5 | 0.5 |
| > 8 | 0 | 1 | 0.5 | -0.5 | 0.5 |
| > 9 | 0 | 1 | 0.5 | -0.5 | 0.5 |
| > 10 | 0 | 1 | 0.5 | -0.5 | 0.5 |
| > 11 | 0 | 1 | 0.5 | -0.5 | 0.5 |
| > 12 | 0 | 1 | 0.5 | -0.5 | 0.5 |
| > 13 | 0 | 1 | 0.5 | -0.5 | 0.5 |
| > 14 | 0 | 1 | 0.5 | -0.5 | 0.5 |
| > 15 | 0 | 1 | 0.5 | -0.5 | 0.5 |
| > 16 | 0 | 1 | 0.5 | -0.5 | 0.5 |
| > 17 | 0 | 1 | 0.5 | -0.5 | 0.5 |
| > 18 | 0 | 1 | 0.5 | -0.5 | 0.5 |
| > 19 | 1 | 1 | 0.5 | -0.5 | 0.5 |
| > 20 | 1 | 1 | 0.5 | -0.5 | 0.5 |
| > 21 | 1 | 1 | 0.5 | -0.5 | 0.5 |
| > 22 | 1 | 1 | 0.5 | -0.5 | 0.5 |
| > 23 | 1 | 1 | 0.5 | -0.5 | 0.5 |
| > 24 | 1 | 1 | 0.5 | -0.5 | 0.5 |


| > 25 | 1 | 1 |  | 0.5 | -0.5 | 0.5 |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| > 26 | 1 | 1 |  | 0.5 | -0.5 | 0.5 |
| > 27 | 1 | 1 |  | 0.5 | -0.5 | 0.5 |
| $>28$ | 1 | 1 |  | 0.5 | -0.5 | 0.5 |
| > 29 | 2 | 1 |  | 0.5 | -0.5 | 0.5 |
| > 30 | 2 | 1 |  | 0.5 | -0.5 | 0.5 |
| > 31 | 2 | 1 |  | 0.5 | -0.5 | 0.5 |
| > 32 | 2 | 1 |  | 0.5 | -0.5 | 0.5 |
| > 33 | 2 | 1 |  | 0.5 | -0.5 | 0.5 |
| > 34 | 2 | 1 |  | 0.5 | -0.5 | 0.5 |
| $>$ | sizebinc_center sizebinc_width group |  |  | comments |  |  |
| $>1$ | 0 | 0 |  | No description |  |  |
| $>2$ | 0 | 0 |  | O No description |  |  |
| $>3$ | 0 | 0 |  | No description |  |  |
| $>4$ | 0 | 0 |  | O No description |  |  |
| $>5$ | 0 | 0 |  | No description |  |  |
| $>6$ | 0 | 1 |  | No description |  |  |
| $>7$ | 0 | 1 |  | No description |  |  |
| $>8$ | 0 | 1 |  | No description |  |  |
| $>9$ | 0 | 1 |  | No description |  |  |
| > 10 | 0 | 1 |  | No description |  |  |
| > 11 | 0 | 1 |  | No description |  |  |
| > 12 | 0 | 1 |  | No description |  |  |
| $>13$ | 0 | 1 |  | No description |  |  |
| > 14 | 0 | 1 |  | No description |  |  |
| $>15$ | 0 | 1 |  | No description |  |  |
| $>16$ | 0 | 1 |  | No description |  |  |
| $>17$ | 0 | 1 |  | No description |  |  |
| > 18 | 0 | 1 |  | No description |  |  |
| > 19 | 0 | 1 |  | No description |  |  |
| > 20 | 0 | 1 |  | No description |  |  |
| > 21 | 0 | 1 |  | O No description |  |  |
| > 22 | 0 | 1 |  | O No description |  |  |
| $>23$ | 0 | 1 |  | No description |  |  |
| > 24 | 0 | 1 |  | No description |  |  |
| $>25$ | 0 | 1 |  | No description |  |  |
| > 26 | 0 | 1 |  | O No description |  |  |
| > 27 | 0 | 1 |  | No description |  |  |
| $>28$ | 0 | 1 |  | O No description |  |  |
| > 29 | 0 | 1 |  | No description |  |  |
| > 30 | 0 | 1 |  | No description |  |  |
| > 31 | 0 | 1 |  | No description |  |  |
| > 32 | 0 | 1 |  | No description |  |  |
| $>33$ | 0 | 1 |  | No description |  |  |
| > 34 | 0 | 1 |  | 0 No description |  |  |
| > [ | reached 'max' / | getOption("max | .print | ") -- omitted 43 |  |  |

The result is our biggest stageframe yet, but a realistic one that we will go back to later.

### 2.4 Automating the creation of large numbers of stages

There may be times when we wish to create extremely large life history models, with many tens of life stages. For example, users wishing to develop IPMs may wish to create stageframes with over 100 stages, most of which would differ only by size. Developing stageframes for these situations can take some time and effort. Fortunately, lefko3 includes a method to make this quick and easy.

The key innovation is to use an option in function sf_create() that allows us to set the minimum and maximum sizes for a group of stages that are alike in every way except for size. A further option can set the number of total size classes to fit within these bounds. Let's try an example, using the Cypripedium candidum case.

The size metrics that we will use in the Cypripedium candidum dataset are all count variables. However, let us pretend that the total number of sprouts is actually a continuous, quantitative variable with a minimum of 1 and a maximum of 23 . If we wished to create 100 stages within these bounds, we could do so by identifying the minimum and maximum size values with the "ipm" marker in the stage names. The ipmbins option defaults to 100 , but let us say so explicitly here. We will then look at just a few key variables in the resulting stageframe.

```
sizevector <- c(0, 0, 0, 0, 0, 0, 1, 23)
stagevector <- c("SD", "P1", "P2", "P3", "SL", "D", "ipm", "ipm")
repvector <- c(0, 0, 0, 0, 0, 0, 1, 1)
obsvector <- c(0, 0, 0, 0, 0, 0, 1, 1)
matvector <- c(0, 0, 0, 0, 0, 1, 1, 1)
immvector <- c(0, 1, 1, 1, 1, 0, 0, 0)
propvector <- c(1, 0, 0, 0, 0, 0, 0, 0)
indataset <- c(0, 0, 0, 0, 0, 1, 1, 1)
binvec <- c(0, 0, 0, 0, 0, 0.5, NA, NA)
cypframe_ipm <- sf_create(sizes = sizevector, stagenames = stagevector,
    repstatus = repvector, obsstatus = obsvector, matstatus = matvector,
    propstatus = propvector, immstatus = immvector, indataset = indataset,
    binhalfwidth = binvec, ipmbins = 100)
cypframe_ipm[,c("stage", "size", "sizebin_min", "sizebin_max")]
> stage size sizebin_min sizebin_max
> 1 SD 0.00 0.00 0.00
>2 P1 0.00 
> 3 P2 0.00 P
>4 P3 0.00 0.00 0.00
> 5 SL 0.00 0.00 0.00
> 6 N 0.00 
> 7 sza_1.1100_0 1.11 1.00 1.22
> 8 sza_1.3300_0 1.33 1.22 1.44
> 9 sza_1.5500_0 1.55 1.44 1.66
> 10 sza_1.7700_0 1.77 1.66 1.88
> 11 sza_1.9900_0 1.99 1.88 2.10
> 12 sza_2.2100_0 2.21 2.10 2.32
> 13 sza_2.4300_0 2.43 2.32 2.54
> 14 sza_2.6500_0 2.65 2.54 2.76
> 15 sza_2.8700_0 2.87 2.76 2.98
> 16 sza_3.0900_0 3.09 2.98 3.20
> 17 sza_3.3100_0 3.31 3.20 3.42
> 18 sza_3.5300_0 3.53 3.42 3.64
```

| > 19 | sza_3.7500_0 3.75 | 3.64 | 3.86 |
| :---: | :---: | :---: | :---: |
| > 20 | sza_3.9700_0 3.97 | 3.86 | 4.08 |
| > 21 | sza_4.1900_0 4.19 | 4.08 | 4.30 |
| > 22 | sza_4.4100_0 4.41 | 4.30 | 4.52 |
| > 23 | sza_4.6300_0 4.63 | 4.52 | 4.74 |
| > 24 | sza_4.8500_0 4.85 | 4.74 | 4.96 |
| > 25 | sza_5.0700_0 5.07 | 4.96 | 5.18 |
| > 26 | sza_5.2900_0 5.29 | 5.18 | 5.40 |
| > 27 | sza_5.5100_0 5.51 | 5.40 | 5.62 |
| > 28 | sza_5.7300_0 5.73 | 5.62 | 5.84 |
| > 29 | sza_5.9500_0 5.95 | 5.84 | 6.06 |
| > 30 | sza_6.1700_0 6.17 | 6.06 | 6.28 |
| > 31 | sza_6.3900_0 6.39 | 6.28 | 6.50 |
| > 32 | sza_6.6100_0 6.61 | 6.50 | 6.72 |
| > 33 | sza_6.8300_0 6.83 | 6.72 | 6.94 |
| > 34 | sza_7.0500_0 7.05 | 6.94 | 7.16 |
| > 35 | sza_7.2700_0 7.27 | 7.16 | 7.38 |
| > 36 | sza_7.4900_0 7.49 | 7.38 | 7.60 |
| > 37 | sza_7.7100_0 7.71 | 7.60 | 7.82 |
| > 38 | sza_7.9300_0 7.93 | 7.82 | 8.04 |
| > 39 | sza_8.1500_0 8.15 | 8.04 | 8.26 |
| > 40 | sza_8.3700_0 8.37 | 8.26 | 8.48 |
| > 41 | sza_8.5900_0 8.59 | 8.48 | 8.70 |
| > 42 | sza_8.8100_0 8.81 | 8.70 | 8.92 |
| > 43 | sza_9.0300_0 9.03 | 8.92 | 9.14 |
| > 44 | sza_9.2500_0 9.25 | 9.14 | 9.36 |
| > 45 | sza_9.4700_0 9.47 | 9.36 | 9.58 |
| > 46 | sza_9.6900_0 9.69 | 9.58 | 9.80 |
| > 47 | sza_9.9100_0 9.91 | 9.80 | 10.02 |
| > 48 | sza_10.130_0 10.13 | 10.02 | 10.24 |
| > 49 | sza_10.350_0 10.35 | 10.24 | 10.46 |
| > 50 | sza_10.570_0 10.57 | 10.46 | 10.68 |
| > 51 | sza_10.790_0 10.79 | 10.68 | 10.90 |
| > 52 | sza_11.010_0 11.01 | 10.90 | 11.12 |
| > 53 | sza_11.230_0 11.23 | 11.12 | 11.34 |
| > 54 | sza_11.450_0 11.45 | 11.34 | 11.56 |
| > 55 | sza_11.670_0 11.67 | 11.56 | 11.78 |
| > 56 | sza_11.890_0 11.89 | 11.78 | 12.00 |
| > 57 | sza_12.110_0 12.11 | 12.00 | 12.22 |
| > 58 | sza_12.330_0 12.33 | 12.22 | 12.44 |
| > 59 | sza_12.550_0 12.55 | 12.44 | 12.66 |
| > 60 | sza_12.770_0 12.77 | 12.66 | 12.88 |
| > 61 | sza_12.990_0 12.99 | 12.88 | 13.10 |
| > 62 | sza_13.210_0 13.21 | 13.10 | 13.32 |
| > 63 | sza_13.430_0 13.43 | 13.32 | 13.54 |
| > 64 | sza_13.650_0 13.65 | 13.54 | 13.76 |
| > 65 | sza_13.870_0 13.87 | 13.76 | 13.98 |
| > 66 | sza_14.090_0 14.09 | 13.98 | 14.20 |
| > 67 | sza_14.310_0 14.31 | 14.20 | 14.42 |
| > 68 | sza_14.530_0 14.53 | 14.42 | 14.64 |
| > 69 | sza_14.750_0 14.75 | 14.64 | 14.86 |


| 70 | sza_14.970_0 14.97 | 14.86 | 15.08 |
| :---: | :---: | :---: | :---: |
| > 71 | sza_15.190_0 15.19 | 15.08 | 15.30 |
| $>72$ | sza_15.410_0 15.41 | 15.30 | 15.52 |
| > 73 | sza_15.630_0 15.63 | 15.52 | 15.74 |
| 74 | sza_15.850_0 15.85 | 15.74 | 15.96 |
| > 75 | sza_16.070_0 16.07 | 15.96 | 16.18 |
| 76 | sza_16.290_0 16.29 | 16.18 | 16.40 |
| 77 | sza_16.510_0 16.51 | 16.40 | 16.62 |
| 78 | sza_16.730_0 16.73 | 16.62 | 16.84 |
| 79 | sza_16.950_0 16.95 | 16.84 | 17.06 |
| 80 | sza_17.170_0 17.17 | 17.06 | 17.28 |
| 81 | sza_17.390_0 17.39 | 17.28 | 17.50 |
| 82 | sza_17.610_0 17.61 | 17.50 | 17.72 |
| > 83 | sza_17.830_0 17.83 | 17.72 | 17.94 |
| > 84 | sza_18.050_0 18.05 | 17.94 | 18.16 |
| > 85 | sza_18.270_0 18.27 | 18.16 | 18.38 |
| > 86 | sza_18.490_0 18.49 | 18.38 | 18.60 |
| > 87 | sza_18.710_0 18.71 | 18.60 | 18.82 |
| > 88 | sza_18.930_0 18.93 | 18.82 | 19.04 |
| > 89 | sza_19.150_0 19.15 | 19.04 | 19.26 |
| > 90 | sza_19.370_0 19.37 | 19.26 | 19.48 |
| > 91 | sza_19.590_0 19.59 | 19.48 | 19.70 |
| > 92 | sza_19.810_0 19.81 | 19.70 | 19.92 |
| 93 | sza_20.030_0 20.03 | 19.92 | 20.14 |
| > 94 | sza_20.250_0 20.25 | 20.14 | 20.36 |
| > 95 | sza_20.470_0 20.47 | 20.36 | 20.58 |
| > 96 | sza_20.690_0 20.69 | 20.58 | 20.80 |
| > 97 | sza_20.910_0 20.91 | 20.80 | 21.02 |
| > 98 | sza_21.130_0 21.13 | 21.02 | 21.24 |
| > 99 | sza_21.350_0 21.35 | 21.24 | 21.46 |
| > 100 | sza_21.570_0 21.57 | 21.46 | 21.68 |
| > 101 | sza_21.790_0 21.79 | 21.68 | 21.90 |
| > 102 | sza_22.010_0 22.01 | 21.90 | 22.12 |
| > 103 | sza_22.230_0 22.23 | 22.12 | 22.34 |
| > 104 | sza_22.450_0 22.45 | 22.34 | 22.56 |
| > 105 | sza_22.670_0 22.67 | 22.56 | 22.78 |
| > 106 | sza_22.890_0 22.89 | 22.78 | 23.00 |

We see that we have created a total of 106 stages - six early-life stages, followed by 100 stages that were developed automatically. These stages have size bins that do not overlap with neighboring sizes, and cover all possible sizes in the range.

We will see more uses of this approach in chapter 7 on integral projection models. However, it is important to point out that this approach can be used for multiple groups of stages, each with its own unique combinations of stage characteristics and with their own size minima and maxima. This approach can also be used in multiple size classification scenarios. More on this later.

### 2.5 Advanced stageframe creation

Finally, users who have developed stageframes and wish for extra flexibility without the error-checking functions of function sf_create() may also use function sf_skeleton() to create a skeleton stageframe to edit by hand. For example, imagine that we wished to create a raw MPM stageframe this
way, using the same values as before. Let's first create the skeleton stageframe itself.


| > | sizebinc_center | sizebinc_width | group | comments |
| :---: | :---: | :---: | :---: | :---: |
| $>1$ | 0 | 0 | 0 | stage 1 comment |
| $>2$ | 0 | 0 | 0 | stage 2 comment |
| $>3$ | 0 | 0 | 0 | stage 3 comment |
| $>4$ | 0 | 0 | 0 | stage 4 comment |
| $>5$ | 0 | 0 | 0 | stage 5 comment |
| $>6$ | 0 | 0 | 0 | stage 6 comment |
| $>7$ | 0 | 0 | 0 | stage 7 comment |
| $>8$ | 0 | 0 | 0 | stage 8 comment |
| $>9$ | 0 | 0 | 0 | stage 9 comment |
| > 10 | 0 | 0 |  | stage 10 comment |
| > 11 | 0 | 0 | 0 | stage 11 comment |

The resulting object is generally full of standard default values that need to be edited. Let's edit the vectors as necessary, as below.

```
alt_sf$size <- c(0, 0, 0, 0, 0, 0, 1, 3, 6, 11, 19.5)
alt_sf$stage <- c("SD", "P1", "P2", "P3", "SL", "D", "XSm", "Sm", "Md", "Lg",
    "XLg")
alt_sf$repstatus <- c(0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1)
alt_sf$obsstatus <- c(0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1)
alt_sf$matstatus <- c(0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1)
alt_sf$immstatus <- c(0, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0)
alt_sf$propstatus <- c(1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
alt_sf$indataset <- c(0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1)
alt_sf$binhalfwidth_raw <- c(0, 0, 0, 0, 0, 0.5, 0.5, 1.5, 1.5, 3.5, 5)
alt_sf$sizebin_min <- alt_sf$size - alt_sf$binhalfwidth_raw
alt_sf$sizebin_max <- alt_sf$size + alt_sf$binhalfwidth_raw
alt_sf$sizebin_center <- alt_sf$size
alt_sf$comments <- c("Dormant seed", "1st yr protocorm", "2nd yr protocorm",
    "3rd yr protocorm", "Seedling", "Dormant adult",
    "Extra small adult (1 shoot)", "Small adult (2-4 shoots)",
    "Medium adult (5-7 shoots)", "Large adult (8-14 shoots)",
    "Extra large adult (>14 shoots)")
alt_sf
```




At this point, we have five stageframes and can move ahead to standardizing and formatting our demographic datasets and creating some MPMs with them. There are further options that we can use to develop more complicated life history models, including the use of stage grouping to help properly format MPMs, and the designation of ages for each stage for use in age-by-stage MPMs. We will explore such issues in later chapters. For further information on life history model development, see chapters 3 and 4 in Caswell (2001) for a good treatment useful for beginners. We also note Kendall et al.
(2019) as a good reference detailing common problems in MPM construction and how to avoid them through proper development and operationalization of life history models. Beissinger and Westphal (1998), Wardle (1998), and Salguero-Gómez and Casper (2010) provide good discussions of the proper application of life history models to understand population dynamics through the MPM approach.

### 2.6 Points to remember

1. All MPMs and IPMs require appropriate and carefully thought out models of the life history of the organism.
2. Good life history models require appropriately defined life history stages, which are defined not simply by their size and reproductive characteristics, but also by their stage duration, their relationships to other stages, by the actual numbers of different combinations of life history characteristics occurring in the dataset, and by the choice of raw vs. function-based MPM.
3. Function sf_create() will develop a data frame identifying each stage in the life history model, and this data frame will be used at each step of the MPM development and analysis process.
4. Function sf_create() can also be used to define large suites of stages using shorthand designations for groups of stages. This is particularly useful in developing discretized IPMs.
5. Function sf_skeleton() can be used to create a skeleton stageframe that may be edited as desired, without the error-checking capabilities of function sf_create().

## Chapter 3

## Preliminaries II: Data Formatting

The most merciful thing in the world, I think, is the inability of the human mind to correlate all its contents.

- H.P. Lovecraft, The Call of Cthulhu

Matrix projection models are estimated with demographic data. Regardless of whether the goal is a set of raw (empirical) or function-based matrices, the dataset used needs to follow a particular format to make the estimation of elements or vital rate models possible. Package lefko3 utilizes a standardized format similar to formats required by other protocols, such as the projection.matrix() function in package popbio. However, unlike other packages, lefko3 includes functions that can standardize a variety of starting dataset formats into the right format. These functions likely do not cover all possibilities, but deal with most of the interesting differences among the protocols that different demographers use.

The format used in analysis is similar to what one would expect in setting up a data frame for a linear model in R or another statistical analysis environment. If we start with a record of observations of a single individual across several years, where each observation was conducted during a single monitoring session per time interval (such as year), then our data needs to be formatted in a way that allows us to assess at the very least whether the state of an individual at some time depends on the state at the previous time. For a historical model, this dependence extends one further time step back, meaning that we have to be able to assess demographic patterns across groups of three consecutive monitoring times.

In order to analyze these patterns of dependence, we break the data up into pairs of consecutive observation times (or groups of three consecutive times if analyzing historical models). These broken up pairs are then stacked together into a data frame. For example, if we have data for an individual in years 2010, 2011, 2012, and 2013, then we would break up these data into a data set for the years 2010 and 2011, followed by the years 2011 and 2012, followed by the years 2012 and 2013 (if historical, then we would break up 2010-2011-2012, followed by 2011-2012-2013). In terms of the language of matrix projection models, these standardized datasets are arranged with time $t$ followed by time $t+1$ if ahistorical, and time $t-1$, followed by time $t$, followed by time $t+1$ if in historical format. Thinking of this in terms of the language of linear modeling, individual status in the final time in each group can be viewed as the response variable, while status in the previous one or two times can be seen as the causal variable(s). We refer to this stacked format as vertical format, because it stretches the data for a single individual vertically across rows. This format can be further divided into historicallyformatted vertical (hfv) format (Shefferson et al., 2021), in which groups of three consecutive monitoring times are stacked, and ahistorically-formatted vertical (ahfv) format, in which pairs of consecutive years are stacked. Note that analysis in lefko3 requires historically-formatted vertical (hfv) format.

The different aspects of the states of individuals can be described by different variables per obser-
vation time. There may be one or more measures of size, an age metric, a measure of reproductive status, and perhaps fecundity, among other variables. Each monitoring occasion should have the same variables for use in analysis. Thus, concatenating states for consecutive pairs or triplets of times in this way will generally result in data frames with many more than two or three columns. Stacking the data in this fashion also ends up typically leading to multiple rows of data originating from the same individual. Until recently, these rows were treated as independent data points, and the simplest analyses still assume independence. However, if a separate variable is created to record the identity of the individual across rows within the dataset, then the non-independence of data points can be dealt with statistically by treating individual identity as a random factor in mixed modeling of vital rates (more on this in later chapters).

### 3.1 Kinds of individual state variables that may be included

What sorts of variables can be handled in MPM analysis through lefko3? Because this package was originally developed to handle historical size-classified Lefkovitch models, it can of course handle size variables. From version 4.0 .0 onward, up to three different size variables can be included in a single analysis. However, lefko3 can handle many, many more variables. At the very least, variables denoting reproductive status and fecundity can be included. These three groups of variables - size, reproductive status, and fecundity - are the classic three variable types used for classification purposes in stage-based models. Wildlife demographers may be more interested in age than size, and so age at each time can also be included. In fact, lefko3 automates the estimation of age and related variables, such as observed age at time $t$ and observed lifespan, and can be used to estimate both Leslie (age-based) and age-by-stage MPMs. Plant demographers may be particularly interested in variables denoting observation status, which is a binomial variable used to denote whether an individual is in an observable stage (lefko3 can automate the identification of these occurrences within the dataset). Some ecologists may also be interest in using multiple measures of reproductive status and fecundity, since many plants and animals have multiple modes of reproduction, and often actual fecundity cannot be accurately known and so must be estimated.

Identity variables may be included and used in analysis. Such variables include population identity, patch identity or subpopulation identity, individual identity, and monitoring occasion or time (denoting the time at time $t$ ). Package lefko3 can use these to subset the data accordingly, and also to estimate variables such as observed age at time $t$ and observed lifespan. If the user has already determined stages for all individuals at all times, then these data may be incorporated as well (although lefko3 can automatically assign stages).

Demographers commonly include censoring variables to denote data points that are suspect. These can also be included in lefko3, and those data points can also be easily removed from analysis. For example, individuals with damaged tags from which identity cannot be properly read can be marked differently in a censor variable, and analyses can then be performed with and without these potentially biasing data. Alternatively, ornithologists may mark individuals with partially missing bands with censor variables. Censor data can be added once per individual, or for each individual in each monitoring occasion.

Spatial coordinates can be included in datasets. When used, spatial coordinates should be Cartesian, and so paired as X-Y coordinates. Package lefko3 includes automated density estimation, and so these coordinates can be used to estimate spatial density.

Finally, variables can be included covering status as alive or dead, status as observed or unobserved, and up to three separate individual or environmental covariates that can be numeric or categorical. Individual covariates may be particularly useful as a means to include environmental variables, or further information on status, such as management regime or presence of injury.

### 3.2 Formatting demographic data properly for analysis

Regardless of how the dataset should be structured for analysis, demographers themselves often have their own ways of structuring data. In general, most demographers keep their data either in some variant of vertical format (explained above), or in some variant of horizontal format, in which the data for each single individual spans only a single row that contains all data on state across all monitoring occasions. In figure 3.1, we see an example of an ahistorically-formatted vertical dataset. In figure ??, we see an example of a horizontally-formatted dataset.

| $\square$ | A | B | c | D | E | F | G | H | 1 | J | K | L | M | N | ¢ |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| 1 | plantid | patch | censor | year2 | Inf2.2 | Inf. 2 | Veg. 2 | Pod. 2 | Inf2.3 | Inf. 3 | Veg. 3 | Pod. 3 |  |  |  |
| 2 | 164 | A | 1 | 2004 | 0 | 0 | 1 | 0 | NA | NA | NA | NA |  |  |  |
| 3 | 165 | A | 1 | 2004 | 0 | 2 | 1 | 1 | 0 | 2 | 0 | 0 |  |  |  |
| 4 | 240 | A | 1 | 2004 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 |  |  |  |
| 5 | 242 | A | 1 | 2004 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 |  |  |  |
| 6 | 243 | A | 1 | 2004 | 0 | 0 | 5 | 0 | 0 | - 0 | 2 | 0 |  |  |  |
| 7 | 246 | A | 1 | 2004 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 |  |  |  |
| 8 | 248 | A | 1 | 2004 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 |  |  |  |
| 9 | 249 | A | 1 | 2004 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 |  |  |  |
| 10 | 251 | A | 1 | 2004 | 0 | 0 | 2 | 0 | 0 | 0 | 2 | 0 |  |  |  |
| 11 | 252 | A | 1 | 2004 | 0 | 0 | 1 | 0 | 0 | 2 | 0 | 0 |  |  |  |
| 12 | 253 | A | 1 | 2004 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 0 |  |  |  |
| 13 | 255 | A | 1 | 2004 | 0 | 0 | 8 | 0 | 0 | 1 | 3 | 1 |  |  |  |
| 14 | 257 | A | 1 | 2004 | 0 | 0 | 6 | 0 | 0 | 3 | 3 | 1 |  |  |  |
| 15 | 259 | A | 1 | 2004 | 0 | 0 | 2 | 0 | 0 | 1 | 2 | 0 |  |  |  |
| 16 | 263 | A | 1 | 2004 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 |  |  |  |
| 17 | 254 | A | 1 | 2004 | 0 | 0 | 2 | 0 | 0 | 1 | 0 | 0 |  |  |  |
| 18 | 265 | A | 1 | 2004 | 0 | 0 | 1 | 0 | 0 | 2 | 1 | 2 |  |  |  |
| 19 | 391 | A | 1 | 2004 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 1 |  |  |  |
| 20 | 393 | A | 1 | 2004 | 0 | 2 | 3 | 2 | 0 | 3 | 1 | 1 |  |  |  |
| 21 | 394 | A | 1 | 2004 | 0 | 0 | 3 | 0 | 0 | 0 | 6 | 0 |  |  |  |
| 22 | 431 | B | 1 | 2004 | 0 | 0 | 6 | 0 | 0 | 0 | 4 | 0 |  |  |  |
| 23 | 434 | B | 1 | 2004 | 0 | 0 | 1 | 0 | 0 | 0 | 8 | 0 |  |  |  |
| 24 | 436 | B | 1 | 2004 | 0 | 2 | 8 | 0 | 1 | 8 | 2 | 0 |  |  |  |
| 25 | 437 | B | 1 | 2004 | 0 | 1 | 3 | 1 | 0 | 3 | 1 | 2 |  |  |  |
| 26 | 439 | B | 1 | 2004 | NA | NA | NA | NA | 0 | 0 | 1 | 0 |  |  |  |
| 27 | 440 | B | 1 | 2004 | 0 | 1 | 9 | 0 | 0 | 4 | 4 | 3 |  |  |  |
| 28 | 441 | B | 1 | 2004 | 0 | 0 | 4 | 0 | 0 | 2 | 4 | 2 |  |  |  |
| 29 | 442 | B | 1 | 2004 | 0 | 0 | 4 | 0 | 0 | 0 | 2 | 0 |  |  |  |

Figure 3.1: Vertically-formatted dataset covering the Cypripedium candidum data


Every demographer has their own way of keeping data, and I personally do not feel it is up to me to judge (the level of chaos in my own record-keeping prevents me from making any harsh judgments). I have therefore created two powerful functions that can take most datasets falling into these two general types and standardize them into formats that can be analyzed by lefko3. These functions are verticalize3() and historicalize3(). We will start by describing the former.

### 3.3 Function verticalize3()

Many demographers (including the author) prefer horizontal data formats for demographic recordkeeping. Here, individual life histories are recorded in a spreadsheet with each unique individual's data organized within a single row. The columns then correspond to descriptor variables and to condition at different times. This format has the advantage that of making an individual's resighting history easy to analyze even by eye, since it must occur all within a single row. In these circumstances, the verticalize3() function can standardize these data into $h f v$ format.

The verticalize3() function takes a number of inputs. Naturally, it needs the data frame being used for analysis (the data field). It also requires input corresponding to the number of monitoring occasions that the dataset covers (field noyears), and which columns identify the population, patch or subpopulation, and individual (fields popidcol, patchidcol, and individcol, respectively).

After these most basic of fields have been entered, the columns conveying each kind of demographic variable need to be identified (this is a very flexible function, so only those variables that actually occur in the dataset need be identified and all others can be ignored). Up to three separate size variables can be identified (sizeacol, sizebcol, and sizeccol), as can up to two reproductive status variables (repstracol and restrbcol), and up to two fecundity variables (fecacol and fecbcol). These variables will naturally repeat across monitoring occasions, and so each variable represents as many actual columns as monitoring occasions covered by the dataset.

To ensure that R finds all columns corresponding to a variable, the user may take one of two approaches. The most flexible is to treat each input field as a vector, and to input the name or column number of each variable in order of monitoring occasion. For example, if we decided to use only one size
variable, perhaps the height of a plant in centimeters, and this variable is labelled ht.2010, ht.2011, ht. 2012, and ht. 2013 for the years 2010, 2011, 2012, and 2013, the we can input:

```
sizeacol = c("ht.2010", "ht.2011", "ht.2012", "ht.2013").
```

All other variables covering status across time can be input in the same way, yielding equal length vectors in the same temporal order.

Alternatively, if the data is arranged such that each monitoring occasion has its data arranged in a strictly repeating order, and each variable is a fixed number of columns away from the equivalent variable in the preceding time and in the next time, then only the first of these columns needs to be designated, and the blocksize field can be set to the number of columns separating these entries. In the plant height example, if each size variable is 10 columns after the previous size variable and this pattern holds for all other variable types, then this would mean entering only
noyears $=4$, blocksize $=10$, sizeacol = "ht. 2010"
Note that the use of the blocksize option absolutely requires that each instance of a particular variable occurs exactly blocksize columns after the previous instance, with no exceptions allowed. Users who occasionally add empty columns here and there in their dataset will likely find mistakes in the resulting standardization if they do not make sure that the exact number of columns between instances is always the same.

While standardizing the data, function verticalize3() can also assign stage designations if given a proper stageframe (the stageassign field). For this purpose, lefko3 also needs the proper size variable designations. Currently, the options include:

- sizea, sizeb, or sizec if only one of these size metrics determines stage
- sizeab, sizeac, or sizebc if stage assignment is based on a pair of size metrics
- sizeabc if stage assignment should utilize three size metrics; or
- sizeadded, if stage assignment should proceed on the basis of the sum of all used size metrics.

Stage assignment is automated using these size metrics as well as reproductive status, observation status, and status as propagule, immature, and mature. Stage assignment will yield warnings and unassigned stages if any combination of variables is found in the demographic dataset that does not fit exactly one stage in the stage frame. We also encourage the use of this function without stage assignment on the first pass, since this will allow users to explore the data more fully prior to the development of a life history model and stageframe.

Function verticalize3() includes a few more fields of interest. The repstrrel and fecrel fields provide means of equalizing reproductive status variables and fecundity variables, respectively, that are set on different scales. For example, if two reproductive status variables are used - the first covering single-flowered individuals and the second covering double-flowered individual - then setting repstrrel $=2$ will tell R that the second variable counts for twice the reproductive status of the first. The same applies to fecundity, if two different fecundity measures are used.

Demographers using censoring variables will find powerful options in this function. First, censorcol can be set to a static censor variable covering an individual's entire resighting history, or to censor variables across time. This can be set with the censorRepeat field, which defaults to FALSE and so assumes that censor values do not vary across time. The value of the censor variable used to denote which individuals to keep can be set to any numeric value (the censorkeep option). The default is censorkeep $=0$, meaning that any data points marked 0 are considered OK, while any other value is assumed to be subject to removal. Any character or numeric value may be used to mark data to keep, even NA. Finally, if all options are set as described, then function verticalize3() will format the data with censor variables but will not remove any data points. To force verticalize3() to remove censored data points, set censor $=$ TRUE.

Function verticalize3() can handle spatial coordinates, and can even estimate spatial density. Cartesian coordinates for each individual are assumed to be a single set. However, if coordinates can change across time and each time point has its own set of coordinates for each individual present, then users can set coordsRepeat = TRUE. If the intent in using these coordinates is to estimate spatial density, then this function will calculate the number of individuals within some radius of each focal individual in each monitoring time if given the length of the radius to use via option spacing (e.g. spacing $=1$ sets the distance radius to check for other individuals nearby at one unit of distance).

This function includes further parameters handling miscellaneous cases. First, demographers often treat missing data as blanks within their spreadsheets. Setting NAas0 $=$ TRUE tells $R$ to interpret missing data as zeros. This can be useful in many circumstances. For example, if fecundity is estimated as the number of flowers per individual in a plant demographic study, but the number of flowers per individual is only recorded if flowers actually exist, then setting NAas $0=$ TRUE can allow 0 values to be incorporated in analyses of this fecundity variable when no data were entered for flower number. Second, the NRasRep option allows stage classification to proceed without matching stage reproductive characteristics to the data. For example, if stages are defined such that mature stages are reproductive but it is nonetheless possible for fecundity to equal 0 , then setting NRasRep $=$ TRUE tells R not to use reproductive status in stage classification. Otherwise, R will assume that a mature individual that produces offspring is in a different stage than a mature individual that does not, leading to warnings, unassigned stages, and further difficulties if the stageframe does not include stages to cover the nonreproductive category. Third, the NAasObs option provides a means of telling R that unobservable stages should be treated as observed during stage classification, if set to TRUE (the default is FALSE). Fourth, the reduce option, if set to TRUE, will tell $R$ that any columns in the output dataset with a single constant value across all rows should be removed, as invariant variables yield no influence on response variables. Fifth, the a2check field should generally be kept to FALSE (the default), but if set to TRUE, then the resulting standardized dataset will include data for individuals not alive in time $t$ (these instances are removed automatically under the default, since it is only individuals alive in time $t$ that can transition to anything in time $t+1$ ). This function also assumes that monitoring studies are conducted using pre-breeding life history models, making the earliest observed age be 1 time unit. To set to post-breeding, set prebreeding = FALSE. Finally, the age_offset option can be used to add a specific number to all ages, as might happen if the first few years of life cannot be monitored.

### 3.3.1 An example of verticalize3()

Now that we have explained the options, let's standardize data frames for the stageframes created in the last chapter (chapter 2). Let's first standardize the vertical dataset for the raw MPM. For this purpose, we will utilize the verticalize3() function. Because we are lumping reproductive and non-reproductive individuals into the non-dormant adult classes, we need to set NRasRep $=$ TRUE. Otherwise, verticalize3() will attempt to use the reproductive status of individuals in classification, and will fail due to the presence of non-reproductive adults. We also need to set NAas0 = TRUE to make sure that NA values in size are turned into 0 entries where necessary, and so aid in the assignment of the vegetative dormancy stage. Note that we will set up three different size variables here, as sizea, sizeb, and sizec, and that we will tell $R$ that we want overall stage classification size to be the sum of these (stagesize = "sizeadded"). Finally, since the individuals in this dataset must be at least five years old when they are first seen as adults, we will use the age_offset option to adjust the estimated age.

```
cypraw_v1 <- verticalize3(data = cypdata, noyears = 6, firstyear = 2004,
    patchidcol = "patch", individcol = "plantid", blocksize = 4,
    sizeacol = "Inf2.04", sizebcol = "Inf.04", sizeccol = "Veg.04",
    repstracol = "Inf.04", repstrbcol = "Inf2.04", fecacol = "Pod.04",
    stageassign = cypframe_raw, stagesize = "sizeadded", NAas0 = TRUE,
    NRasRep = TRUE, age_offset = 4)
```

Let's now take a look at a summary of this dataset. We have two options for this. First, we could use R's summary () function, which would treat our new historically vertically (hfv) formatted dataset as a standard data frame and use the standard summary procedure for data frames. Or, we may use lefko3's summary_hfv() function, which gives us a little extra information.

```
summary_hfv(cypraw_v1)
>
> This hfv dataset contains }320\mathrm{ rows, 57 variables, 1 population,
> 3 patches, }74\mathrm{ individuals, and 5 time steps.
> rowid popid patchid individ year2
> Min. : 1.00 Length:320 A: 93 Min. : 164.0 Min. :2004
> 1st Qu.:21.00 Class :character B:154 1st Qu.: 391.0 1st Qu.:2005
> Median :37.50 Mode :character C: 73 Median : 453.0 Median :2006
> Mean :38.45
> 3rd Qu.:56.00
> Max. :77.00
> firstseen
Min. :2004 Min. :2004 Min. :5.000 Min. :0.000
> 1st Qu.:2004 1st Qu.:2009 1st Qu.:6.000 1st Qu.:5.000
> Median :2004 Median :2009 Median :7.000 Median :5.000
> Mean :2004 Mean :2009 Mean :6.853 Mean :4.556
> 3rd Qu.:2004 3rd Qu.:2009 3rd Qu.:8.000 3rd Qu.:5.000
> Max. :2008 Max. :2009 Max. :9.000 Max. :5.000
> sizea1 sizeb1 sizec1 size1added
> Min. :0.000000 Min. : 0.0000 Min. : 0.0 Min. : 0.000
> 1st Qu.:0.000000 1st Qu.: 0.0000 1st Qu.: 0.0 1st Qu.: 0.000
> Median :0.000000 Median : 0.0000 Median : 1.0 Median : 2.000
> Mean :0.009375 Mean : 0.7469 Mean : 1.9 Mean : 2.656
> 3rd Qu.:0.000000 3rd Qu.: 1.0000 3rd Qu.: 3.0 3rd Qu.: 4.000
> Max. :1.000000 Max. :18.0000 Max. :13.0 Max. :21.000
> repstra1 repstrb1 repstr1added feca1
>Min. : 0.0000 Min. :0.000000 Min. : 0.0000 Min. :0.0000
> 1st Qu.: 0.0000 1st Qu.:0.000000 1st Qu.: 0.0000 1st Qu.:0.0000
> Median : 0.0000 Median :0.000000 Median : 0.0000 Median :0.0000
> Mean :0.7469 Mean :0.009375 Mean : 0.7562 Mean :0.2656
> 3rd Qu.: 1.0000 3rd Qu.:0.000000 3rd Qu.: 1.0000 3rd Qu.:0.0000
> Max. :18.0000 Max. :1.000000 Max. :18.0000 Max. :7.0000
> fec1added obsstatus1 repstatus1 fecstatus1
> Min. :0.0000 Min. :0.0000 Min. :0.0000 Min. :0.0000
> 1st Qu.:0.0000 1st Qu.:0.0000 1st Qu.:0.0000 1st Qu.:0.0000
> Median :0.0000 Median :1.0000 Median :0.0000 Median :0.0000
> Mean :0.2656 Mean :0.7469 Mean :0.2875 Mean :0.1344
> 3rd Qu.:0.0000 3rd Qu.:1.0000 3rd Qu.:1.0000 3rd Qu.:0.0000
> Max. :7.0000 Max. :1.0000 Max. :1.0000 Max. :1.0000
> matstatus1 alive1 stage1 stage1index
> Min. :0.0000 Min. :0.0000
> 1st Qu.:1.0000
> Median :1.0000 Median :1.0000
> Mean :0.7688 Mean :0.7688
> 3rd Qu.:1.0000 3rd Qu.:1.0000
Max. :1.0000 Max. :1.0000
    sizeb2 sizec2 size2added
```

```
> 1st Qu.:0.000000
    1st Qu.: 0.0000
    Median : 0.0000 Median : 2.000
    Mean : 0.8969 Mean : 2.416 Mean : 3.322
    3rd Qu.: 1.0000 3rd Qu.: 3.000 3rd Qu.: 4.000
    Max. :18.0000 Max. :13.000 Max. :24.000
    repstrb2 repstr2added feca2
Min. : 0.0000 Min. :0.000000 Min. : 0.0000 Min. :0.0000
1st Qu.: 0.0000 1st Qu.:0.000000 1st Qu.: 0.0000 1st Qu.:0.0000
Median : 0.0000 Median :0.000000 Median : 0.0000 Median :0.0000
Mean : 0.8969 Mean :0.009375 Mean : 0.9062 Mean :0.2906
3rd Qu.: 1.0000 3rd Qu.:0.000000 3rd Qu.: 1.0000 3rd Qu.:0.0000
Max. :18.0000 Max. :1.000000 Max. :18.0000 Max. :7.0000
    fec2added obsstatus2 repstatus2 fecstatus2
Min. :0.0000 Min. :0.0000 Min. :0.0000 Min. :0.0000
1st Qu.:0.0000 1st Qu.:1.0000 1st Qu.:0.0000 1st Qu.:0.0000
Median :0.0000 Median :1.0000 Median :0.0000 Median :0.0000
Mean :0.2906 Mean :0.9531 Mean :0.3688 Mean :0.1562
3rd Qu.:0.0000 3rd Qu.:1.0000 3rd Qu.:1.0000 3rd Qu.:0.0000
Max. :7.0000 Max. :1.0000 Max. :1.0000 Max. :1.0000
    matstatus2 alive2 stage2 stage2index
Min. :1 Min. :1 Length:320 Min. : 6.000
1st Qu.:1 1st Qu.:1 Class :character 1st Qu.: 7.000
Median :1 Median :1 Mode :character Median : 8.000
Mean :1 Mean :1 Mean : 7.919
3rd Qu.:1 3rd Qu.:1 3rd Qu.: 8.000
Max. :1 Max. :1 Max. :11.000
    sizea3 sizeb3 sizec3 size3added
Min. :0.000000 Min. : 0.000 Min. : 0.000 Min. : 0.000
1st Qu.:0.000000 1st Qu.: 0.000 1st Qu.: 1.000 1st Qu.: 1.000
Median :0.000000 Median : 0.000 Median : 1.000 Median : 2.000
Mean :0.009375 Mean : 1.069 Mean : 2.209 Mean : 3.288
3rd Qu.:0.000000 3rd Qu.: 1.000 3rd Qu.: 3.000 3rd Qu.: 4.000
Max. :1.000000 Max. :18.000 Max. :13.000 Max. :24.000
        repstra3 repstrb3 repstr3added feca3
Min. : 0.000
1st Qu.: 0.000
Median : 0.000
Mean : 1.069
3rd Qu.: 1.000
Max. :18.000
        fec3added
Min. :0.0000
1st Qu.:0.0000
Median :0.0000
Mean :0.4562
3rd Qu.:0.0000
Max. :8.0000
    alive3
        Min. :0.000000
        1st Qu.:0.000000
        Median :0.000000
        Mean :0.009375
        3rd Qu.:0.000000 3rd Qu.: 1.000 3rd Qu.:0.0000
        Max. :1.000000 Max. :18.000 Max. :8.0000
        obsstatus3 repstatus3 fecstatus3 matstatus3
        Min. :0.0 Min. :0.0 Min. :0.0000 Min. :1
    1st Qu.:1.0 1st Qu.:0.0 1st Qu.:0.0000 1st Qu.:1
    Median :1.0 Median :0.0 Median :0.0000 Median :1
    Mean :0.9 Mean :0.4 Mean :0.2219 Mean :1
    3rd Qu.:1.0 3rd Qu.:1.0 3rd Qu.:0.0000 3rd Qu.:1
    Max. :1.0 Max. :1.0 Max. :1.0000 Max. :1
        stage3 stage3index
    Min. :0.0000 Length:320 Min. : 0.000
    1st Qu.:1.0000 Class :character 1st Qu.: 7.000
```

| $>$ | Median :1.0000 | Mode : character | Median : 8.000 |
| :---: | :---: | :---: | :---: |
| > | Mean :0.9469 |  | Mean : 7.544 |
| $>$ | 3rd Qu.:1.0000 |  | 3rd Qu.: 8.000 |
| > | Max. :1.0000 |  | Max. :11.000 |

In the summary of the resulting data frame, we first see that there are 320 rows, 57 variables, one population, three patches, 74 individuals, and five time steps. This information gives us a good understanding of the dimensions of the dataset. This is followed by R's data frame summary, where we see that the first four variables are identifying information - they show the data row (rowid), the population ID (popid), the patch ID (patchid), and the individual ID (individ), in order. These are followed by two variables identifying in what monitoring occasion the individual was first seen (firstseen), and in what occasion it was last seen (lastseen). The next variable, obsage, gives the estimated age at time $t$. The next variable, obslifespan, gives the full length of time that the individual was observed in the dataset. After that, we see a group of 15 variables typically ending with the number 1. These 15 variables correspond to the state of each individual in time $t-1$ (generally referred to as time1 in the actual code). Similarly, the next 15 variables typically end with the number 2 and represent state in time $t$, while the last 15 variables typically end in the number 3 and refer to state in time $t+1$. Most of these variables should be obvious to interpret given the names (definitions are in the help file for this function). Note that the final seven variables in each group are calculated by verticalize3(). These variables represent observation status, reproductive status, fecundity status, maturity status, status as alive or dead, stage name, and stage number (with reference to the stageframe), in order.

The summary-hfv() function can also produce just the initial dimensional output, as below.

```
summary_hfv(cypraw_v1, full = FALSE)
>
> This hfv dataset contains 320 rows, 57 variables, 1 population,
> 3 patches, 74 individuals, and 5 time steps.
```

Let's also create our standardized data frame for the function-based MPM, as below. Remember that some of the settings need to change here because we will use a different life history model. Particularly, we are now going to separate adults not just by size but by reproductive status. So, we will NOT set NRasRep $=$ TRUE here (the default is NRasRep $=$ FALSE).

```
cypfb_v1 <- verticalize3(data = cypdata, noyears = 6, firstyear = 2004,
    patchidcol = "patch", individcol = "plantid", blocksize = 4,
    sizeacol = "Inf2.04", sizebcol = "Inf.04", sizeccol = "Veg.04",
    repstracol = "Inf.04", repstrbcol = "Inf2.04", fecacol = "Pod.04",
    stageassign = cypframe_fb, stagesize = "sizeadded", NAas0 = TRUE,
    age_offset = 4)
summary_hfv(cypfb_v1)
>
> This hfv dataset contains 320 rows, 57 variables, 1 population,
> 3 patches, }74\mathrm{ individuals, and 5 time steps.
> rowid popid patchid individ year2
> Min. : 1.00 Length:320 A: 93 Min. : 164.0 Min. :2004
> 1st Qu.:21.00 Class :character B:154 1st Qu.: 391.0 1st Qu.:2005
> Median :37.50 Mode :character C: 73 Median : 453.0 Median :2006
> Mean :38.45 Mean : 651.5 Mean :2006
> 3rd Qu.:56.00 3rd Qu.: 476.0 3rd Qu.:2007
> Max. :77.00 Max. :1560.0 Max. :2008
```



```
1st Qu.:0.0000
Median :0.0000
Mean :0.2906
3rd Qu.:0.0000
Max. :7.0000
            matstatus2
Min. :1 Min. :1 Length:320 Min. : 6.00 Min. :0.000000
1st Qu.:1 1st Qu.:1 Class :character 1st Qu.: 7.00 1st Qu.:0.000000
Median :1 Median :1 Mode :character Median :10.00 Median :0.000000
Mean :1 Mean :1 Mean :18.17 Mean :0.009375
3rd Qu.:1 3rd Qu.:1 3rd Qu.:32.00 3rd Qu.:0.000000
Max. :1 Max. :1 Max. :54.00 Max. :1.000000
            sizeb3 sizec3
Min. : 0.000
1st Qu.: 0.000 1st Qu.: 1.000
Median : 0.000 Median : 1.000
Mean : 1.069 Mean : 2.209 Mean : 3.288 Mean : 1.069
3rd Qu.: 1.000 3rd Qu.: 3.000 3rd Qu.: 4.000 3rd Qu.: 1.000
Max. :18.000 Max. :13.000 Max. :24.000 Max. :18.000
    repstrb3 repstr3added feca3 fec3added
Min. :0.000000 Min. : 0.000 Min. :0.0000 Min. :0.0000
1st Qu.:0.000000 1st Qu.: 0.000 1st Qu.:0.0000 1st Qu.:0.0000
Median :0.000000 Median : 0.000 Median :0.0000 Median :0.0000
Mean :0.009375 Mean : 1.078 Mean :0.4562 Mean :0.4562
3rd Qu.:0.000000 3rd Qu.: 1.000 3rd Qu.:0.0000 3rd Qu.:0.0000
Max. :1.000000 Max. :18.000 Max. :8.0000 Max. :8.0000
    obsstatus3 repstatus3 fecstatus3 matstatus3 alive3
Min. :0.0 Min. :0.0 Min. :0.0000 Min. :1 Min. :0.0000
1st Qu.:1.0 1st Qu.:0.0 1st Qu.:0.0000 1st Qu.:1 1st Qu.:1.0000
Median :1.0 Median :0.0 Median :0.0000 Median :1 Median :1.0000
Mean :0.9 Mean :0.4 Mean :0.2219 Mean :1 Mean :0.9469
3rd Qu.:1.0 3rdQu.:1.0 3rdQu.:0.0000 3rdQu.:1 3rd Qu.:1.0000
Max. :1.0 Max. :1.0 Max. :1.0000 Max. :1 Max. :1.0000
    stage3 stage3index
Length:320 Min. : 0.00
Class :character 1st Qu.: 7.00
Mode :character Median :10.00
    Mean :18.57
    3rd Qu.:33.00
    Max. :54.00
```

The output dataset includes a number of summary variables, but the data is essentially broken down into groups of three consecutive monitoring occasions each (occasions $t+1, t$, and $t-1$, corresponding to year3, year2, and year1 in the output, respectively), with individuals spread across multiple rows. The output dataset is further limited to those entries in which the individual is alive in occasion $t$ (year2), meaning that all rows in which an individual is dead or not yet recruited in occasion $t$ are dropped. Since the input data is the same, we should see the same numbers of rows and columns in the raw and function-based cases, regardless of the different stageframes used. Thus, we have 320 rows of data and 57 variables in the raw case, and 320 rows of data and 57 variables in the function-based case.

### 3.4 Function historicalize3()

What should be done if the dataset is in ahistorical vertical format, in which an individual's condition across time is recorded across rows in a spreadsheet, with a single row corresponding to either a single observation or to a pair of consecutive observations? In that case, the historicalize3() function can standardize the dataset properly. The inputs to this function are very similar to verticalize3(). However, the historicalize3() function assumes that the input dataset is organized with rows corresponding to individual status in either only one monitoring occasion, or two consecutive occasions. Inputs include a series of variables more or less equivalent to input options in verticalize3(), but some variable names end with 2 col , and others end with 3 col . The former variables denote status in time $t$, and are the minimum required for this function to run. If each row includes status in paired consecutive times, then variables ending in 3 col can be used to designate status in time $t+1$. Additionally, this function requires a single variable identifying individuals across rows (individcol), so that each individual's resighting history can be inferred and each group of three consecutive monitoring times can be put together. All other fields work essentially the same as in verticalize3().

### 3.4.1 An example of historicalize3()

Package lefko3 also includes dataset cypvert, which is the same dataset as cypdata but set in ahistorical vertical format. Here, we will use the historicalize3() function to standardize this dataset, using the plantid variable as the individual identity term.

```
data(cypvert)
cypraw_v2 <- historicalize3(data = cypvert, patchidcol = "patch",
    individcol = "plantid", year2col = "year2", sizea2col = "Inf2.2",
    sizea3col = "Inf2.3", sizeb2col = "Inf.2", sizeb3col = "Inf.3",
    sizec2col = "Veg.2", sizec3col = "Veg.3", repstra2col = "Inf2.2",
    repstra3col = "Inf2.3", repstrb2col = "Inf.2", repstrb3col = "Inf.3",
    feca2col = "Pod.2", feca3col = "Pod.3", repstrrel = 2,
    stageassign = cypframe_raw, stagesize = "sizeadded", censorcol = "censor",
    censorkeep = 1, censor = FALSE, NAas0 = TRUE, NRasRep = TRUE, age_offset = 4,
    reduce = TRUE)
summary_hfv(cypraw_v2)
>
> This hfv dataset contains 320 rows, 57 variables, 1 population,
> 3 patches, }74\mathrm{ individuals, and 5 time steps.
popid patchid
> Min. : 0.00 Length:320 A: 93 Min. : 164.0 Min. :2004
> 1st Qu.: 79.75 Class :character B:154 1st Qu.: 391.0 1st Qu.:2005
> Median :159.50 Mode :character C: 73 Median : 453.0 Median :2006
.159.70
\ Mean Qu.159.70
> 3rd Qu.:239.25
> Max. :321.00
> firstseen
> Min :2004 Min -2004
> 1st Qu.:2004 1st Qu.:2009 1st Qu.:6.000 1st Qu.:5.00
> Median :2004 Median :2009 Median :7.000 Median :5.000
> Mean :2004 Mean :2009 Mean :6.853 Mean :4.556
> 3rd Qu.:2004 3rd Qu.:2009 3rd Qu.:8.000 3rd Qu.:5.000
> Max. :2008 Max. :2009 Max. :9.000 Max. :5.000
```




Let's also create the function-based MPM version, which differs primarily in the use of the functionbased model stageframe.

```
cypfb_v2 <- historicalize3(data = cypvert, patchidcol = "patch",
    individcol = "plantid", year2col = "year2", sizea2col = "Inf2.2",
    sizea3col = "Inf2.3", sizeb2col = "Inf.2", sizeb3col = "Inf.3",
    sizec2col = "Veg.2", sizec3col = "Veg.3", repstra2col = "Inf2.2",
    repstra3col = "Inf2.3", repstrb2col = "Inf.2", repstrb3col = "Inf.3",
    feca2col = "Pod.2", feca3col = "Pod.3", repstrrel = 2,
    stageassign = cypframe_fb, stagesize = "sizeadded", censorcol = "censor",
    censorkeep = 1, censor = FALSE, NAas0 = TRUE, age_offset = 4, reduce = TRUE)
summary_hfv(cypfb_v2)
>
> This hfv dataset contains }320\mathrm{ rows, 57 variables, 1 population,
> 3 patches, }74\mathrm{ individuals, and 5 time steps.
> rowid popid patchid individ year2
> Min. : 0.00 Length:320 A: 93 Min. : 164.0 Min. :2004
```




One important consideration is the use of censor variables. Censoring a demographic dataset is typically accomplished by including a binary variable in the dataset denoting whether an individual datum is to be kept or excluded. The objects cypraw_v2 and cpyfb_v2 were both created without using censoring variables. However, because the datasets actually included censor variables (all data were set to be included, with no suspect data), we wished to incorporate those variables in the final datasets. Hence, although censor = FALSE in both the call to verticalize3() and the
call to historicalize3(), we also noted censorcol = "censor" and censorkeep $=1$ in the call to historicalize3(). Failing to add these options to the call to historicalize3() will produce approximately the same dataset, but with some zeros entering variables censor1, censor2, and censor3 in the historicalized dataset that do not exist in the first, verticalized dataset. This occurs because the vertical dataset that we standardized only includes entries for each individual actually observed at each time, while the horizontal dataset includes entries for all individuals at all times, regardless of whether they were observed. Thus, by setting these options, we have produced datasets that are exactly the same except for variable order. We can see hints of that by looking at the dimensions of our datasets, as below.

```
summary_hfv(cypraw_v1, full = FALSE)
>
> This hfv dataset contains 320 rows, 57 variables, 1 population,
> 3 patches, }74\mathrm{ individuals, and 5 time steps.
summary_hfv(cypraw_v2, full = FALSE)
>
> This hfv dataset contains 320 rows, 57 variables, 1 population,
> 3 patches, 74 individuals, and 5 time steps.
summary_hfv(cypfb_v1, full = FALSE)
>
> This hfv dataset contains 320 rows, 57 variables, 1 population,
> 3 patches, }74\mathrm{ individuals, and 5 time steps.
summary_hfv(cypfb_v2, full = FALSE)
>
> This hfv dataset contains 320 rows, 57 variables, 1 population,
> 3 patches, }74\mathrm{ individuals, and 5 time steps.
```

One final consideration in historicalize3() regards the use of the 3col sets of options. If users have vertical datasets with pairs of consecutive states, then setting the 3 col options to the latter state in each row means that the final year will be included in the historicalized dataset. Failing to enter the 3 col set in these cases would mean that the final monitoring occasion would be dropped, since it likely only appears in rows holding data for the second to last monitoring occasion.

### 3.5 Handling spatial data and density

Both functions verticalize3() and historicalize3() handle the formatting of spatial coordinates and the estimation of density. There are four important settings that need to be used in the former case, and three in the latter. In both functions, the first two settings correspond to the columns coding for the X coordinate ( xcol ), and the Y coordinate (ycol). In both cases, the name of the variable or the column number can be used. The third setting (spacing), used in both functions, sets the radius from the coordinate of the individual to search for other individuals. The approach used for density estimation is to find and count all individuals in the same occasion within this radius from each individual, with density estimation performed for each individual in each time. The final setting, used only in verticalize3(), is a logical variable telling $R$ whether the X and Y coordinate variables are single variables for the individual, or whether each individual has potentially new X and Y coordinates at each time that it is observed (coordsRepeat).

Here, we will produce a new version of the function-based cypdata $h f v$ dataset with density at time $t$ estimated using a 1 m radius. Notice that the result includes xpos and ypos variables for times $t+1$, $t$, and $t-1$, and local density (density) at time $t$.

```
cypfb_vdens <- verticalize3(data = cypdata, noyears = 6, firstyear = 2004,
    patchidcol = "patch", individcol = "plantid", blocksize = 4,
    xcol = "X", ycol = "Y", sizeacol = "Inf2.04", sizebcol = "Inf.04",
    sizeccol = "Veg.04", repstracol = "Inf.04", repstrbcol = "Inf2.04",
    fecacol = "Pod.04", stageassign = cypframe_fb, stagesize = "sizeadded",
    NAas0 = TRUE, age_offset = 4, coordsRepeat = FALSE, spacing = 1)
summary_hfv(cypfb_vdens)
>
> This hfv dataset contains 320 rows, }64\mathrm{ variables, 1 population,
> 3 patches, 74 individuals, and 5 time steps.
\(>\) rowid popid patchid individ year2
\(>\) Min. : 1.00 Length:320 A: 93 Min. : 164.0 Min. :2004
> 1st Qu.:21.00 Class :character B:154 1st Qu.: 391.0 1st Qu.:2005
> Median :37.50 Mode :character C: 73 Median : 453.0 Median :2006
Mean :38.45 Mean : 651.5 Mean :2006
3rd Qu.:56.00 3rd Qu.: 476.0 3rd Qu.:2007
    Max. :77.00 Max. :1560.0 Max. :2008
> firstseen
> Min.
> 1st Qu.:2004
> Median :2004
> Mean :2004
> 3rd Qu.:2004 3rd Qu.:2009
Max. :2008 Max. :2009 Max. :9.000 Max. :5.000 Max. :166.30
ypos1 sizea1 sizeb1 sizec1
Min. :-28.00
    1st Qu.: 0.00
    Median : 70.90
    Mean : 44.93 Mean :0.009375
    3rd Qu.: 79.85 3rd Qu.:0.000000
    Max. :142.40 Max. :1.000000
        size1added
    Min. : 0.000
    1st Qu.: 0.000
    Median : 2.000
    Mean : 2.656
    3rd Qu.: 4.000
    Max. :21.000
        feca1
    Min. :0.0000
    1st Qu.:0.0000
    Median :0.0000
    Mean :0.2656
    3rd Qu.:0.0000
    Max. :7.0000
        fecstatus1
    Min. :0.0000
    1st Qu.:0.0000
    Median :0.0000
    Mean :0.1344 Mean :0.7688 Mean :0.7688
    Min. :0.000000
        Min. : 0.0000 Min. : 0.0
            1st Qu.:0.000000 1st Qu.: 0.0000 1st Qu.: 0.0
            Median :0.000000 Median : 0.0000 Median : 1.0
            Mean :0.009375 Mean : 0.7469 Mean : 1.9
            3rd Qu.:0.000000 3rd Qu.: 1.0000 3rd Qu.: 3.0
            Max. :1.000000 Max. :18.0000 Max. :13.0
        repstra1 repstrb1
        Max. :13.0
    Min. : 0.0000 Min. :0.000000 Min. : 0.0000
    1st Qu.: 0.0000 1st Qu.:0.000000 1st Qu.: 0.0000
    Median : 0.0000 Median :0.000000 Median : 0.0000
    Mean : 0.7469 Mean :0.009375 Mean : 0.7562
    3rd Qu.: 1.0000 3rd Qu.:0.000000 3rd Qu.: 1.0000
    Max. :18.0000 Max. :1.000000 Max. :18.0000
    fec1added obsstatus1 repstatus1
    Min. :0.0000 Min. :0.0000 Min. :0.0000
    1st Qu.:0.0000 1st Qu.:0.0000 1st Qu.:0.0000
    Median :0.0000 Median :1.0000 Median :0.0000
    Mean :0.2656 Mean :0.7469 Mean :0.2875
    3rd Qu.:0.0000 3rd Qu.:1.0000 3rd Qu.:1.0000
    Max. :7.0000 Max. :1.0000 Max. :1.0000
        matstatus1 alive1
    Min. :0.0000 Min. :0.0000
    1st Qu.:1.0000 1st Qu.:1.0000
    Median :1.0000 Median :1.0000
        Class : character
        Mode :character
```

| > | 3rd Qu. 0.0000 | 3rd Qu.:1.0000 | 3rd Qu.:1.0000 |  |
| :---: | :---: | :---: | :---: | :---: |
| > | Max. $: 1.0000$ | Max. :1.0000 | Max. : 1.0000 |  |
| > | stage1index | xpos2 | ypos2 | sizea2 |
| > | Min. : 0.00 | Min. : 46.50 M | Min. :-28.00 M | Min. $\quad 0.000000$ |
| > | 1st Qu.: 6.00 | 1st Qu.: 60.10 1 | 1st Qu.: 23.30 1 | 1st Qu.:0.000000 |
| $>$ | Median : 8.00 | Median : 90.65 M | Median : 77.00 M | Median :0.000000 |
| > | Mean : 14.17 | Mean : 91.19 M | Mean : 56.98 M | Mean :0.009375 |
| > | 3rd Qu.:31.00 | 3rd Qu.:141.80 3 | 3rd Qu.: 80.40 3 | 3rd Qu.:0.000000 |
| > | Max. :51.00 | Max. :173.00 M | Max. :142.40 M | Max. :1.000000 |
| > | sizeb2 | sizec2 | size2added | repstra2 |
| > | Min. : 0.0000 | Min. : 0.000 | Min. : 0.000 | Min. : 0.0000 |
| > | 1st Qu.: 0.0000 | 1st Qu.: 1.000 | 1st Qu.: 1.000 | 1st Qu.: 0.0000 |
| > | Median : 0.0000 | Median : 2.000 | Median : 2.000 | Median : 0.0000 |
| > | Mean : 0.8969 | Mean : 2.416 | Mean : 3.322 | Mean : 0.8969 |
| > | 3rd Qu.: 1.0000 | 3rd Qu.: 3.000 | 3rd Qu.: 4.000 | 3rd Qu.: 1.0000 |
| > | Max. 18.0000 | Max. : 13.000 | Max. $: 24.000$ | Max. : 18.0000 |
| > | repstrb2 | repstr2added | feca2 | fec2added |
| > | Min. $: 0.000000$ | 0 Min. : 0.0000 | 0 Min. $: 0.0000$ | 0 Min. 00.0000 |
| > | 1st Qu.:0.000000 | 1st Qu.: 0.0000 | 0 1st Qu.:0.0000 | 0 1st Qu.:0.0000 |
| > | Median :0.000000 | 0 Median : 0.0000 | 0 Median :0.0000 | 0 Median :0.0000 |
| > | Mean :0.009375 | 5 Mean : 0.9062 | 2 Mean :0.2906 | 6 Mean :0.2906 |
| > | 3rd Qu.:0.000000 | 3rd Qu.: 1.0000 | O 3rd Qu.:0.0000 | 0 3rd Qu.:0.0000 |
| > | Max. :1.000000 obsstatus2 | Max. :18.0000 <br> repstatus2 | 0 Max. :7.0000 <br> fecstatus2 | 0 Max. :7.0000 <br> matstatus2 alive2 |
| > | Min. 0.0000 | Min. 00.0000 | Min. 00.0000 | Min. :1 Min. 11 |
| > | 1st Qu.:1.0000 | 1st Qu.:0.0000 | 1st Qu.:0.0000 | 1st Qu.:1 1st Qu.:1 |
| > | Median :1.0000 | Median :0.0000 | Median :0.0000 | Median :1 Median :1 |
| > | Mean :0.9531 | Mean :0.3688 | Mean :0.1562 | Mean :1 Mean :1 |
| > | 3rd Qu.:1.0000 | 3rd Qu.:1.0000 | 3rd Qu.:0.0000 | 3rd Qu.:1 3rd Qu.:1 |
| > | Max. $: 1.0000$ | Max. $: 1.0000$ | Max. $: 1.0000$ | Max. :1 Max. :1 |
| > | stage2 | stage2index | xpos3 | ypos3 |
| > | Length:320 | Min. : 6.00 | Min. : 46.50 | Min. : -28.00 |
| > | Class : character | r 1st Qu.: 7.00 | 1st Qu.: 60.10 | 1st Qu.: 23.30 |
| > | Mode : character | r Median :10.00 | Median : 90.65 | Median : 77.00 |
| > |  | Mean : 18.17 | Mean : 91.19 | Mean : 56.98 |
| > |  | 3rd Qu.:32.00 | 3rd Qu.:141.80 | 3rd Qu.: 80.40 |
| > |  | Max. :54.00 | Max. : 173.00 | Max. : 142.40 |
| > | sizea3 | sizeb3 | sizec3 | size3added |
| > | Min. 0.000000 | Min. : 0.000 | Min. : 0.000 | Min. : 0.000 |
| > | 1st Qu.:0.000000 | 1st Qu.: 0.000 | 1st Qu.: 1.000 | 1st Qu.: 1.000 |
| > | Median :0.000000 | Median : 0.000 | Median : 1.000 | Median : 2.000 |
| > | Mean :0.009375 | 5 Mean : 1.069 | Mean : 2.209 | Mean : 3.288 |
| > | 3rd Qu.:0.000000 | 3rd Qu.: 1.000 | 3rd Qu.: 3.000 | 3rd Qu.: 4.000 |
| > | Max. $: 1.000000$ | Max. :18.000 | Max. : 13.000 | Max. $: 24.000$ |
| $>$ | repstra3 | repstrb3 | repstr3added | feca3 |
| > | Min. : 0.000 | Min. 0.000000 | Min. : 0.000 | Min. $\quad 0.0000$ |
| > | 1st Qu.: 0.000 | 1st Qu.:0.000000 | 1st Qu.: 0.000 | 1st Qu.:0.0000 |
| > | Median : 0.000 | Median :0.000000 | Median : 0.000 | Median :0.0000 |
| > | Mean : 1.069 | Mean :0.009375 | Mean : 1.078 | Mean :0.4562 |
| > | 3rd Qu.: 1.000 | 3rd Qu.:0.000000 | 3rd Qu.: 1.000 | 3rd Qu.:0.0000 |
| > | Max. :18.000 | Max. 1.000000 | Max. 18.000 | Max. 8.80000 |

> 3rd Qu.:0.0000
:1.0000
$>$ Min. : 0.00
> 1st Qu.: 6.00
$>$ Median : 8.00
$>$ Mean : 14.17
$>$ 3rd Qu.:31.00
$>$ Max. :51.00
sizeb2
Min. : 0.0000
$>$ 1st Qu.: 0.0000
$>$ Median : 0.0000
$>$ Mean : 0.8969
$>$ 3rd Qu.: 1.0000
$>$ Max. : 18.0000
repstrb2
$>$ Min. 0.000000
> 1st Qu.:0.000000
$>$ Median :0.000000
$>$ Mean :0.009375
$>$ 3rd Qu.:0.000000
Max. :1.000000
obsstatus2
Min. :0.0000
1st Qu.:1.0000
Median :1.0000
Mean : 0.9531
3rd Qu.: 1.0000
Max. :1.0000
stage2
Length: 320
Mode :character
Median :10.00
Mean : 18.17
3rd Qu.:32.00
Max. :54.00 sizeb3
Min. : 0.000
1st Qu.: 0.000
Median : 0.000
Mean : 1.069
3rd Qu.: 1.000
repstrb3
Min. :0.000000
1st Qu.:0.000000
Median :0.000000
Mean :0.009375
3rd Qu.:0.000000
Max. :1.000000

3rd Qu.:1.0000
Max. :1.0000

matstatus2 alive2
Min. :1 Min. :1
1st Qu.:1 1st Qu.:1
Median :1 Median :1
Mean :1 Mean :1
3rd Qu.:1 3rd Qu.:1
Max. :1 Max. :1
ypos3
Min. :-28.00
1st Qu.: 23.30
Median : 77.00
ean : 56.98
3rd Qu.: 80.40
Max. : 142.40
size3added
Min. : 0.000
1st Qu.: 1.000
Median : 2.000
Mean : 3.288
3rd Qu.: 4.000
Max. :24.000
feca3
Min. :0.0000
1st Qu.:0.0000
Median :0.0000
Mean :0.4562
3rd Qu.:0.0000
Max. :8.0000


Next, let's try creating the same formatted dataset, but using the historicalize3() function applied on cypvert. The options to add here are the same as in verticalize3(), except for coordsRepeat (the latter is only available for horizontally formatted dataset inputs). Note that the summaries look essentially the same for both standardized versions of the dataset.

```
cypfb_hdens <- historicalize3(data = cypvert, patchidcol = "patch",
    individcol = "plantid", xcol = "X", ycol = "Y", year2col = "year2",
    sizea2col = "Inf2.2", sizea3col = "Inf2.3", sizeb2col = "Inf.2",
    sizeb3col = "Inf.3", sizec2col = "Veg.2", sizec3col = "Veg.3",
    repstra2col = "Inf2.2", repstra3col = "Inf2.3", repstrb2col = "Inf.2",
    repstrb3col = "Inf.3", feca2col = "Pod.2", feca3col = "Pod.3", repstrrel = 2,
    stageassign = cypframe_fb, stagesize = "sizeadded", censorcol = "censor",
    censorkeep = 1, censor = FALSE, NAas0 = TRUE, age_offset = 4, spacing = 1,
    reduce = TRUE)
summary_hfv(cypfb_hdens)
>
> This hfv dataset contains }320\mathrm{ rows, }64\mathrm{ variables, 1 population,
> 3 patches, 74 individuals, and 5 time steps.
\(>\) rowid popid patchid individ year2
> Min. : 0.00 Length:320 A: 93 Min. : 164.0 Min. :200
> 1st Qu.: 79.75
> Median :159.50
> Mean :159.70
> 3rd Qu.:239.25
> Max. :321.00
            firstseen lastseen
    Min. :2004 Min. :2004 Min. :5.000 Min. :0.000 Min. : 46.50
                        1st Qu.:2009
                        Median :2009
                        Mean :2009 Mean :6.853
> 1st Qu.:2004
> Median :2004
> Mean :2004 Mean :2009
> 3rd Qu.:2004 3rd Qu.:2009 3rd Qu.:8.000 3rd Qu.:5.000 3rd Qu.:141.80
> Max. :2008 Max. :2009 Max. :9.000 Max. :5.000 Max. :173.00
> ypos1 sizea1 sizeb1 sizec1
> Min. :-28.00 Min. :0.000000 Min. : 0.0000 Min. : 0.0
> 1st Qu.: 23.30 1st Qu.:0.000000 1st Qu.: 0.0000 1st Qu.: 0.0
> Median : 77.00 Median :0.000000 Median : 0.0000 Median : 1.0
```

| > | Mean : 56.98 | Mean :0.009375 | $\begin{aligned} & \text { Mean : } 0.7469 \\ & \text { 3rd Qu.: } 1.0000 \end{aligned}$ | 9 Mean : 1.9 |
| :---: | :---: | :---: | :---: | :---: |
| > | 3rd Qu.: 80.40 | 3rd Qu.:0.000000 |  | O 3rd Qu.: 3.0 |
| > | $\text { Max. }: 142.40$ <br> size1added | Max. : 1.000000 <br> repstra1 | $\begin{gathered} \text { Max. } \quad: 18.0000 \\ \text { repstrb1 } \end{gathered}$ | $\begin{aligned} & \text { Max. }: 13.0 \\ & \text { repstr1added } \end{aligned}$ |
| > | Min. : 0.000 | Min. $: 0.000000$ | Min. : 0.0000 | 0 Min. : 0.000 |
| > | 1st Qu.: 0.000 | 1st Qu.:0.000000 | 1st Qu.: 0.0000 | 0 1st Qu.: 0.000 |
| > | Median : 2.000 | Median :0.000000 | Median : 0.0000 | 0 Median : 0.000 |
| > | Mean : 2.656 | Mean :0.009375 | Mean : 0.7469 | 9 Mean : 1.503 |
| > | 3rd Qu.: 4.000 | 3rd Qu.:0.000000 | 3rd Qu.: 1.0000 | 0 3rd Qu.: 2.000 |
| > | Max. :21.000 | Max. : 1.000000 | Max. : 18.0000 <br> obsstatus1 | 0 Max. :36.000 |
| > | feca1 | fec1added |  | repstatus1 |
| > | Min. $: 0.0000$ | Min. 0.0000 | Min. $\quad 0.0000$ | Min. 00.0000 |
| > | 1st Qu.:0.0000 | 1st Qu.:0.0000 | 1st Qu.:0.0000 1 | 1st Qu.:0.0000 |
| > | Median :0.0000 | Median :0.0000 | Median :1.0000 Med | Median :0.0000 |
| > | Mean :0.2656 | Mean :0.2656 | Mean :0.7469 Mea | Mean :0.2875 |
| > | 3rd Qu.:0.0000 | 3rd Qu.:0.0000 | 3rd Qu.:1.0000 3 | 3rd Qu.:1.0000 |
| > | $\begin{aligned} & \text { Max. } \quad: 7.0000 \\ & \text { fecstatus } 1 \end{aligned}$ | $\begin{aligned} & \text { Max. } \quad: 7.0000 \\ & \text { matstatus } 1 \end{aligned}$ | Max. : 1.0000 <br> alive1 | $\begin{gathered} \text { Max. }: 1.0000 \\ \text { stage } 1 \end{gathered}$ |
| > | Min. $: 0.0000$ | Min. 00.0000 | Min. 0.0000 L | Length: 320 |
| > | 1st Qu.:0.0000 | 1st Qu.:1.0000 | 1st Qu.:1.0000 | Class : character |
| > | Median :0.0000 | Median :1.0000 | Median :1.0000 | Mode :chara |
| > | Mean :0.1344 | Mean :0.7688 | Mean : 0.7688 |  |
| > | 3rd Qu.:0.0000 | 3rd Qu.:1.0000 | 3rd Qu.:1.0000 |  |
| > | Max. :1.0000 | Max. :1.0000 | Max. $: 1.0000$ |  |
| > | stage1index | xpos2 | ypos2 | sizea2 |
| > | Min. : 0.00 | Min. : 46.50 M | Min. : -28.00 Min | Min. 00.000000 |
| > | 1st Qu.: 6.00 | 1st Qu.: 60.10 | 1st Qu.: 23.30 1s | 1st Qu.:0.000000 |
| > | Median : 8.00 | Median : 90.65 M | Median : 77.00 Med | Median :0.000000 |
| > | Mean :14.17 | Mean : 91.19 | Mean : 56.98 Me | Mean :0.009375 |
| > | 3rd Qu.:31.00 | 3rd Qu.:141.80 | 3rd Qu.: 80.40 3rd | 3rd Qu.:0.000000 |
| > | Max. $\quad 51.00$sizeb2 | Max. :173.00 Max | Max. :142.40 Max <br> size2added | Max. :1.000000 |
| > |  | Max. sizec2 |  | Max. repstra2 |
| > | Min. : 0.0000 | Min. : 0.000 | Min. : 0.000 | Min. 00.000000 |
| > | 1st Qu.: 0.0000 | 1st Qu.: 1.000 | 1st Qu.: 1.000 | 1st Qu.:0.000000 |
| > | Median : 0.0000 | Median : 2.000 | Median : 2.000 | Median :0.000000 |
| > | Mean : 0.8969 | Mean : 2.416 | Mean : 3.322 | Mean :0.009375 |
| > | 3rd Qu.: 1.0000 | 3rd Qu.: 3.000 | 3rd Qu.: 4.000 | 3rd Qu.:0.000000 |
| > | $\begin{aligned} & \text { Max. } \quad: 18.0000 \\ & \text { repstrb2 } \end{aligned}$ | Max. : 13.000 | Max. :24.000 <br> feca2 | Max. :1.000000 |
| > |  | repstr2added |  | fec2added |
| > | Min. : 0.0000 | Min. : 0.000 | Min. 00.0000 | Min. 00.0000 |
| > | 1st Qu.: 0.0000 | 1st Qu.: 0.000 | 1st Qu.:0.0000 | 1st Qu.:0.0000 |
| > | Median : 0.0000 | Median : 0.000 | Median :0.0000 | Median :0.0000 |
| > | Mean : 0.8969 | Mean : 1.803 | Mean :0.2906 | Mean :0.2906 |
| > | 3rd Qu.: 1.0000 | 3rd Qu.: 2.000 | 3rd Qu.:0.0000 | 3rd Qu.:0.0000 |
| > | Max. :18.0000 | Max. $: 36.000$ | Max. $\quad 7.0000$ | Max. 7.0000 |
| > | obsstatus2 | repstatus2 | fecstatus2 | matstatus2 alive2 |
| > | Min. 00.0000 | Min. 00.0000 | Min. 00.0000 | Min. :1 Min. :1 |
| > | 1st Qu.:1.0000 | 1st Qu.:0.0000 | 1st Qu.:0.0000 | 1st Qu.:1 1st Qu.:1 |
| > | Median :1.0000 | Median :0.0000 | Median :0.0000 | Median :1 Median :1 |
| > | Mean :0.9531 | Mean :0.3688 | Mean :0.1562 | Mean :1 Mean :1 |
| > | 3rd Qu.:1.0000 | 3rd Qu.:1.0000 | 3rd Qu.:0.0000 | 3rd Qu.:1 3rd Qu.:1 |



Before moving on to supplying R with the proxy transitions that we need to properly parameterize our models, let's take a look at how we might explore our standardized dataset.

### 3.6 Exploring stage-classified standardized datasets

It is often useful to explore our data before using it to build MPMs. One way to explore the data is to look at the frequencies of the stage assignments that R has developed for each monitoring occasion. For this purpose, we can use function actualstage3(). Below, we show the simplest use of this function - to explore single stage assignments, as for ahistorical MPM development.

```
explore_2 <- actualstage3(cypraw_v1)
explore_2
> rowid stageindex stage stage2 stage1 year2 Freq actual_prop
1 2004 D 6 D D D D 2004 0 0.00000000
```

| > 2 |  | 2004 XSm | 7 | XSm | XSm | 2004 | 23 | 0.35384615 |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| > 3 |  | 2004 Sm | 8 | Sm | Sm | 2004 | 24 | 0.36923077 |
| $>4$ |  | 2004 Md | 9 | Md | Md | 2004 | 10 | 0.15384615 |
| $>5$ |  | 2004 Lg | 10 | Lg | Lg | 2004 | 7 | 0.10769231 |
| $>6$ |  | 2004 XLg | 11 | XLg | XLg | 2004 | 1 | 0.01538462 |
| > 7 |  | 2005 D | 6 | D | D | 2005 | 1 | 0.01470588 |
| > 8 |  | 2005 XSm | 7 | XSm | XSm | 2005 | 23 | 0.33823529 |
| > 9 |  | 2005 Sm | 8 | Sm | Sm | 2005 | 27 | 0.39705882 |
| > 10 |  | 2005 Md | 9 | Md | Md | 2005 | 9 | 0.13235294 |
| > 1 |  | 2005 Lg | 10 | Lg | Lg | 2005 | 6 | 0.08823529 |
| > 1 |  | 2005 XLg | 11 | XLg | XLg | 2005 | 2 | 0.02941176 |
| > 1 |  | 2006 D | 6 | D | D | 2006 | 3 | 0.04687500 |
| > 1 |  | 2006 XSm | 7 | XSm | XSm | 2006 | 22 | 0.34375000 |
| > 1 |  | 2006 Sm | 8 | Sm | Sm | 2006 | 27 | 0.42187500 |
| > 1 |  | 2006 Md | 9 | Md | Md | 2006 | 9 | 0.14062500 |
| > 1 |  | 2006 Lg | 10 | Lg | Lg | 2006 | 2 | 0.03125000 |
| > 1 |  | 2006 XLg | 11 | XLg | XLg | 2006 | 1 | 0.01562500 |
| > 1 |  | 2007 D | 6 | D | D | 2007 | 3 | 0.04838710 |
| > 20 |  | 2007 XSm | 7 | XSm | XSm | 2007 | 20 | 0.32258065 |
| > 2 |  | 2007 Sm | 8 | Sm | Sm | 2007 | 25 | 0.40322581 |
| $>2$ |  | 2007 Md | 9 | Md | Md | 2007 | 8 | 0.12903226 |
| $>2$ |  | 2007 Lg | 10 | Lg | Lg | 2007 | 6 | 0.09677419 |
| $>2$ |  | 2007 XLg | 11 | XLg | XLg | 2007 | 0 | 0.00000000 |
| $>2$ |  | 2008 D | 6 | D | D | 2008 | 8 | 0.13114754 |
| $>2$ |  | 2008 XSm | 7 | XSm | XSm | 2008 | 20 | 0.32786885 |
| > 2 |  | 2008 Sm | 8 | Sm | Sm | 2008 | 17 | 0.27868852 |
|  |  | 2008 Md | 9 | Md | Md | 2008 | 12 | 0.19672131 |
|  |  | 2008 Lg | 10 | Lg | Lg | 2008 | 2 | 0.03278689 |
|  |  | 2008 XLg | 11 | XLg | XLg | 2008 | 2 | 0.03278689 |
| > 3 |  | 2009 D | 6 | D | D | 2009 | 0 | 0.00000000 |
| > 3 |  | 2009 XSm | 7 | XSm | XSm | 2009 | 15 | 0.26315789 |
| > 3 | 33 | 2009 Sm | 8 | Sm | Sm | 2009 | 23 | 0.40350877 |
| > 3 |  | 2009 Md | 9 | Md | Md | 2009 | 11 | 0.19298246 |
| > 3 |  | 2009 Lg | 10 | Lg | Lg | 2009 | 6 | 0.10526316 |
| > 3 |  | 2009 XLg | 11 | XLg | XLg | 2009 | 2 | 0.03508772 |

Each stage is shown in each year for which data is available. The number of individuals in each stage (column frequency), and the proportion of individual in that stage in each year (column actual_prop), are both shown. Deaths have been removed, yielding proportions that sum to 1.0.

Function actualstage3() can also be used to explore the frequencies of ages, age-stages, and historical stage pairs. For example, below we show the frequencies of ages.

```
explore_age <- actualstage3(cypraw_v1, check_stage = FALSE, check_age = TRUE)
explore_age
\begin{tabular}{|c|c|c|c|c|c|}
\hline & & & & & actual_prop \\
\hline 1 & 20045 & 5 & 2004 & 65 & 1.00000000 \\
\hline 2 & 20046 & 6 & 2004 & 0 & 0.00000000 \\
\hline 3 & 20047 & 7 & 2004 & 0 & 0.00000000 \\
\hline 4 & 20048 & 8 & 2004 & 0 & 0.00000000 \\
\hline 5 & 20049 & 9 & 2004 & 0 & 0.00000000 \\
\hline 6 & 200410 & 10 & 2004 & 0 & 0.00000000 \\
\hline 7 & 20055 & 5 & 200 & 4 & 0. \\
\hline
\end{tabular}
```

| $>$ | 8 | 2005 | 6 | 6 | 2005 | 64 | 0.94117647 |
| :--- | :--- | ---: | ---: | ---: | ---: | ---: | ---: |
| $>$ | 9 | 2005 | 7 | 7 | 2005 | 0 | 0.00000000 |
| $>$ | 10 | 2005 | 8 | 8 | 2005 | 0 | 0.00000000 |
| $>$ | 11 | 2005 | 9 | 9 | 2005 | 0 | 0.00000000 |
| $>$ | 12 | 2005 | 10 | 10 | 2005 | 0 | 0.00000000 |
| $>$ | 13 | 2006 | 5 | 5 | 2006 | 0 | 0.00000000 |
| $>$ | 14 | 2006 | 6 | 6 | 2006 | 3 | 0.04687500 |
| $>$ | 15 | 2006 | 7 | 7 | 2006 | 61 | 0.95312500 |
| $>$ | 16 | 2006 | 8 | 8 | 2006 | 0 | 0.00000000 |
| $>$ | 17 | 2006 | 9 | 9 | 2006 | 0 | 0.00000000 |
| $>$ | 18 | 2006 | 10 | 10 | 2006 | 0 | 0.00000000 |
| $>$ | 19 | 2007 | 5 | 5 | 2007 | 1 | 0.01612903 |
| $>$ | 20 | 2007 | 6 | 6 | 2007 | 0 | 0.00000000 |
| $>$ | 21 | 2007 | 7 | 7 | 2007 | 2 | 0.03225806 |
| $>$ | 22 | 2007 | 8 | 8 | 2007 | 59 | 0.95161290 |
| $>$ | 23 | 2007 | 9 | 9 | 2007 | 0 | 0.00000000 |
| $>$ | 24 | 2007 | 10 | 10 | 2007 | 0 | 0.00000000 |
| $>$ | 25 | 2008 | 5 | 5 | 2008 | 4 | 0.06557377 |
| $>$ | 26 | 2008 | 6 | 6 | 2008 | 1 | 0.01639344 |
| $>$ | 27 | 2008 | 7 | 7 | 2008 | 0 | 0.00000000 |
| $>$ | 28 | 2008 | 8 | 8 | 2008 | 2 | 0.03278689 |
| $>$ | 29 | 2008 | 9 | 9 | 2008 | 54 | 0.88524590 |
| $>$ | 30 | 2008 | 10 | 10 | 2008 | 0 | 0.00000000 |
| $>$ | 31 | 2009 | 5 | 5 | 2009 | 0 | 0.00000000 |
| $>$ | 32 | 2009 | 6 | 6 | 2009 | 4 | 0.06557377 |
| $>$ | 33 | 2009 | 7 | 7 | 2009 | 1 | 0.01639344 |
| $>$ | 34 | 2009 | 8 | 8 | 2009 | 0 | 0.00000000 |
| $>$ | 35 | 2009 | 9 | 9 | 2009 | 2 | 0.03278689 |
| $>$ | 36 | 2009 | 10 | 10 | 2009 | 54 | 0.88524590 |

The results show a pattern in which only age 5 individuals are alive in the first year, followed by age 5 and age 6 in the second year, etc. This pattern occurs because R has assigned age 5 to all individuals seen in the first year automatically (this is due to the age_offset and prebreeding fields in functions verticalize3() and historicalize3()).

Other explorations are possible. For example, if we removed the check_stage $=$ FALSE option from the inputs, then we would get the frequencies of age-stage combinations for each year. Finally, if we remove check_stage $=$ FALSE, check_age $=$ TRUE and add historical $=$ TRUE, then we can see the frequencies of historical stage pairs, as below.
explore_3 <- actualstage3(cypraw_v1, historical $=$
explore_3

| > 11 | 2004 Lg D | 0 |  | Lg D | Lg | D | 2004 | 0 |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| > 12 | 2004 XLg D | 0 |  | XLg D | XLg | D | 2004 | 0 |
| $>13$ | 2004 D D | 0 |  | D D | D | D | 2004 | 0 |
| > 14 | 2004 XSm D | 0 |  | XSm D | XSm | D | 2004 | 0 |
| > 15 | 2004 Sm XSm | 0 |  | Sm XSm | Sm | XSm | 2004 | 0 |
| $>16$ | 2004 Md XSm | 0 |  | Md XSm | Md | XSm | 2004 | 0 |
| > 17 | 2004 Lg XSm | 0 |  | Lg XSm | Lg | XSm | 2004 | 0 |
| > 18 | 2004 XLg XSm | 0 |  | XLg XSm | XLg | XSm | 2004 | 0 |
| > 19 | 2004 D XSm | 0 |  | D XSm | D | XSm | 2004 | 0 |
| > 20 | 2004 XSm XSm | 0 |  | XSm XSm | XSm | XSm | 2004 | 0 |
| > 21 | 2004 Sm XSm | 0 |  | Sm XSm | Sm | XSm | 2004 | 0 |
| > 22 | 2004 Md Sm | 0 |  | Md Sm | Md | Sm | 2004 | 0 |
| > 23 | 2004 Lg Sm | 0 |  | Lg Sm | Lg | Sm | 2004 | 0 |
| $>24$ | 2004 XLg Sm | 0 |  | XLg Sm | XLg | Sm | 2004 | 0 |
| $>25$ | 2004 D Sm | 0 |  | D Sm | D | Sm | 2004 | 0 |
| > 26 | 2004 XSm Sm | 0 |  | XSm Sm | XSm | Sm | 2004 | 0 |
| > 27 | 2004 Sm Sm | 0 |  | Sm Sm | Sm | Sm | 2004 | 0 |
| > 28 | 2004 Md Sm | 0 |  | Md Sm | Md | Sm | 2004 | 0 |
| $>29$ | 2004 Lg Md | 0 |  | Lg Md | Lg | Md | 2004 | 0 |
| > 30 | 2004 XLg Md | 0 |  | XLg Md | XLg | Md | 2004 | 0 |
| > 31 | 2004 D Md | 0 |  | D Md | D | Md | 2004 | 0 |
| > 32 | 2004 XSm Md | 0 |  | XSm Md | XSm | Md | 2004 | 0 |
| > 33 | 2004 Sm Md | 0 |  | Sm Md | Sm | Md | 2004 | 0 |
| > 34 | 2004 Md Md | 0 |  | Md Md | Md | Md | 2004 | 0 |
| > 35 | 2004 Lg Md | 0 |  | Lg Md | Lg | Md | 2004 | 0 |
| > 36 | 2004 XLg Lg | 0 |  | XLg Lg | XLg | Lg | 2004 | 0 |
| > 37 | 2004 D Lg | 0 |  | D Lg | D | Lg | 2004 | 0 |
| > 38 | 2004 XSm Lg | 0 |  | XSm Lg | XSm | Lg | 2004 | 0 |
| > 39 | 2004 Sm Lg | 0 |  | Sm Lg | Sm | Lg | 2004 | 0 |
| > 40 | 2004 Md Lg | 0 |  | Md Lg | Md | Lg | 2004 | 0 |
| > 41 | 2004 Lg Lg | 0 |  | Lg Lg | Lg | Lg | 2004 | 0 |
| $>42$ | 2004 XLg Lg | 0 |  | XLg Lg | XLg | Lg | 2004 | 0 |
| > 43 | 2005 D XLg | 0 |  | D XLg | D | XLg | 2005 | 0 |
| > 44 | 2005 XSm XLg | 0 |  | XSm XLg | XSm | XLg | 2005 | 0 |
| $>45$ | 2005 Sm XLg | 0 |  | Sm XLg | Sm | XLg | 2005 | 0 |
| > 46 | 2005 Md XLg | 0 |  | Md XLg | Md | XLg | 2005 | 0 |
| > 47 | 2005 Lg XLg | 0 |  | Lg XLg | Lg | XLg | 2005 | 0 |
| $>48$ | 2005 XLg XLg | 0 |  | XLg XLg | XLg | XLg | 2005 | 1 |
| > 49 | 2005 D XLg | 0 |  | D XLg | D | XLg | 2005 | 0 |
| > 50 | 2005 XSm NotAlive | 0 | XSm | NotAlive | XSm | NotAlive | 2005 | 2 |
| > 51 | 2005 Sm NotAlive | 0 | Sm | NotAlive | Sm | NotAlive | 2005 | 2 |
| $>52$ | 2005 Md NotAlive | 0 | Md | NotAlive | Md | NotAlive | 2005 | 0 |
| > 53 | 2005 Lg NotAlive | 0 | Lg | NotAlive | Lg | NotAlive | 2005 | 0 |
| > 54 | 2005 XLg NotAlive | 0 | XLg | NotAlive | XLg | NotAlive | 2005 | 0 |
| $>55$ | 2005 D NotAlive | 0 | D | NotAlive | D | NotAlive | 2005 | 0 |
| > 56 | 2005 XSm NotAlive | 0 | XSm | NotAlive | XSm | NotAlive | 2005 | 2 |
| $>57$ | 2005 Sm D | 0 |  | Sm D | Sm | D | 2005 | 0 |
| $>58$ | 2005 Md D | 0 |  | Md D | Md | D | 2005 | 0 |
| > 59 | 2005 Lg D | 0 |  | Lg D | Lg | D | 2005 | 0 |
| > 60 | 2005 XLg D | 0 |  | XLg D | XLg | D | 2005 | 0 |
| > 61 | 2005 D D | 0 |  | D D | D | D | 2005 | 0 |


| > 62 | 2005 XSm D | 0 | XSm D | XSm | D | 2005 | 0 |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| > 63 | 2005 Sm D | 0 | Sm D | Sm | D | 2005 | 0 |
| > 64 | 2005 Md XSm | 0 | Md XSm | Md | XSm | 2005 | 0 |
| > 65 | 2005 Lg XSm | 0 | Lg XSm | Lg | XSm | 2005 | 1 |
| > 66 | 2005 XLg XSm | 0 | XLg XSm | XLg | XSm | 2005 | 0 |
| > 67 | 2005 D XSm | 0 | D XSm | D | XSm | 2005 | 1 |
| > 68 | 2005 XSm XSm | 0 | XSm XSm | XSm | XSm | 2005 | 15 |
| > 69 | 2005 Sm XSm | 0 | Sm XSm | Sm | XSm | 2005 | 5 |
| > 70 | 2005 Md XSm | 0 | Md XSm | Md | XSm | 2005 | 0 |
| > 71 | 2005 Lg Sm | 0 | Lg Sm | Lg | Sm | 2005 | 0 |
| $>72$ | 2005 XLg Sm | 0 | XLg Sm | XLg | Sm | 2005 | 0 |
| > 73 | 2005 D Sm | 0 | D Sm | D | Sm | 2005 | 0 |
| > 74 | 2005 XSm Sm | 0 | XSm Sm | XSm | Sm | 2005 | 6 |
| > 75 | 2005 Sm Sm | 0 | Sm Sm | Sm | Sm | 2005 | 14 |
| > 76 | 2005 Md Sm | 0 | Md Sm | Md | Sm | 2005 | 4 |
| > 77 | 2005 Lg Sm | 0 | Lg Sm | Lg | Sm | 2005 | 0 |
| > 78 | 2005 XLg Md | 0 | XLg Md | XLg | Md | 2005 | 0 |
| > 79 | 2005 D Md | 0 | D Md | D | Md | 2005 | 0 |
| > 80 | 2005 XSm Md | 0 | XSm Md | XSm | Md | 2005 | 0 |
| > 81 | 2005 Sm Md | 0 | Sm Md | Sm | Md | 2005 | 5 |
| > 82 | 2005 Md Md | 0 | Md Md | Md | Md | 2005 | 4 |
| > 83 | 2005 Lg Md | 0 | Lg Md | Lg | Md | 2005 | 1 |
| > 84 | 2005 XLg Md | 0 | XLg Md | XLg | Md | 2005 | 0 |
| > 85 | 2006 D Lg | 0 | D Lg | D | Lg | 2006 | 0 |
| > 86 | 2006 XSm Lg | 0 | XSm Lg | XSm | Lg | 2006 | 0 |
| > 87 | 2006 Sm Lg | 0 | Sm Lg | Sm | Lg | 2006 | 2 |
| > 88 | 2006 Md Lg | 0 | Md Lg | Md | Lg | 2006 | 3 |
| > 89 | 2006 Lg Lg | 0 | Lg Lg | Lg | Lg | 2006 | 1 |
| > 90 | 2006 XLg Lg | 0 | XLg Lg | XLg | Lg | 2006 | 0 |
| > 91 | 2006 D Lg | 0 | D Lg | D | Lg | 2006 | 0 |
| > 92 | 2006 XSm XLg | 0 | XSm XLg | XSm | XLg | 2006 | 0 |
| > 93 | 2006 Sm XLg | 0 | Sm XLg | Sm | XLg | 2006 | 0 |
| > 94 | 2006 Md XLg | 0 | Md XLg | Md | XLg | 2006 | 0 |
| > 95 | 2006 Lg XLg | 0 | Lg XLg | Lg | XLg | 2006 | 1 |
| > 96 | 2006 XLg XLg | 0 | XLg XLg | XLg | XLg | 2006 | 1 |
| > 97 | 2006 D XLg | 0 | D XLg | D | XLg | 2006 | 0 |
| > 98 | 2006 XSm XLg | 0 | XSm XLg | XSm | XLg | 2006 | 0 |
| > 99 | 2006 Sm NotAlive | 0 | Sm NotAlive | Sm | NotAlive | 2006 | 0 |
| > 100 | 2006 Md NotAlive | 0 | Md NotAlive | Md | NotAlive | 2006 | 0 |
| > 101 | 2006 Lg NotAlive | 0 | Lg NotAlive | Lg | NotAlive | 2006 | 0 |
| > 102 | 2006 XLg NotAlive | 0 | XLg NotAlive | XLg | NotAlive | 2006 | 0 |
| > 103 | 2006 D NotAlive | 0 | D NotAlive | D | NotAlive | 2006 | 0 |
| > 104 | 2006 XSm NotAlive | 0 | XSm NotAlive | XSm | NotAlive | 2006 | 0 |
| > 105 | 2006 Sm NotAlive | 0 | Sm NotAlive | Sm | NotAlive | 2006 | 0 |
| > 106 | 2006 Md D | 0 | Md D | Md | D | 2006 | 0 |
| > 107 | 2006 Lg D | 0 | Lg D | Lg | D | 2006 | 0 |
| > 108 | 2006 XLg D | 0 | XLg D | XLg | D | 2006 | 0 |
| > 109 | 2006 D D | 0 | D D | D | D | 2006 | 0 |
| > 110 | 2006 XSm D | 0 | XSm D | XSm | D | 2006 | 0 |
| > 111 | 2006 Sm D | 0 | Sm D | Sm | D | 2006 | 1 |
| > 112 | 2006 Md D | 0 | Md D | Md | D | 2006 | 0 |


| > 113 | 2006 Lg XSm | O | Lg XSm | Lg | XSm | 2006 | 0 |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| > 114 | 2006 XLg XSm | 0 | XLg XSm | XLg | XSm | 2006 | 0 |
| > 115 | 2006 D XSm | 0 | D XSm | D | XSm | 2006 | 2 |
| > 116 | 2006 XSm XSm | 0 | XSm XSm | XSm | XSm | 2006 | 14 |
| > 117 | 2006 Sm XSm | 0 | Sm XSm | Sm | XSm | 2006 | 4 |
| > 118 | 2006 Md XSm | 0 | Md XSm | Md | XSm | 2006 | 0 |
| > 119 | 2006 Lg XSm | 0 | Lg XSm | Lg | XSm | 2006 | 0 |
| > 120 | 2006 XLg Sm | 0 | XLg Sm | XLg | Sm | 2006 | 0 |
| > 121 | 2006 D Sm | 0 | D Sm | D | Sm | 2006 | 1 |
| > 122 | 2006 XSm Sm | 0 | XSm Sm | xSm | Sm | 2006 | 7 |
| > 123 | 2006 Sm Sm | 0 | Sm Sm | Sm | Sm | 2006 | 16 |
| > 124 | 2006 Md Sm | 0 | Md Sm | Md | Sm | 2006 | 2 |
| > 125 | 2006 Lg Sm | 0 | Lg Sm | Lg | Sm | 2006 | 0 |
| $>$ | actual_prop |  |  |  |  |  |  |
| > 1 | 0.00000000 |  |  |  |  |  |  |
| > 2 | 0.35384615 |  |  |  |  |  |  |
| > 3 | 0.36923077 |  |  |  |  |  |  |
| > 4 | 0.15384615 |  |  |  |  |  |  |
| > 5 | 0.10769231 |  |  |  |  |  |  |
| > 6 | 0.01538462 |  |  |  |  |  |  |
| > 7 | 0.00000000 |  |  |  |  |  |  |
| > 8 | 0.00000000 |  |  |  |  |  |  |
| > 9 | 0.00000000 |  |  |  |  |  |  |
| > 10 | 0.00000000 |  |  |  |  |  |  |
| > 11 | 0.00000000 |  |  |  |  |  |  |
| > 12 | 0.00000000 |  |  |  |  |  |  |
| > 13 | 0.00000000 |  |  |  |  |  |  |
| > 14 | 0.00000000 |  |  |  |  |  |  |
| > 15 | 0.00000000 |  |  |  |  |  |  |
| > 16 | 0.00000000 |  |  |  |  |  |  |
| > 17 | 0.00000000 |  |  |  |  |  |  |
| > 18 | 0.00000000 |  |  |  |  |  |  |
| > 19 | 0.00000000 |  |  |  |  |  |  |
| > 20 | 0.00000000 |  |  |  |  |  |  |
| > 21 | 0.00000000 |  |  |  |  |  |  |
| > 22 | 0.00000000 |  |  |  |  |  |  |
| > 23 | 0.00000000 |  |  |  |  |  |  |
| > 24 | 0.00000000 |  |  |  |  |  |  |
| > 25 | 0.00000000 |  |  |  |  |  |  |
| > 26 | 0.00000000 |  |  |  |  |  |  |
| > 27 | 0.00000000 |  |  |  |  |  |  |
| > 28 | 0.00000000 |  |  |  |  |  |  |
| > 29 | 0.00000000 |  |  |  |  |  |  |
| > 30 | 0.00000000 |  |  |  |  |  |  |
| > 31 | 0.00000000 |  |  |  |  |  |  |
| > 32 | 0.00000000 |  |  |  |  |  |  |
| > 33 | 0.00000000 |  |  |  |  |  |  |
| > 34 | 0.00000000 |  |  |  |  |  |  |
| > 35 | 0.00000000 |  |  |  |  |  |  |
| > 36 | 0.00000000 |  |  |  |  |  |  |
| > 37 | 0.00000000 |  |  |  |  |  |  |


| > 38 | 0.00000000 |
| :---: | :---: |
| > 39 | 0.00000000 |
| > 40 | 0.00000000 |
| > 41 | 0.00000000 |
| > 42 | 0.00000000 |
| > 43 | 0.00000000 |
| > 44 | 0.00000000 |
| > 45 | 0.00000000 |
| > 46 | 0.00000000 |
| > 47 | 0.00000000 |
| > 48 | 0.01587302 |
| > 49 | 0.00000000 |
| > 50 | 0.03174603 |
| > 51 | 0.03174603 |
| > 52 | 0.00000000 |
| > 53 | 0.00000000 |
| > 54 | 0.00000000 |
| > 55 | 0.00000000 |
| > 56 | 0.03174603 |
| > 57 | 0.00000000 |
| > 58 | 0.00000000 |
| > 59 | 0.00000000 |
| > 60 | 0.00000000 |
| > 61 | 0.00000000 |
| > 62 | 0.00000000 |
| > 63 | 0.00000000 |
| > 64 | 0.00000000 |
| > 65 | 0.01587302 |
| > 66 | 0.00000000 |
| > 67 | 0.01587302 |
| > 68 | 0.23809524 |
| > 69 | 0.07936508 |
| > 70 | 0.00000000 |
| > 71 | 0.00000000 |
| > 72 | 0.00000000 |
| $>73$ | 0.00000000 |
| > 74 | 0.09523810 |
| $>75$ | 0.22222222 |
| $>76$ | 0.06349206 |
| $>77$ | 0.00000000 |
| > 78 | 0.00000000 |
| > 79 | 0.00000000 |
| > 80 | 0.00000000 |
| > 81 | 0.07936508 |
| > 82 | 0.06349206 |
| > 83 | 0.01587302 |
| > 84 | 0.00000000 |
| > 85 | 0.00000000 |
| > 86 | 0.00000000 |
| > 87 | 0.03636364 |
| > 88 | 0.05454545 |

```
> 89 0.01818182
> 90 0.00000000
> 91 0.00000000
> 92 0.00000000
> 93 0.00000000
> 94 0.00000000
> 95 0.01818182
> 96 0.01818182
> 97 0.00000000
> 98 0.00000000
> 99 0.00000000
> 100 0.00000000
> 101 0.00000000
> 102 0.00000000
> 103 0.00000000
> 104 0.00000000
> 105 0.00000000
> 106 0.00000000
> 107 0.00000000
> 108 0.00000000
> 109 0.00000000
> 110 0.00000000
> 111 0.01818182
> 112 0.00000000
> 113 0.00000000
> 114 0.00000000
> 115 0.03636364
> 116 0.25454545
> 117 0.07272727
> 118 0.00000000
> 119 0.00000000
> 120 0.00000000
> 121 0.01818182
> 122 0.12727273
> 123 0.29090909
> 124 0.03636364
> 125 0.00000000
> [ reached 'max' / getOption("max.print") -- omitted 127 rows ]
```

Historical stage pairs include stage in time $t$ and stage in time $t-1$. In the first year of observation, the stage in time $t-1$ is unknowable, so is shown above as NotAlive (although it is certainly possible that the individual was alive in the previous year). Starting from the second year, we see many knowable stage pairs with individuals, and as before their proportions within each year sum to 1.0.

Let's now move on to adding supplemental information to our MPMs.

### 3.7 Formatting supplemental data

MPMs are often estimated only partially from available demographic datasets. Some transitions are parameterized using information gathered from other studies, whether through direct input in the matrix or through the development of kernels contingent on external information. Other transitions might also be estimated via proxy transitions elsewhere in the matrix. In lefko3, this information can
be provided in one of two ways. The preferred and most recently developed method is the supplement table, which can be developed using the supplemental () function. This function allows users to create a data frame detailing:

1. specific transitions to set as constants,
2. specific transitions to estimate via proxy transitions, which can also be multiplied by fixed values, and
3. specific multipliers for fecundity in cases in which fecundity estimates resulting from linear modeling must be modified to characterize the full transition (such multipliers can also be used to inform the matrix creation process with the correct fecundity transitions).

Examples might illustrate where this approach is useful. If I lack my own data on subterranean juvenile stages in a plant species, but I have estimates of survival for those stages form another study, then I might use those estimates as constants in the MPM. Further, if I lack demographic data on the development of germinated seeds to the seedling or earliest adult stage, but I have reason to believe that the survival probabilities should be similar to the survival within an observed stage of seedlings or small adults, then I can use the latter survival-transitions as proxies. Finally, if fecundity is a function of seed production, survival to the next year, and germination probability, then germination probability might be estimated via a separate field germination study. I might wish to incorporate this germination probability both as a constant transition, and as a multiplier on estimated fecundity. Supplement tables provide a means to include all of this information.

Function supplemental() was developed to provide a robust, general solution to the addition of these data to MPMs. To use this function, first identify each transition that needs to be altered, and split each transition into stage in times $t+1$, $t$, and, if conducting a historical analysis, $t-1$. These stages must then be entered in the same order across the stage3, stage2, and stage1 fields. For example, if I wished to alter the historical transitions from stage A to B to C, and from A to A to C, then I would enter these two transitions as
stage3 = c("C", "C"), stage2 = c("B", "A"), stage1 = c("A", "A")
If I were interested in ahistorical transitions, then I would either set all stage1 values to NA, such as stage $1=c(N A, N A)$, or simply not include stage1 in the input (the default is a vector of NA elements). If I wished to replace my two transitions with proxy transitions that lefko3 will estimate, then I can designate those proxy transitions using the eststage fields. For example:

```
eststage3 = c("F", "F"), eststage2 = c("E", "D"), eststage1 = c("D", "D")
```

The above would replace my original transitions (A-B-C and A-A-C) with the values estimated for transitions D-E-F and D-D-F, respectively. Alternatively, if I wished to set these two transitions to fixed constants, then I could designate those with givenrate. For example, givenrate $=c(0.15$, 0.25 ) would set these two transitions to 0.15 and 0.25 , respectively. Estimated fecundity, as well as proxy values, can be modified by a multiplier using the multiplier field, and it is important to identify the kind of transition to modify as a survival transition or a fecundity rate using the type field (use type12 also to designate the kind of transition occurs between times $t-1$ and $t$ in the historical case). Fecundity multipliers should be identified separately, since they are not proxy values. The function requires the input of the stageframe used to generate the stages.

We will start off by creating two supplement tables taking all of these sorts of data for the functionbased MPMs. The first is the supplement table for the historical analysis, and the second table covers the ahistorical analysis. Each row refers to a specific transition, and in the historical case, there are codes for 23 given transitions ( 12 for the ahistorical case). The first nine of the historical transitions are set to specific probabilities (six in the ahistorical case), and the next 12 are transitions that will be set to other, estimated transitions (four in the ahistorical case; these are the non-NA transitions
in eststage set below) multiplied by specific constants (either 0.7 or 1.0 ). The final two terms are fecundity multipliers, which will here be used to handle the seed dormancy rate and the germination rate (both including seed survival over the course of the year). Based on the literature, the proxies for entry into the adult classes are transitions from dormancy, as below. However, in the raw dataset, dormancy is not common enough to use as an effective proxy in raw matrix creation. Hence, we can use different proxies for function-based matrix estimation than for raw matrix estimation. Where necessary, we also use rep and mat as shorthand to code for all reproductive stages and all mature stages, respectively.

Here is the historical supplement table. We will also introduce a variable representing the number of seeds expected per fruit. Note that the number of seeds per fruit is actually likely higher than this, but we are using this number to reflect the relatively stable population dynamics that we see in the population.

```
seeds_per_fruit <- 5000
sl_mult <- 0.7
cypsupp3_fb <- supplemental(stage3 = c("SD", "SD", "P1", "P1", "P2", "P3", "SL",
        "SL", "SL", "D", "V1", "V2", "V3", "D", "V1", "V2", "V3", "mat", "mat",
        "mat", "mat", "SD", "P1"),
    stage2 = c("SD", "SD", "SD", "SD", "P1", "P2", "P3", "SL", "SL", "SL", "SL",
        "SL", "SL", "SL", "SL", "SL", "SL", "D", "V1", "V2", "V3", "rep", "rep"),
    stage1 = c("SD", "rep", "SD", "rep", "SD", "P1", "P2", "P3", "SL", "P3", "P3",
        "P3", "P3", "SL", "SL", "SL", "SL", "SL", "SL", "SL", "SL", "mat", "mat"),
    eststage3 = c(NA, NA, NA, NA, NA, NA, NA, NA, NA, "D", "V1", "V2", "V3", "D",
        "V1", "V2", "V3", "mat", "mat", "mat", "mat", NA, NA),
    eststage2 = c(NA, NA, NA, NA, NA, NA, NA, NA, NA, "D", "D", "D", "D", "D",
        "D", "D", "D", "D", "V1", "V2", "V3", NA, NA),
    eststage1 = c(NA, NA, NA, NA, NA, NA, NA, NA, NA, "D", "D", "D", "D", "D",
        "D", "D", "D", "V1", "V1", "V1", "V1", NA, NA),
    givenrate = c(0.08, 0.08, 0.1, 0.1, 0.1, 0.1, 0.1, 0.05, 0.05, NA, NA, NA, NA,
        NA, NA, NA, NA, NA, NA, NA, NA, NA, NA),
    multiplier = c(NA, NA, NA, NA, NA, NA, NA, NA, NA, sl_mult, sl_mult,
        sl_mult, sl_mult, sl_mult, sl_mult, sl_mult, sl_mult, 1, 1, 1, 1,
        0.5 * seeds_per_fruit, 0.5 * seeds_per_fruit),
    type = c("S", "S", "S", "S", "S", "S", "S", "S", "S", "S", "S", "S", "S", "S",
        "S", "S", "S", "S", "S", "S", "S", "R", "R"),
    type_t12 = c("S", "F", "S", "F", "S", "S", "S", "S", "S", "S", "S", "S", "S",
        "S", "S", "S", "S", "S", "S", "S", "S", "S", "S"),
    stageframe = cypframe_fb, historical = TRUE)
cypsupp3_fb
```




Now we can input the ahistorical supplement table.

```
cypsupp2_fb <- supplemental(stage3 = c("SD", "P1", "P2", "P3", "SL", "SL", "D",
    "V1", "V2", "V3", "SD", "P1"),
    stage2 = c("SD", "SD", "P1", "P2", "P3", "SL", "SL", "SL", "SL", "SL", "rep",
        "rep"),
    eststage3 = c(NA, NA, NA, NA, NA, NA, "D", "V1", "V2", "V3", NA, NA),
    eststage2 = c(NA, NA, NA, NA, NA, NA, "D", "D", "D", "D", NA, NA),
    givenrate = c(0.08, 0.1, 0.1, 0.1, 0.05, 0.05, NA, NA, NA, NA, NA, NA),
    multiplier = c(NA, NA, NA, NA, NA, NA, sl_mult, sl_mult, sl_mult, sl_mult,
        0.5 * seeds_per_fruit, 0.5 * seeds_per_fruit),
    type = c("S", "S", "S", "S", "S", "S", "S", "S", "S", "S", "R", "R"),
    stageframe = cypframe_fb, historical = FALSE)
```



These supplement tables provide the best means of adding external data to our MPMs because they allow both specific transitions to be isolated. They also allow the use of shorthand to identify large groups of transitions (e.g. using mat, rep, immat, prop, npr, obs, nobs, or all to signify all mature stages, reproductive stages, immature stages, propagule stages, non-propagule stages, observable stages, unobservable stages, or simply all stages, respectively). Let's now also supply the supplement tables for the raw MPMs.

```
cypsupp3_raw <- supplemental(stage3 = c("SD", "SD", "P1", "P1", "P2", "P2",
        "P3", "SL", "SL", "SL", "D", "XSm", "Sm", "D", "XSm", "Sm", "mat", "mat",
        "mat", "SD", "P1"),
    stage2 = c("SD", "SD", "SD", "SD", "P1", "P1", "P2", "P3", "SL", "SL", "SL",
        "SL", "SL", "SL", "SL", "SL", "D", "XSm", "Sm", "rep", "rep"),
    stage1 = c("SD", "rep", "SD", "rep", "SD", "rep", "P1", "P2", "P3", "SL",
        "SL", "SL", "SL", "P3", "P3", "P3", "SL", "SL", "SL", "mat", "mat"),
    eststage3 = c(NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, "D", "XSm", "Sm", "D",
        "XSm", "Sm", "mat", "mat", "mat", NA, NA),
    eststage2 = c(NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, "XSm", "XSm", "XSm",
        "XSm", "XSm", "XSm", "D", "XSm", "Sm", NA, NA),
    eststage1 = c(NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, "XSm", "XSm", "XSm",
        "XSm", "XSm", "XSm", "XSm", "XSm", "XSm", NA, NA),
    givenrate = c(0.08, 0.10, 0.10, 0.10, 0.10, 0.10, 0.10, 0.05, 0.05, 0.05, NA,
        NA, NA, NA, NA, NA, NA, NA, NA, NA, NA),
    multiplier = c(NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, sl_mult, sl_mult,
        sl_mult, sl_mult, sl_mult, sl_mult, 1, 1, 1, 0.5 * seeds_per_fruit,
```

| cypsupp3_raw |  |  |  |  |  |  |  |  |  |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| > | stage3 | stage2 | stage1 | age2 | eststage3 | eststage2 | eststage1 | estage2 | givenrate |
| $>1$ | SD | SD | SD | NA | <NA> | <NA> | <NA> | NA | 0.08 |
| $>2$ | SD | SD | rep | NA | <NA> | <NA> | <NA> | NA | 0.10 |
| $>3$ | P1 | SD | SD | NA | <NA> | <NA> | <NA> | NA | 0.10 |
| $>4$ | P1 | SD | rep | NA | <NA> | <NA> | <NA> | NA | 0.10 |
| $>5$ | P2 | P1 | SD | NA | <NA> | <NA> | <NA> | NA | 0.10 |
| $>6$ | P2 | P1 | rep | NA | <NA> | <NA> | <NA> | NA | 0.10 |
| $>7$ | P3 | P2 | P1 | NA | <NA> | <NA> | <NA> | NA | 0.10 |
| $>8$ | SL | P3 | P2 | NA | <NA> | <NA> | <NA> | NA | 0.05 |
| $>9$ | SL | SL | P3 | NA | <NA> | <NA> | <NA> | NA | 0.05 |
| > 10 | SL | SL | SL | NA | <NA> | <NA> | <NA> | NA | 0.05 |
| > 11 | D | SL | SL | NA | D | XSm | XSm | NA | NA |
| > 12 | XSm | SL | SL | NA | XSm | XSm | XSm | NA | NA |
| > 13 | Sm | SL | SL | NA | Sm | XSm | XSm | NA | NA |
| > 14 | D | SL | P3 | NA | D | XSm | XSm | NA | NA |
| > 15 | XSm | SL | P3 | NA | XSm | XSm | XSm | NA | NA |
| > 16 | Sm | SL | P3 | NA | Sm | XSm | XSm | NA | NA |
| > 17 | mat | D | SL | NA | mat | D | XSm | NA | NA |
| > 18 | mat | XSm | SL | NA | mat | XSm | XSm | NA | NA |
| > 19 | mat | Sm | SL | NA | mat | Sm | XSm | NA | NA |
| > 20 | SD | rep | mat | NA | <NA> | <NA> | <NA> | NA | NA |
| > 21 | P1 | rep | mat | NA | <NA> | <NA> | <NA> | NA | NA |
| > multiplier convtype convtype_t12 |  |  |  |  |  |  |  |  |  |
| $>1$ |  | NA | 1 |  | 1 |  |  |  |  |
| $>2$ |  | NA | 1 |  | 1 |  |  |  |  |
| $>3$ |  | NA | 1 |  | 1 |  |  |  |  |
| $>4$ |  | NA | 1 |  | 1 |  |  |  |  |
| $>5$ |  | NA | 1 |  | 1 |  |  |  |  |
| $>6$ |  | NA | 1 |  | 1 |  |  |  |  |
| $>7$ |  | NA | 1 |  | 1 |  |  |  |  |
| $>8$ |  | NA | 1 |  | 1 |  |  |  |  |
| $>9$ |  | NA | 1 |  | 1 |  |  |  |  |
| > 10 |  | NA | 1 |  | 1 |  |  |  |  |
| > 11 |  | 0.7 | 1 |  | 1 |  |  |  |  |
| > 12 |  | 0.7 | 1 |  | 1 |  |  |  |  |
| $>13$ |  | 0.7 | 1 |  | 1 |  |  |  |  |
| > 14 |  | 0.7 | 1 |  | 1 |  |  |  |  |
| > 15 |  | 0.7 | 1 |  | 1 |  |  |  |  |
| > 16 |  | 0.7 | 1 |  | 1 |  |  |  |  |
| > 17 |  | 1.0 | 1 |  | 1 |  |  |  |  |
| > 18 |  | 1.0 | 1 |  | 1 |  |  |  |  |
| > 19 |  | 1.0 | 1 |  | 1 |  |  |  |  |
| > 20 |  | 00.0 | 3 |  | 1 |  |  |  |  |
| > 21 |  | 00.0 | 3 |  | 1 |  |  |  |  |

```
cypsupp2_raw <- supplemental(stage3 = c("SD", "P1", "P2", "P3", "SL", "SL", "D",
```

```
    "XSm", "Sm", "SD", "P1"),
stage2 = c("SD", "SD", "P1", "P2", "P3", "SL", "SL", "SL", "SL", "rep","rep"),
eststage3 = c(NA, NA, NA, NA, NA, NA, "D", "XSm", "Sm", NA, NA),
eststage2 = c(NA, NA, NA, NA, NA, NA, "XSm", "XSm", "XSm", NA, NA),
givenrate = c(0.08, 0.10, 0.10, 0.10, 0.05, 0.05, NA, NA, NA, NA, NA),
multiplier = c(NA, NA, NA, NA, NA, NA, sl_mult, sl_mult, sl_mult,
    0.5 * seeds_per_fruit, 0.5 * seeds_per_fruit),
type =c(1, 1, 1, 1, 1, 1, 1, 1, 1, 3, 3),
stageframe = cypframe_raw, historical = FALSE)
```

cypsupp2_raw

| > | stage3 | stage 2 | stage1 | age2 | eststage3 | eststage2 | eststage1 | estage2 | givenrate |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| $>1$ | SD | SD | <NA> | NA | <NA> | <NA> | <NA> | NA | 0.08 |
| $>2$ | P1 | SD | <NA> | NA | <NA> | <NA> | <NA> | NA | 0.10 |
| $>3$ | P2 | P1 | <NA> | NA | <NA> | <NA> | <NA> | NA | 0.10 |
| $>4$ | P3 | P2 | <NA> | NA | <NA> | <NA> | <NA> | NA | 0.10 |
| $>5$ | SL | P3 | <NA> | NA | <NA> | <NA> | <NA> | NA | 0.05 |
| $>6$ | SL | SL | <NA> | NA | <NA> | <NA> | <NA> | NA | 0.05 |
| $>7$ | D | SL | <NA> | NA | D | XSm | <NA> | NA | NA |
| $>8$ | XSm | SL | <NA> | NA | XSm | XSm | <NA> | NA | NA |
| $>9$ | Sm | SL | <NA> | NA | Sm | XSm | <NA> | NA | NA |
| > 10 | SD | rep | <NA> | NA | <NA> | <NA> | <NA> | NA | NA |
| > 11 | P1 | rep | <NA> | NA | <NA> | <NA> | <NA> | NA | NA |

> multiplier convtype convtype_t12
$\begin{array}{llll}> & \text { NA } & 1 & 1\end{array}$
$\begin{array}{llll}>2 & \text { NA } & 1 & 1\end{array}$
$\begin{array}{llll}>3 & \text { NA } & 1 & 1\end{array}$
$\begin{array}{llll}>4 & \text { NA } & 1 & 1\end{array}$
$\begin{array}{llll}>5 & \text { NA } & 1 & 1\end{array}$
$\begin{array}{llll}>6 & \text { NA } & 1\end{array}$
$\begin{array}{llll}>7 & 0.7 & 1 & 1\end{array}$
$\begin{array}{llll}>8 & 0.7 & 1 & 1\end{array}$
$\begin{array}{llll}> & 9 & 0.7 & 1\end{array}$
$\begin{array}{llll}10 & 2500.0 & 3\end{array}$
$\begin{array}{llll}11 & 2500.0 & 3 & 1\end{array}$

### 3.7.1 Historical transitions and supplemental data

The supplement tables above have different dimensions depending both on the life history model used, and on whether the MPM is to be historical or not. A full introduction to historical MPMs is provided in the next chapter, but here we will illustrate why there is such a difference in dimension and why users should be careful in creating these tables.

A standard MPM includes information on the rates and probabilities associated with transitions from the different life history stages at one time to the next. We generally say that such an MPM shows the probability of transition from one stage at the current time to the next, or the expected rate of offspring production from the current time to the next. Because conceptually such a matrix shows the expected rates and probabilities moving forward from the current time, the standard MPM is not considered to have any individual history. As such, we may refer to these MPMs as ahistorical.

A historical MPM includes information on the rates and probabilities associated with the transition from the current time and one time step back, to the next time forward. Such a MPM includes one extra time step of information from the individual's history. For example, in a historical model,
the probability of transition from an extra small adult to a small adult will differ depending on what stage the individual was in immediately before being extra small. So, the probability of transition from extra small at time $t$ to small in time $t+1$ will differ between those individuals that were extra small in time $t-1$ and those individuals that were small in time $t-1$. In an ahistorical model, these transition probabilities would all have the exact same value.

Supplement tables for historical MPMs and IPMs need to be written to include all three-time transitions that cannot be estimated from the dataset. Because a single ahistorical transition represents a transition that does no depend on stage in time $t-1$, this means that a single ahistorical transition is broken down into potentially many historical transitions. The result is that supplement tables for historical MPMs are bigger than those for ahistorical MPMs. Failing to account for any relevant historical transition that is not represented in a dataset forces its value to zero, meaning that failing to account for all relevant transitions in a supplement table will cause a break in the life history that will yield a set of matrices leading to low population growth and often complete insensitivity to alterations in fecundity and other vital rates.

For example, in the raw historical supplement table above, the final two transitions noted are the fecundity transitions. The last of these is mat $\rightarrow$ rep $\rightarrow$ P1, and is shorthand showing that we are setting the transition from all mature stages in time $t-1$ and all reproductive stages in time $t$ to the $1^{\text {st }}$ year protocorm stage in time $t+1$. Since we do not have actual individual demographic data for juvenile stages in our dataset, we also need to specify the next transition, which is the survival transition set rep $\rightarrow$ P1 $->$ P2. Note that here, the transition between times $t-1$ and $t$ is actually a fecundity step, and so refers to the production of offspring by the mother, while the transition from time $t$ to time $t+1$ is a survival probability for the offspring itself. In deVries format, we would instead use AlmostBorn -> P1 -> P2, since deVries format does not allow mixing fecundity and survival transitions together in historical transitions unless they refer to the same individual (more on this issue in the next chapter). We need still more transitions here, including at least the following to connect offspring stages to the adult portions of the life history:
$P 1 \rightarrow P 2 \rightarrow P 3$
$P 2 \rightarrow P 3 \rightarrow S L$
$P 3 \rightarrow S L \rightarrow D$
$P 3 \rightarrow S L \rightarrow X S m$
$P 3 \rightarrow S L \rightarrow S m$
$S L \rightarrow D \rightarrow$ mat
$S L \rightarrow X S m \rightarrow$ mat
$S L \rightarrow S m \rightarrow$ mat

Let's consider another example. In figure 3.2 below, we show the portion of the life history most closely tied to fecundity. The "rep" argument refers to reproductive stages, and the "mat" argument refers to mature stages. The "rep" to "DS" ahistorical transition, then, refers to 24 different transitions in the function-based stageframe, because there are 24 reproductive (i.e. flowering) stages. Counting up the ahistorical transitions, then, we find that we have $24+24+1+1+1=51$ ahistorical transitions shown. In the historical case, the "mat" to "rep" to "DS" transition set refers to $24 \times 49=1176$ transitions. So, counting everything up, we actually see $1176+1176+24+24+1+$ $1=2402$ transitions here. Missing even a single one of these impacts projected population dynamics.

The most important thing to consider in building a supplement table, then, is that all stages must be connected to all other stages as diagrammed in the original life history model. If a single transition noted in the above paragraph is missed, then at least one stage is not properly connected to the other stages in the model, and there is a probability of zero associated with a particular transition that should have a non-zero value. In that circumstance, users will find that their MPM or IPM likely predicts a declining growth rate, and nothing that they do will be able to change it (typically, the


Figure 3.2: The fecundity transition and directly related survival transitions in both ahistorical and historical format
asymptotic growth rate will not just decline, but be completely insensitive to changes in survival or fecundity transitions).

### 3.7.2 An older alternative to the supplement table

Supplement tables provide all of the supplemental information that we need. However, there is an alternative means of supplying this information, and that is through the creation of a reproductive matrix and an overwrite table. The reproductive matrix is an ahistorical matrix detailing where fecundity rates are set, and at what level. Cypripedium candidum produces seeds that germinate by the following growing season (stage P1, or a first year protocorm), or that remain dormant for the next year (stage SD). In the following matrix, we detail that the fecundity of each reproductive stage needs to be split into two between each of these output stages. The actual split places $50 \%$ of the fecundity of a reproductive stage into each category of recruit, where the full fecundity is estimated by linear models that we will create. This reproductive matrix can be used in historical MPM construction, and it is assumed that stage at occasion $t-1$ is set to all.

| > | [,1] | [,2] | [,3] | [,4] | [,5] | [,6] | [,7] | [,8] | [,9] | [,10] | [,11] | [,12] | [,13] |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| $>$ [1,] | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| $>$ [2,] | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| $>$ [3,] | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| $>$ [4,] | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| $>$ [5,] | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| $>$ [6,] | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| > [7,] | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| $>$ [8,] | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| > [9,] | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| $>$ [10,] | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| > [11,] | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| $>[12$, | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| > [13, ] | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |


| > | [14,] | 0 | 0 | 0 | 0 | 00 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| > | [15,] | 0 | 0 | 0 | 0 | 00 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| > | [16,] | 0 | 0 | 0 | 0 | 00 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| > | [17,] | 0 | 0 | 0 | 0 | 00 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| > | [18,] | 0 | 0 | 0 | 0 | 00 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| > |  | [,14] | [,15] | [,16] | [,17] | [,18] | [,19] | [,20] | [,21] | [,22] | [,23] | [,24] | [,25] |
| > | [1,] | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| > | [2,] | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| > | [3,] | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| > | [4, ] | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| > | [5, ] | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| > | [6, ] | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| > | [7, ] | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| > | [8,] | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| > | [9, ] | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| > | [10,] | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| > | [11,] | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| > | [12,] | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| > | [13,] | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| > | [14,] | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| > | [15,] | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| > | [16,] | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| > | [17,] | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| > | [18,] | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| > |  | [,26] | [,27] | [,28] | [,29] | [,30] | [,31] | [,32] | [,33] | [,34] | [,35] | [,36] | [,37] |
| > | [1, ] | 0 | 0 | 0 | 0 | 0 | 2500 | 2500 | 2500 | 2500 | 2500 | 2500 | 2500 |
| $>$ | [2,] | 0 | 0 | 0 | 0 | 0 | 2500 | 2500 | 2500 | 2500 | 2500 | 2500 | 2500 |
| > | [3,] | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| > | [4, ] | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| > | [5, ] | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| $>$ | [6, ] | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| $>$ | [7, ] | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| $>$ | [8,] | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| > | [9,] | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| > | [10,] | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| > | [11,] | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| > | [12,] | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| > | [13,] | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| > | [14,] | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| > | [15,] | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| > | [16,] | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| > | [17,] | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| > | [18,] | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| > |  | [,38] | [,39] | [,40] | [,41] | [,42] | [,43] | [,44] | [,45] | [,46] | [,47] | [,48] | [,49] |
| > | [1,] | 2500 | 2500 | 2500 | 2500 | 2500 | 2500 | 2500 | 2500 | 2500 | 2500 | 2500 | 2500 |
| > | [2,] | 2500 | 2500 | 2500 | 2500 | 2500 | 2500 | 2500 | 2500 | 2500 | 2500 | 2500 | 2500 |
| > | [3,] | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| > | [4, ] | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| > | [5,] | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| > | [6,] | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
|  | [7, ] | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |



Next we will create the overwrite tables, which outline transitions that cannot be estimated from the data set and need to be set by other means. Here is an example of two overwrite tables for the Cypripedium candidum analysis. Note that they are fundamentally similar to supplement tables, but do not allow the addition of multipliers and transition types from occasion $t-1$ to $t$, nor do they involve a check of entered stages against the stageframe.

```
cypover3_fb <- overwrite(stage3 = c("SD", "SD", "P1", "P1", "P2", "P3", "SL",
        "SL", "SL", "D", "V1", "V2", "V3", "D", "V1", "V2", "V3", "mat", "mat",
        "mat", "mat"),
    stage2 = c("SD", "SD", "SD", "SD", "P1", "P2", "P3", "SL", "SL", "SL", "SL",
        "SL", "SL", "SL", "SL", "SL", "SL", "D", "V1", "V2", "V3"),
    stage1 = c("SD", "rep", "SD", "rep", "SD", "P1", "P2", "P3", "SL", "P3", "P3",
        "P3", "P3", "SL", "SL", "SL", "SL", "SL", "SL", "SL", "SL"),
    eststage3 = c(NA, NA, NA, NA, NA, NA, NA, NA, NA, "D", "V1", "V2", "V3", "D",
        "V1", "V2", "V3", "mat", "mat", "mat", "mat"),
    eststage2 = c(NA, NA, NA, NA, NA, NA, NA, NA, NA, "D", "D", "D", "D", "D",
        "D", "D", "D", "D", "V1", "V2", "V3"),
    eststage1 = c(NA, NA, NA, NA, NA, NA, NA, NA, NA, "D", "D", "D", "D", "D",
        "D", "D", "D", "V1", "V1", "V1", "V1"),
    givenrate = c(0.08, 0.08, 0.1, 0.1, 0.1, 0.1, 0.1, 0.05, 0.05, NA, NA, NA, NA,
```

NA, NA, NA, NA, NA, NA, NA, NA))
cypover3_fb
> stage3 stage2 stage1 eststage3 eststage2 eststage1 givenrate convtype

| $>1$ | SD | SD | SD | <NA> | <NA> | <NA> | 0.08 | 1 |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| $>2$ | SD | SD | rep | <NA> | <NA> | <NA> | 0.08 | 1 |
| $>3$ | P1 | SD | SD | <NA> | <NA> | <NA> | 0.10 | 1 |
| $>4$ | P1 | SD | rep | <NA> | <NA> | <NA> | 0.10 | 1 |
| $>5$ | P2 | P1 | SD | <NA> | <NA> | <NA> | 0.10 | 1 |
| $>6$ | P3 | P2 | P1 | <NA> | <NA> | <NA> | 0.10 | 1 |
| $>7$ | SL | P3 | P2 | <NA> | <NA> | <NA> | 0.10 | 1 |
| $>8$ | SL | SL | P3 | <NA> | <NA> | <NA> | 0.05 | 1 |
| > 9 | SL | SL | SL | <NA> | <NA> | <NA> | 0.05 | 1 |
| > 10 | D | SL | P3 | D | D | D | NA | 1 |
| > 11 | V1 | SL | P3 | V1 | D | D | NA | 1 |
| > 12 | V2 | SL | P3 | V2 | D | D | NA | 1 |
| > 13 | V3 | SL | P3 | V3 | D | D | NA | 1 |
| > 14 | D | SL | SL | D | D | D | NA | 1 |
| > 15 | V1 | SL | SL | V1 | D | D | NA | 1 |
| > 16 | V2 | SL | SL | V2 | D | D | NA | 1 |
| > 17 | V3 | SL | SL | V3 | D | D | NA | 1 |
| > 18 | mat | D | SL | mat | D | V1 | NA | 1 |
| > 19 | mat | V1 | SL | mat | V1 | V1 | NA | 1 |
| > 20 | mat | V2 | SL | mat | V2 | V1 | NA | 1 |
| > 21 | mat | V3 | SL | mat | V3 | V1 | NA | 1 |

> convtype_t12
$>1 \quad 1$
$>2 \quad 1$
$>31$
$>41$
$>5$
$>6$
> 7
> 8
> 9
> 101
$>11 \quad 1$
$>12 \quad 1$
$>13 \quad 1$
$>14 \quad 1$
$>15 \quad 1$
$>16 \quad 1$
$>171$
$>18 \quad 1$
$>191$
$>20 \quad 1$
> $21 \quad 1$

```
cypover2_fb <- overwrite(stage3 = c("SD", "P1", "P2", "P3", "SL", "SL", "D",
    "V1", "V2", "V3"),
    stage2 = c("SD", "SD", "P1", "P2", "P3", "SL", "SL", "SL", "SL", "SL"),
```

```
eststage3 = c(NA, NA, NA, NA, NA, NA, "D", "V1", "V2", "V3"),
eststage2 = c(NA, NA, NA, NA, NA, NA, "D", "D", "D", "D"),
givenrate =c(0.08, 0.1, 0.1, 0.1, 0.05, 0.05, NA, NA,NA,NA),
type = c("S", "S", "S", "S", "S", "S", "S", "S", "S"))
```

cypover2_fb
$>11$
$>21$
$>31$
$>41$
$>5 \quad 1$
$>6 \quad 1$
$>71$
$>8 \quad 1$
$>9 \quad 1$
$>10 \quad 1$

Let's now also supply the overwrite tables and reproductive matrix for the raw MPMs.

```
rep_cyp_raw <- matrix(0, 11, 11)
rep_cyp_raw[1:2,7:11] <- 0.5 * seeds_per_fruit
\begin{tabular}{lrrrrrrrrrrr} 
rep_cyp_raw \\
\(>\) & & {\([, 1]\)} & {\([, 2]\)} & {\([, 3]\)} & {\([, 4]\)} & {\([, 5]\)} & {\([, 6]\)} & {\([, 7]\)} & {\([, 8]\)} & {\([, 9]\)} & {\([, 10]\)} \\
\(>\) & {\([1]\),} & 0 & 0 & 0 & 0 & 0 & 0 & 2500 & 2500 & 2500 & 2500 \\
2500 \\
\(>\) & {\([2]\),} & 0 & 0 & 0 & 0 & 0 & 0 & 2500 & 2500 & 2500 & 2500 \\
\(>\) & {\([3]\),} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\(>\) & {\([4]\),} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\(>\) & {\([5]\),} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\(>\) & {\([6]\),} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\(>\) & {\([7]\),} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\(\gg\) & {\([8]\),} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\(>\) & {\([9]\),} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\(>\) & {\([10]\),} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\(>\) & {\([11]\),} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\hline
\end{tabular}
cypover2_raw <- overwrite(stage3 = c("SD", "P1", "P2", "P3", "SL", "SL", "D",
    "XSm", "Sm"),
    stage2 = c("SD", "SD", "P1", "P2", "P3", "SL", "SL", "SL", "SL"),
    eststage3 = c(NA, NA, NA, NA, NA, NA, "D", "XSm", "Sm"),
```



| > 7 | P3 | P2 | P1 | <NA> | <NA> | <NA> | 0.10 | 1 |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| $>8$ | SL | P3 | P2 | <NA> | <NA> | <NA> | 0.05 | 1 |
| $>9$ | SL | SL | P3 | <NA> | <NA> | <NA> | 0.05 | 1 |
| > 10 | SL | SL | SL | <NA> | <NA> | <NA> | 0.05 | 1 |
| > 11 | D | SL | SL | D | XSm | XSm | NA | 1 |
| $>12$ | XSm | SL | SL | XSm | XSm | XSm | NA | 1 |
| $>13$ | Sm | SL | SL | Sm | XSm | XSm | NA | 1 |
| > 14 | D | SL | P3 | D | XSm | XSm | NA | 1 |
| > 15 | XSm | SL | P3 | XSm | XSm | XSm | NA | 1 |
| > 16 | Sm | SL | P3 | Sm | XSm | XSm | NA | 1 |
| $>17$ | mat | D | SL | mat | D | XSm | NA | 1 |
| > 18 | mat | XSm | SL | mat | XSm | XSm | NA | 1 |
| > 19 | mat | Sm | SL | mat | Sm | XSm | NA | 1 |
| > | convtyp |  |  |  |  |  |  |  |
| > 1 |  | 1 |  |  |  |  |  |  |
| $>2$ |  | 1 |  |  |  |  |  |  |
| $>3$ |  | 1 |  |  |  |  |  |  |
| $>4$ |  | 1 |  |  |  |  |  |  |
| $>5$ |  | 1 |  |  |  |  |  |  |
| $>6$ |  | 1 |  |  |  |  |  |  |
| $>7$ |  | 1 |  |  |  |  |  |  |
| $>8$ |  | 1 |  |  |  |  |  |  |
| > 9 |  | 1 |  |  |  |  |  |  |
| > 10 |  | 1 |  |  |  |  |  |  |
| > 11 |  | 1 |  |  |  |  |  |  |
| > 12 |  | 1 |  |  |  |  |  |  |
| > 13 |  | 1 |  |  |  |  |  |  |
| > 14 |  | 1 |  |  |  |  |  |  |
| > 15 |  | 1 |  |  |  |  |  |  |
| > 16 |  | 1 |  |  |  |  |  |  |
| > 17 |  | 1 |  |  |  |  |  |  |
| > 18 |  | 1 |  |  |  |  |  |  |
| > 19 |  | 1 |  |  |  |  |  |  |

The overwrite() approach is much less powerful than the supplement table approach, particularly because it lacks the ability to handle multipliers, it lacks some shorthand stage abbreviations, and it lacks inconsistency checks. The supplemental() approach can also provide supplement tables for all kinds of MPs / IPMs developed in lefko3, including age-by-stage and even Leslie MPMs, and is actively maintained while overwrite() is essentially deprecated. So, we encourage the use of supplemental() rather than overwrite().

### 3.8 Handling complex life histories

Situations may occur in which the life history of the organism includes two sets of stages with transitions only occurring in one direction between them. For example, let's consider the following life history model (figure 3.3).

Raw MPMs have no problem with this situation, provided that each stage has its own, unique set of characteristics. However, function-based MPMs (including IPMs) will have a default tendency to estimate survival transitions from the adult reproductive stages (in blue box) back to the pre-adult non-reproductive stages (in red box). We can deal with this problem using stage grouping.


Figure 3.3: Imaginary life history for perennial plant species $X$, with a pre-adult non-reproductive set of 3 stages that eventually leads to a reproductive set of stages, from which the plant cannot return.

First, we will create a new stageframe that encapsulates all of the stages above. Note that the primary addition that we will make to the sf_create() input is the group option.

```
sizevector <- c(0, 0, 0, 0, 0, 0, 1, 3, 1, 3, 6, 11, 20, 31)
stagevector <- c("SD", "P1", "P2", "P3", "SL", "pD", "pXSm", "pSm", "XSm", "Sm",
    "Md", "Lg", "XLg", "XXLg")
repvector <- c(0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1)
obsvector <- c(0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1)
matvector <- c(0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1)
immvector <- c(0, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0)
propvector <- c(1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
indataset <- c(0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1)
binvec <- c(0, 0, 0, 0, 0, 0.5, 0.5, 1.5, 0.5, 1.5, 1.5, 3.5, 5.5, 5.5)
group <- c(0, 0, 0, 0, 0, 1, 1, 1, 2, 2, 2, 2, 2, 2)
comments <- c("Dormant seed", "1st yr protocorm", "2nd yr protocorm",
    "3rd yr protocorm", "Seedling", "Dormant pre-adult",
    "Extra small pre-adult (1 shoot)", "Small pre-adult (2-4 shoots)",
    "Extra small adult (1 shoot)", "Small adult (2-4 shoots)",
    "Medium adult (5-7 shoots)", "Large adult (8-14 shoots)",
    "Extra large adult (20 shoots)", "Extra extra large adult (>24.5 shoots")
cypframe_alt <- sf_create(sizes = sizevector, stagenames = stagevector,
    repstatus = repvector, obsstatus = obsvector, matstatus = matvector,
    propstatus = propvector, immstatus = immvector, indataset = indataset,
    binhalfwidth = binvec, group = group, comments = comments)
cypframe_alt
> stage size size_b size_c min_age max_age repstatus obsstatus propstatus
\begin{tabular}{llllllllll}
\(>1\) & SD & 0 & NA & NA & NA & NA & 0 & 0 & 1 \\
\(>2\) & P1 & 0 & NA & NA & NA & NA & 0 & 0 & 0
\end{tabular}
```




Now that we have our stageframe, the only thing we need to do is to specify that survival-transitions from group 2 stages to group 1 stages are equal to 0 . We can do this with the supplemental () function. Here, we will set these transitions, and also set the fecundity multipliers as before.

```
cypsupp2_alt <- supplemental(stage3 = c("group1", "SD", "P1"),
    stage2 = c("group2", "rep", "rep"), eststage3 = c(NA, NA, NA),
    eststage2 = c(NA, NA, NA), givenrate =c(0, NA, NA),
    multiplier = c(NA, (0.5 * seeds_per_fruit), (0.5 * seeds_per_fruit)),
    type =c(1, 3, 3), stageframe = cypframe_alt, historical = FALSE)
cypsupp2_alt
l
> multiplier convtype convtype_t12
> 1 NA 1 1
>2 2500 3 1
> 3 2500 
```

This supplement table can now be input into a matrix estimator function to tell that function that these transitions must be 0 . We will see how to do this with different styles of MPM in the next four chapters.

We can now proceed in the next chapter to create raw matrices.

### 3.9 Points to remember

1. Functions verticalize3() and historicalize3() can be used to take horizontally or vertically arranged demographic data and standardize it into the appropriate format for MPM creation and analysis.
2. Function summary_hfv() provides summaries of hfv datasets, including the overall dimensions.
3. Function actualstage3() provides a means of exploring standardized datasets, showing the actual frequencies and proportions of each stage, age, age-stage, or historical stage-pair alive in each year.
4. Function supplemental () provides the means to incorporate given transition values obtained from external studies, set transitions to proxy values to be estimated by lefko3, and incorporate multipiers on survival transitions, fecundity estimates, and proxy survival transitions.
5. Great care must be taken to include every missing transition within the supplement table. Otherwise, models will predict dying populations, and asymptotic analyses will be invariant to changes in survival and fecundity terms.
6. Stage groups can be used to add directionality and other complex relationships between groups of life history stages.

## Chapter 4

## Matrix Models I: Intro to MPMs, and Raw (Empirical) MPMs

An ounce of algebra is worth a ton of verbal argument.

- J.B.S. Haldane

Matrix projection models (MPMs) are representations of the dynamics of a population across all life history stages deemed relevant, across the most relevant time interval (typically one year, but sometimes different, and assumed to be consistent within each analysis). In the simplest MPMs, matrix elements $\left(a_{k j}\right)$ show either the probability of transition for an individual in stage $j$ at occasion $t$ (along the columns), to stage $k$ at occasion $t+1$ (along the rows), or the mean rate of production of offspring into stage $k$ at occasion $t+1$ (along the rows) by adult individuals in reproductive stage $j$ in occasion $t$ (along the columns). Conceptually, each individual is in a particular stage in the instance of monitoring or observation, and then either dies or completes a transition in the interval between that observation occasion and the next. Death is not an explicit life stage and so is not modeled as such, instead becoming a potential endpoint of each transition.


Figure 4.1: MPM for Cypripedium candidum, a North American herbaceous plant

Figure 4.1 shows a simple matrix that assumes five stages in the Cypripedium candidum life cycle. These stages are dormant seed $(S)$, juvenile ( $J$, comprised of protocorms and seedlings), non-flowering but sprouting adult ( $V$ ), non-sprouting adult $(D)$, and flowering adult $(F)$. In equation 4.1, we project a column vector of numbers of individuals alive in each stage in time 1 (e.g., $n_{1, S}$ is the number of individuals in stage $S$ in time 1) a single time step forward using standard matrix multiplication with the projection matrix. The product is another column vector showing the predicted numbers of individuals alive in each stage in the next occasion (e.g., $n_{2, S}$ is the number of individuals in stage $S$
in time 2).

$$
\left[\begin{array}{c}
n_{2, S}  \tag{4.1}\\
n_{2, J} \\
n_{2, V} \\
n_{2, D} \\
n_{2, F}
\end{array}\right]=\left[\begin{array}{ccccc}
S_{S S} & 0 & 0 & 0 & F_{S F} \\
S_{J S} & S_{J J} & 0 & 0 & F_{J F} \\
0 & S_{V J} & S_{V V} & S_{V D} & S_{V F} \\
0 & S_{D J} & S_{D V} & S_{D D} & S_{D F} \\
0 & 0 & S_{F V} & S_{F D} & S_{F F}
\end{array}\right]\left[\begin{array}{c}
n_{1, S} \\
n_{1, J} \\
n_{1, V} \\
n_{1, D} \\
n_{1, F}
\end{array}\right]
$$

Given this, the total population size in any occasion is the sum of the numbers of individuals in all stages in that occasion, or in other words the sum of the elements in the column vector. The true discrete population growth rate $(\lambda)$ is calculated as the ratio of the true total population size in occasion $t+1$ to the true total population size in occasion $t$. Naturally, projections are estimates, so the estimated population growth rate $(\hat{\lambda})$ can be calculated as either the ratio of projected population size in time $t+1$ to projected population size in time $t$, or as the dominant eigenvalue of the projection matrix. In the latter case, estimated $\hat{\lambda}$ is actually the expected population growth rate assuming that the population is at stable stage equilibrium, and so may differ from the true $\lambda$ depending on conditions.

How does one estimate the elements composing the matrix? First, we note that there are two primary kinds of elements - survival-transition elements, and fecundity elements. The former are probabilities denoting the chance of transitioning from one stage in time $t$ to another in stage $t+1$, and the latter are rates denoting the number of offspring produced per reproductive individual in time $t$ that survive to time $t+1$ (this is the pre-breeding model as described in Chapter 2, while fecundity in the post-breeding case would equal the survival of the reproductive parent from time $t$ to time $t+1$ multiplied by the number of offspring produced). In the raw MPM (also called an empirical MPM), which is the first and still dominant kind of MPM, survival-transition elements are estimated as

$$
\begin{equation*}
a_{k j}=\frac{n_{k j}}{\sum_{i=1}^{d} n_{i j}} \tag{4.2}
\end{equation*}
$$

Here, we can refer to matrix elements as $a_{k j}$, where the element represents the rate at which individuals transition to stage $k$ in occasion $t+1$ after having been stage $j$ in occasion $t$. Then, $n_{k j}$ is the number of individuals making this transition, and $d$ is the number of stages plus death in the life history model ( $d=m+1$, where $m$ is the number of life history stages). So the denominator represents the total number of individuals alive in stage $j$ in time $t$ regardless of whether they are alive or dead in time $t+1$. For example, if 100 individuals are alive in stage $D$ in year 2010 , and 20 of these are alive in stage $D$ in year 2011, as are a further 40 in stage $V$ and a further 25 in stage $F$, then the associated transition probabilities are $0.20,0.40$, and 0.25 , respectively. The overall survival probability for stage $D$ is the sum of these transitions, or 0.85 , with $15 \%$ of individuals alive in 2010 dying by the time of the monitoring occasion in 2011. Fecundity in the pre-breeding model can be estimated as

$$
\begin{equation*}
a_{k j}=s_{k} \frac{\sum_{i=1}^{n_{. j}} f e c_{i}}{n_{\cdot j}} \tag{4.3}
\end{equation*}
$$

Here, $s_{k}$ is the survival of newborns to the first-year stage from time $t$ to time $t+1, f e c_{i}$ is the actual fecundity of individual $i$ in time $t, n_{. j}$ is the number of reproductive adults in stage $j$ at time $t$, and the right-hand term, $\frac{\sum_{i=1}^{n . j} f e c_{i}}{n_{. j}}$, is the average offspring production per reproductive adult.

### 4.1 Developing raw MPMs

In package lefko3, the primary function to create basic raw MPMs is rlefko2(). In the code below, we use this function to create a series of raw matrices for our Cypripedium candidum dataset. In our call to rlefko2(), we need to provide $R$ at the very least with the dataset that it will use to
estimate the matrix with (data = cypraw_v1), the stageframe identifying and describing all stages in the life history model (stageframe = cypframe_raw), and the names of the variables coding for stage in times $t+1$ and $t$ (stages $=c(" s t a g e 3 ", ~ " s t a g e 2 "))$. Our call also includes a statement that we wish to estimate matrices for all occasions possible (year = "all", which also requires the identification of the monitoring time variable, yearcol = "year2"). We will also specify that we wish matrices to be estimated for all three patches (patch = "all", which also requires the identification of the patch variable, patchcol = "patchid"). This function assumes default variable names for most of the status variables in $h f v$ datasets, and since we are using size variables other than the defaults, we need to supply those names (size = c("size3added", "size2added")). Finally, we will supply our supplement table (supplement = cypsupp2_raw), and note the name of the variable coding for individual identity (indivcol = "individ").

```
cypmatrix2rp <- rlefko2(data = cypraw_v1, stageframe = cypframe_raw,
    year = "all", patch = "all", stages = c("stage3", "stage2"),
    size = c("size3added", "size2added"), supplement = cypsupp2_raw,
    yearcol = "year2", patchcol = "patchid", indivcol = "individ")
```

cypmatrix2rp
$>$ \$A
$>$ \$A[[1]]
$>$ [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
$>\quad[1] 0.08 ~ 0.0 ~ 0.00 .000 .0000000 \quad 0 \quad$,
$>[2] 0.10 \quad 0.0 \quad 0.00 .000 .0000000 \quad 0 \quad$,
$>\begin{array}{lllllllllll} & {[3,]} & 0.00 & 0.1 & 0.0 & 0.00 & 0.0000000 & 0 & 0.0000000 & 0.0 & 0.0000000\end{array} 0$
$>\begin{array}{llllllllll} & {[4,]} & 0.00 & 0.0 & 0.1 & 0.00 & 0.0000000 & 0 & 0.0000000 & 0.0 \\ 0.0000000 & 0\end{array}$
$>\begin{array}{lllllllllll}> & {[5,]} & 0.00 & 0.0 & 0.0 & 0.05 & 0.0500000 & 0 & 0.0000000 & 0.0 & 0.0000000\end{array} 0$
$>\begin{array}{lllllllllll} & {[6,]} & 0.00 & 0.0 & 0.0 & 0.00 & 0.0000000 & 0 & 0.0000000 & 0.0 & 0.0000000\end{array}$
$\begin{array}{lllllllllll}> & {[7,]} & 0.00 & 0.0 & 0.0 & 0.00 & 0.4454545 & 0 & 0.6363636 & 0.2 & 0.0000000\end{array} 0$
$>[8] \quad 0.00 ~ 0.0 ~ 0.0 ~ 0.00 ~ 0.1909091 \quad 00.2727273 \quad 0.6 \quad 0.6666667 \quad$,
$\begin{array}{lrrrrrrrrrrr}> & {[9,]} & 0.00 & 0.0 & 0.0 & 0.00 & 0.0000000 & 0.0000000 & 0.2 & 0.3333333 & 0 \\ >[10,] & 0.00 & 0.0 & 0.0 & 0.00 & 0.0000000 & 0 & 0.0000000 & 0.0 & 0.0000000 & 0\end{array}$
$>[11] 0.00 ~ 0.0 ~ 0.0 ~ 0.00 ~ 0.0000000 \quad 00.0000000 \quad 0.0 \quad 0.0000000 \quad$,
$>\quad[, 11]$
$>\quad[1] \quad$,
$>\quad[2] \quad$,
$>[3] \quad$,
$>[4] \quad$,
$>\quad[5] \quad$,
$>\quad[6] \quad$,
$>\quad[7] \quad$,
$>\quad[8] \quad$,
$>\quad[9] \quad$,
$>$ [10,] 0
$>[11] \quad$,
$>$
$>$ \$A[[2]]
$>\quad[, 1][, 2][, 3][, 4] \quad[, 5][, 6] \quad[, 7] \quad[, 8][, 9][, 10][, 11]$
$>\begin{array}{lllllllllllllllll}> & {[1,]} & 0.08 & 0.0 & 0.0 & 0.00 & 0.0000 & 0 & 312.500 & 1111.1111111 & 1250 & 0 & 0\end{array}$
$>\begin{array}{lllllllllllll}l & {[2,]} & 0.10 & 0.0 & 0.0 & 0.00 & 0.0000 & 0 & 312.500 & 1111.1111111 & 1250 & 0 & 0\end{array}$
$>\begin{array}{llllllllllll} & {[3,]} & 0.00 & 0.1 & 0.0 & 0.00 & 0.0000 & 0 & 0.000 & 0.0000000 & 0 & 0\end{array} 0$
$\begin{array}{llllllllllll}> & {[4,]} & 0.00 & 0.0 & 0.1 & 0.00 & 0.0000 & 0 & 0.000 & 0.0000000 & 0 & 0 \\ > & {[5,]} & 0.00 & 0.0 & 0.0 & 0.05 & 0.0500 & 0 & 0.000 & 0.0000000 & 0 & 0\end{array}$



```
> [2,] 5000.00 17500
> [3,] 0.00 0
> [4,] 0.00 0
> [5,] 0.00 0
> [6,] 0.00 0
> [7,] 0.00 0
> [8,] 0.50 0
> [9,] 0.25 0
> [10,] 0.25 0
> [11,] 0.00 1
>
> $A[[8]]
> [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
> [1,] 0.08 0.0 0.0 0.00 0.0000000 0 277.7777778 781.2500 6250.0 5000
> [2,] 0.10}00.0 0.0 0.00 0.0000000 0 277.7777778 781.2500 6250.0 5000
> [3,] 0.00 0.1 0.0 0.00 0.0000000 0
```



```
> [5,] 0.00 0.0 0.0 0.05 0.0500000 0 0.0000000 0.0000
> [6,] 0.00 0.0 0.0 0.00 0.0000000 0
> [7,] 0.00 0.0 
> [9,] 0.00 0.0 0.0 0.00 0.0000000 0 0.1111111 
> [10,] 0.00 0.0 0.0 0.00 0.0000000 
> [11,] 0.00 0.0 0.0 0.00 0.0000000 0
> [1,] 10000
> [2,] 10000
> [3,] 0
> [4,] 0
> [5,] 0
> [6,] 0
> [7,] 0
> [8,] 0
> [9,] 0
> [10,] 1
> [11,] 0
>
> $A[[9]]
> [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
> [1,] 0.08 0.0 0.0 0.00 0.0000000 0.0000000 0.0000000 250.0 0.0 1250.00
> [2,] 0.10 0.0 0.0 0.00 0.0000000 0.0000000 0.0000000 250.0 0.0 1250.00
> [3,] 0.00 0.1 0.0 0.00 0.0000000 0.0000000 0.0000000
> [4,] 0.00 0.0 0.1 0.00 0.0000000 0.0000000 0.0000000
> [5,] 0.00 0.0 0.0 0.05 0.0500000 0.0000000 0.0000000
> [6,] 0.00 0.0 0.0 0.00 0.1272727 0.3333333 0.1818182 
> [7,] 0.00 0.0 0.0 0.00 0.1909091 0.3333333 0.2727273 
```



```
> [9,] 0.00 0.0
```



```
> [11,] 0.00 0.0 0.0 0.00 0.0000000 0.0000000 0.0000000 0.0 0.0
> [,11]
```

```
> [1,] 0
> [2,] 0
> [3,] 0
> [4,] 0
> [5,] 0
> [6,] 0
> [7,] 0
> [8,] 0
> [9,] 0
> [10,] 0
> [11,] 0
>
> $A[[10]]
> [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11]
> [1,] 0.08 0.0 0.0 0.00 0.00 0 0.0000000 0.000 714.2857143 1250 2500
> [2,] 0.10 0.0 0.0 0.00 0.00 0 0 0.0000000 0.000 714.2857143 1250 2500
> [3,] 0.00 0.1 0.0 0.00 0.00 0 0.0000000 0.000 0.0000000 0 0
> [4,] 0.00 0.0 0.1 0.00 0.00 0.00.0000000 0.000 0.0000000 0 0 0
> [5,] 0.00 0.0 0.0 0.05 0.05 0.0.0.0000000 0.000 
> [6,] 0.00 0.0 0.0 0.00 0.00 
> [7,] 0.00 0.0 
> [8,] 0.00 
> [10,] 0.00 0.0 0.0 0.00 0.00 0.00.0000000 0.000
> [11,] 0.00 0.0 0.0 0.00 0.00 00.0000000 0.000 0.0000000 
>
> $A[[11]]
> [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
> [1,] 0.08 0.0 0.0 0.00 0.0000000 0 0.0000000 625.00 833.3333333 0
> [2,] 0.10
> [3,] 0.00 0.1 0.0 0.00 0.0000000 
> [4,] 0.00 0.0 0.1 0.00 0.0000000 
```



```
> [6,] 0.00 0.0 0.0 0.00 0.0000000 
```



```
> [9,] 0.00 0.0 0.0 0.00 0.0000000 0 0.0000000 0.25 0.6666667 0
> [10,] 0.00 0.0 
> [11,] 0.00 0.0 0.0 0.00 0.0000000 0 0.0000000 0.00 0.0000000 0
> [,11]
> [1,] 7500
> [2,] 7500
> [3,] 0
> [4,] 0
> [5,] 0
> [6,] 0
> [7,] 0
> [8,] 0
> [9,] 0
> [10,] 0
> [11,] 1
```

```
>
> $A[[12]]
> [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11]
> [1,] 0.08 0.0 0.0 0.00 0.000 0 0.00 0.0000000 4166.6666667 1250 5000
> [2,] 0.10 0.0 0.0 0.00 0.000 0 0.00 0.0000000 4166.6666667 1250 5000
> [3,] 0.00 0.1 0.00}00.000.00
> [4,] 0.00
> [5,] 0.00 0.0
> [6,] 0.00 0.0
> [7,] 0.00 0.0
> [8,] 0.00 0.0
> [9,] 0.00}00.0 0.0 0.00 0.000
> [10,] 0.00 0.0 0.0 0.00 0.000 
>
> $A[[13]]
> [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11]
> [1,] 0.08 0.0 0.0 0.00 0.000 0 0.00 1071.4285714 2500.0000000 2500 0
```



```
> [3,] 0.00 0.1 0.0 0.00 0.000 0 0.00 0.0000000 0.0000000 0 0
> [4,] 0.00 0.0 0.1 0.00 0.000 0 0.00 0.0000000
> [5,] 0.00 0.0
> [6,] 0.00 0.0
> [7,] 0.00 0.0
> [8,] 0.00 0.0 0.0 0.00 0.175 0
```



```
> [11,] 0.00 0.0
>
> $A[[14]]
> [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
> [1,] 0.08 0.0 0.0 0.00 0.0000000 0 0.0000000 416.6666667 0.0000000 0.0
> [2,] 0.10 0.0 0.0 0.00 0.0000000 0 0.0000000 416.6666667 0.0000000 0.0
> [3,] 0.00 0.1 0.0 0.00 0.0000000 0 0.0000000 0.0000000 0.0000000 0.0
```



```
> [5,] 0.00 0.0 0.0 0.05 0.0500000 0 0.0000000 0.0000000 0.0000000}00.
> [6,] 0.00 0.0 0.0 0.00 0.2333333 0 0.3333333 0.1666667 0.0000000 0.0
> [7,] 0.00 0.0 0.0 0.00 0.2333333 0 0.3333333 0.1666667 0.3333333 0.0
> [8,] 0.00 0.0 0.0 0.00 0.0000000 0 0.0000000 0.3333333 0.3333333 0.5
> [9,] 0.00 0.0 0.0 0.00 0.0000000 0 0.3333333 0.3333333 0.3333333 0.0
> [10,] 0.00 0.0 0.0 0.00 0.0000000 0 0.0000000 0.0000000 0.0000000 0.0
> [11,] 0.00 0.0 0.0 0.00 0.0000000 0 0.0000000 0.0000000 0.0000000 0.5
> [,11]
> [1,] 0
> [2,] 0
> [3,] 0
> [4,] 0
> [5,] 0
> [6,] 0
> [7,] 0
> [8,] 0
```

```
> [9,] 0
> [10,] 0
> [11,] 0
>
> $A[[15]]
> [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11]
> [1,] 0.08 0.0 0.0 0.00 0.000 0.0 0.00 0.00 625 0 0
> [2,] 0.10
> [3,] 0.00
> [4,]}0.0
```



```
> [7,] 0.00 0.0
> [8,] 0.00
> [9,] 0.00
> [10,] 0.00 
>
> $U
> $U[[1]]
> [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11]
> [1,] 0.08 0.0 0.0 0.00 0.0000000 0 0.0000000 0.0 0.0000000 0 0
> [2,] 0.10
```



```
> [4,] 0.00 0.0
> [5,] 0.00 0.0 0.0 0.05 0.0500000 0 0.0000000 0.0 0.0000000 0 0
> [6,] 0.00 0.0 0.0 0.00 0.0000000 0 0.0000000 0.0 0.0000000 0 0 0
> [7,] 0.00 0.0 0.0 0.00 0.4454545 0 0.6363636 0.2 0.0000000 0 0
> [8,] 0.00 0.0 0.0 0.00 0.1909091 
> [9,] 0.00 0.0.0
> [11,] 0.00 0.0
> $U[[2]]
> [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11]
```




```
> [3,] 0.00 0.1 0.0 0.00 0.0000 0 0.000 0.0000000 0 0 0 0
> [4,] 0.00
> [5,] 0.00 0.0 0.0 0.05 0.0500 
> [6,] 0.00 0.0.0
> [8,] 0.00 0.0 0.0 0.00 0.0000 0.0.000 0.4444444 
> [9,] 0.00 0.0 0.0 0.00 0.0000 
> [10,] 0.00 0.0 0.0 0.00 0.0000 
>
> $U[[3]]
> [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11]
> [1,] 0.08 0.0 0.0 0.00 0.0000000 00.0000000 0 0.00 0 0 0
```


$>$

```
> $U[[6]]
> [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11]
> [1,] 0.08 0.0 0.0 0.00 0.00000000 0 0.0000000 0.0000000 0.00 0.00 0
> [2,] 0.10 0.0 0.0 0.00 0.00000000 0 0.0000000 0.0000000 0.00 0.00 0
> [3,] 0.00 0.1 0.0 0.00 0.00000000
> [4,] 0.00 0.0 0.1 0.00 0.00000000
> [5,] 0.00 0.0 0.0 0.05 0.05000000
> [6,] 0.00 0.0
> [7,] 0.00 0.0 0.0 0.00 0.46666667
> [8,] 0.00 0.0 0.0 0.00 0.07777778
> [9,] 0.00 0.0}00.00.00 0.0000000
> [10,] 0.00 0.0}00.00.00 0.0000000
> [11,] 0.00 0.0}00.0 0.00 0.0000000
>
> $U[[7]]
> [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
> [1,] 0.08 0.0 0.0 0.00 0.00000000 0 0.00000000 0.00000000 0.00 0.00
> [2,] 0.10 0.0 0.0 0.00 0.00000000 0 0.00000000 0.00000000 0.00 0.00
> [3,] 0.00 0.1 0.0 0.00 0.00000000 0 0.00000000 0.00000000 0.00 0.00
> [4,] 0.00 0.0}00.10.000.0000000
> [5,] 0.00 0.0 0.0 0.05 0.05000000
> [6,] 0.00 0.0}00.00.000.0636363
> [7,] 0.00 0.0 0.0 0.00 0.31818182
> [8,] 0.00}00.0 0.0 0.00 0.25454545
> [9,] 0.00 0.0}00.00.00 0.00000000
> [10,] 0.00 0.0}00.00.00 0.0000000
> [11,] 0.00 0.0 0.0 0.00 0.00000000
> [,11]
> [1,] 0
> [2,] 0
> [3,] 0
> [4,] 0
> [5,] 0
> [6,] 0
> [7,] 0
> [8,] 0
> [9,] 0
> [10,] 0
> [11,] 1
>
> $U[[8]]
> [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11]
> [1,] 0.08 0.0 0.0 0.00 0.0000000 0 0.0000000 0.0000 0.0 0 0
```



```
> [3,] 0.00 0.1 0.0 0.00 0.0000000 0 0.0000000 0.0000 0.0 0 0
> [4,] 0.00 0.0 0.1 0.00 0.0000000 0 0.0000000 0.0000 0.0 0 0
```




```
> [7,] 0.00
> [8,] 0.00 0.0 0.0 0.00 0.2333333 
```




```
> [5,] 0.00 0.0 0.0 0.05 0.050 0 0.00 0.0000000 0.0000000 0 0
[6,] 0.00 0.0
[7,] 0.00 0.0
    [8,] 0.00 0.0
    [9,] 0.00 0.0
[10,] 0.00 0.0}00.00.00 0.00
[11,] 0.00 0.0}00.00.00 0.00
$U[[13]]
> [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11]
> [1,] 0.08 0.0 0.0 0.00 0.000 0 0.00 0.0000000 0.0000000 0 0
> [2,] 0.10 0.0 0.0 0.00 0.000 0 0.00 0.0000000 0.0000000 0 0
> [3,] 0.00 0.1 0.0 0.00 0.000 0 0.00 0.0000000 0.0000000 0 0
> [4,] 0.00 0.0 0.1 0.00 0.000 0 0.00 0.0000000 0.0000000 0 0
> [5,] 0.00 0.0 0.0 0.05 0.050 0 0.00 0.0000000 0.0000000 0 0
> [6,] 0.00 0.0 0.0 0.00 0.000 0 0.00 0.0000000 0.0000000 0 0
> [7,] 0.00 0.0 0.0 0.00 0.350 0 0.50 0.0000000 0.3333333 0 0
> [8,] 0.00 0.0 0.0 0.00 0.175 0 0.25 0.5714286 0.3333333 0 0
> [9,] 0.00 0.0 0.0 0.00 0.000 0 0.00 0.2857143 0.3333333 0
> [10,] 0.00 0.0.0
> [11,] 0.00 0.0 0.0 0.00 0.000
$U[[14]]
        [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
    [1,] 0.08 0.0 0.0 0.00 0.0000000 0 0.0000000 0.0000000 0.0000000}00.
```



```
    [3,] 0.00 0.1 0.0 0.00 0.0000000 0 0.0000000 0.0000000 0.0000000}00.
    [4,] 0.00 0.0 0.1 0.00 0.0000000 0 0.0000000 0.0000000 0.0000000 0.0
> [5,] 0.00 0.0 0.0 0.05 0.0500000 0 0.0000000 0.0000000 0.0000000 0.0
> [6,] 0.00 0.0 0.0 0.00 0.2333333 0 0.3333333 0.1666667 0.0000000 0.0
> [7,] 0.00 0.0 0.0 0.00 0.2333333 0 0.3333333 0.1666667 0.3333333 0.0
> [8,] 0.00 0.0 0.0 0.00 0.0000000 0 0.0000000 0.3333333 0.3333333 0.5
> [9,] 0.00 0.0 0.0 0.00 0.0000000 0 0.3333333 0.3333333 0.3333333 0.0
> [10,] 0.00 0.0 0.0 0.00 0.0000000 0 0.0000000 0.0000000 0.0000000 0.0
> [11,] 0.00 0.0 0.0 0.00 0.0000000 0 0.0000000 0.0000000 0.0000000 0.5
> [,11]
> [1,] 0
> [2,] 0
> [3,] 0
> [4,] 0
> [5,] 0
> [6,] 0
> [7,] 0
> [8,] 0
> [9,] 0
> [10,] 0
> [11,] 0
>
> $U[[15]]
> [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11]
```



| > | [2,] | 0.10 | 0.0 | 0.0 | 0.00 | 0.000 | 0.0 | 0.0 .00 | 0.00 |  | 0 | 0 | 0 |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| > | [3,] | 0.00 | 0.1 | 0.0 | 0.00 | 0.000 | 0.0 | 0.0 .00 | 0.00 |  | 0 | 0 | 0 |
| > | [4, ] | 0.00 | 0.0 | 0.1 | 0.00 | 0.000 | 0.0 | 0.000 | 0.00 |  | 0 | 0 | 0 |
| > | [5,] | 0.00 | 0.0 | 0.0 | 0.05 | 0.050 | 0.0 | 0.0 .00 | 0.00 |  | 0 | 0 | 0 |
| > | [6,] | 0.00 | 0.0 | 0.0 | 0.00 | 0.000 | 0.0 | 0.0 .00 | 0.00 |  | 0 | 0 | 0 |
| > | [7, ] | 0.00 | 0.0 | 0.0 | 0.00 | 0.175 | 0.5 | 5 0.25 | 0.00 |  | 0 | 0 | 0 |
| > | [8,] | 0.00 | 0.0 | 0.0 | 0.00 | 0.525 | 0.5 | . 0.75 | 0.50 |  | 0 | 0 | 0 |
| > | [9,] | 0.00 | 0.0 | 0.0 | 0.00 | 0.000 | 0.0 | 0.0 .00 | 0.25 |  | 1 | 0 | 0 |
| > | [10,] | 0.00 | 0.0 | 0.0 | 0.00 | 0.000 | 0.0 | 0.000 | 0.25 |  | 0 | 0 | 0 |
| > | [11,] | 0.00 | 0.0 | 0.0 | 0.00 | 0.000 | 0.0 | 0.000 | 0.00 |  | 0 | 0 | 1 |
| > |  |  |  |  |  |  |  |  |  |  |  |  |  |
| > |  |  |  |  |  |  |  |  |  |  |  |  |  |
| > | \$F |  |  |  |  |  |  |  |  |  |  |  |  |
|  | > \$F[[1]] |  |  |  |  |  |  |  |  |  |  |  |  |
| > |  | [,1] | [,2] | [,3] | [,4] | [,5] | [,6] | [,7] | [, 8] |  | [,9] | [,10] | [,11] |
| > | [1,] | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 500 | 1666. | . 667 | 0 | 0 |
| > | [2,] | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 500 | 1666. | . 667 | 0 | 0 |
| > | [3,] | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |  | . 000 | 0 | 0 |
| $>$ | [4, ] | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |  | . 000 | 0 | 0 |
| > | [5, ] | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |  | . 000 | 0 | 0 |
| > | [6, ] | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |  | . 000 | 0 | 0 |
| > | [7, ] | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |  | . 000 | 0 | 0 |
| > | [8,] | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |  | . 000 | 0 | 0 |
| > | [9,] | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |  | . 000 | 0 | 0 |
| > | [10,] | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |  | . 000 | 0 | 0 |
| > | [11,] | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |  | . 000 | 0 | 0 |
| > |  |  |  |  |  |  |  |  |  |  |  |  |  |
| \$F[[2]] |  |  |  |  |  |  |  |  |  |  |  |  |  |
| > |  | [,1] | [,2] | [,3] | [,4] | [,5] | [,6] | [,7] |  | [,8] | [,9] | [,10] | [,11] |
| > | [1, ] | 0 | 0 | 0 | 0 | 0 | 0 | 312.5 | 1111 | 1.111 | 1250 | 0 | 0 |
| > | [2,] | 0 | 0 | 0 | 0 | 0 | 0 | 312.5 | 1111 | 1.111 | 1250 | 0 | 0 |
| > | [3,] | 0 | 0 | 0 | 0 | 0 | 0 | 0.0 |  | 0.000 | 0 | 0 | 0 |
| > | [4, ] | 0 | 0 | 0 | 0 | 0 | 0 | 0.0 |  | 0.000 | 0 | 0 | 0 |
| > | [5,] | 0 | 0 | 0 | 0 | 0 | 0 | 0.0 |  | 0.000 | 0 | 0 | 0 |
| > | [6,] | 0 | 0 | 0 | 0 | 0 | 0 | 0.0 |  | 0.000 | 0 | 0 | 0 |
| > | [7, ] | 0 | 0 | 0 | 0 | 0 | 0 | 0.0 |  | 0.000 | 0 | 0 | 0 |
| > | [8,] | 0 | 0 | 0 | 0 | 0 | 0 | 0.0 |  | 0.000 | 0 | 0 | 0 |
| > | [9,] | 0 | 0 | 0 | 0 | 0 | 0 | 0.0 |  | 0.000 | 0 | 0 | 0 |
| > | [10,] | 0 | 0 | 0 | 0 | 0 | 0 | 0.0 |  | 0.000 | 0 | 0 | 0 |
| > | [11,] | 0 | 0 | 0 | 0 | 0 | 0 | 0.0 |  | 0.000 | 0 | 0 | 0 |
| > |  |  |  |  |  |  |  |  |  |  |  |  |  |
| > \$F[[3] $]$ |  |  |  |  |  |  |  |  |  |  |  |  |  |
| > |  | [,1] | [,2] | [,3] | [,4] | [,5] | [,6] | [,7] | [,8] | [,9] | [,10] | [,11 |  |
| > | [1, ] | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |  | 0 | 0 |
| > | [2,] | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |  | 0 | 0 |
| > | [3,] | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |  | 0 | 0 |
| > | [4, ] | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |  | 0 | 0 |
| > | [5,] | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |  | 0 | 0 |
| > | [6,] | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |  | 0 | 0 |
|  | [7,] | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |  | 0 | 0 |
|  | [8,] | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |  | 0 | 0 |

```
> >[9,] 
>
> $F[[4]]
> [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11]
> [1,] 0
> [2,] 0
> [3,] 0
> [4,] 
> [5,] 
> [6,] 0
> [7,] 
> [8,] 
> [10,] 
>
> $F[[5]]
> [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11]
> [1,] 0
> [2,] 0
> [3,] 0
> [4,]}00000\mp@code{0
> [5,] 0
> [6,] [10
> [7,] 
> [9,] 
> [10,] 
>
> $F[[6]]
> [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11]
> [1,] 0
> [2,] 0
> [3,] 0
> [4,] 0
> [5,] 0
> [6,] 0
> [7,] 0
> [8,] 0
> [9,] 
> [10,] 
>
> $F[[7]]
> [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11]
> [1,] 0
> [2,] 
```



```
> $F[[11]]
> [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11]
> [1,] 0
> [2,] 0
> [3,] 0
> [4,] 0
> [5,] 0
> [6,] 
> [7,] 
> [9,] 
> [10,] 
>
> $F[[12]]
\begin{tabular}{lrrrrrrrrrrrr}
\(>\) & & {\([, 1]\)} & {\([, 2]\)} & {\([, 3]\)} & {\([, 4]\)} & {\([, 5]\)} & {\([, 6]\)} & {\([, 7]\)} & {\([, 8]\)} & {\([, 9]\)} & {\([, 10]\)} & {\([, 11]\)} \\
\(>\) & {\([1]\),} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 4166.667 & 1250 & 5000 \\
\(>\) & {\([2]\),} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 4166.667 & 1250 & 5000 \\
\(>\) & {\([3]\),} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.000 & 0 & 0 \\
\(>\) & {\([4]\),} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.000 & 0 & 0 \\
\(>\) & {\([5]\),} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.000 & 0 & 0 \\
\(>\) & {\([6]\),} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.000 & 0 & 0 \\
\(>\) & {\([7]\),} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.000 & 0 & 0 \\
\(>\) & {\([8]\),} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.000 & 0 & 0 \\
\(>\) & {\([9]\),} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.000 & 0 & 0 \\
\(>\) & {\([10]\),} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.000 & 0 & 0 \\
\(>\) & {\([11]\),} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.000 & 0 & 0
\end{tabular}
>
> $F[[13]]
> [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11]
> [1,] 0
> [2,] 
> [4,] [10 0
> [5,] 0
> [6,] 0
> [7,] 
> [8,] 
> [10,] 
>
> $F[[14]]
> [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11]
> [1,] 0
> [2,] 
> [3,] 0
> [4,] 0
> [5,] 
> [7,] 0
```

| $>$ | $[8]$, | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0.0000 | 0 | 0 | 0 |
| :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- |
| $>[9]$, | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0.0000 | 0 | 0 | 0 |  |
| $>[10]$, | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0.0000 | 0 | 0 | 0 |  |
| $>[11]$, | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0.0000 | 0 | 0 | 0 |  |

$>$
> \$F[[15]]
$>\quad[, 1][, 2][, 3][, 4][, 5][, 6][, 7][, 8][, 9][, 10][, 11]$
$>\begin{array}{cccccccccccc}{[1,]} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 625 & 0 & 0\end{array}$
$>\begin{array}{llllllllllll}{[2,]} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 625 & 0 & 0\end{array}$
$>\begin{array}{cccccccccccc}{[3,]} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\end{array}$
$>\begin{array}{cccccccccccc}{[4,]} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\end{array}$
$>\begin{array}{llllllllllll}{[5,]} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\end{array}$
$>\begin{array}{cccccccccccc}{[6,]} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\end{array}$
$>\begin{array}{cccccccccccc}{[7,]} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\end{array}$
$>\begin{array}{cccccccccccc}{[8,]} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\end{array}$
$\begin{array}{llllllllllll}>[9,] & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ >[10,] & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ >[11,] & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\end{array}$
$>\left[\begin{array}{llllllllllll}>11,] & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\end{array}\right.$
$>$
> \$ahstages

$>$ repstatus obsstatus propstatus immstatus matstatus entrystage indataset

| $\gg 1$ | 0 | 0 | 1 | 0 | 0 | 1 | 0 |
| :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- |
| $>2$ | 0 | 0 | 0 | 1 | 0 | 1 | 0 |
| $>$ | 3 | 0 | 0 | 0 | 1 | 0 | 0 |
| $\gg$ | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| $>5$ | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| $>6$ | 0 | 0 | 0 | 0 | 1 | 0 | 1 |
| $>7$ | 1 | 1 | 0 | 0 | 1 | 0 | 1 |
| $>8$ | 1 | 1 | 0 | 0 | 1 | 0 | 1 |
| $>9$ | 1 | 1 | 0 | 0 | 1 | 0 | 1 |
| $>10$ | 1 | 1 | 0 | 0 | 1 | 0 | 1 |
| $>11$ | 1 | 1 | 0 | 0 | 1 | 0 | 1 |

> binhalfwidth_raw sizebin_min sizebin_max sizebin_center sizebin_width
> 1
$>2$
0.0
$0.0 \quad 0.0$
$0.0 \quad 0.0$

| 0.0 | 0 |
| :--- | :--- |
| 0.0 | 0 |
| 0.0 | 0 |
| 0.0 | 0 |
| 0.0 | 0 |



```
> $labels
> pop patch year2
> 1 1 A 2004
>2 1 A 2005
> 3 1 A 2006
>4 1 A 2007
> 5 1 A 2008
> 6 1 B 2004
> 7 1 B 
> 8 1 B 2006
> 9 1 B 2007
> 10 1 B 2008
>11 1 C 2004
> 12 1 C 2005
>13 1 C 2006
>14 1 C 2007
> 15 1 C 2008
>
> $matrixqc
> [1] 266 70 15
>
> $dataqc
> [1] 74 320
>
> attr(,"class")
> [1] "lefkoMat"
```

The output from this analysis is a lefkoMat object, which is described in detail in section 1.7.1. In brief, this object is a list with the following elements:

1. A: a list of full population projection matrices, in order of population, patch, and occasion in time $t$.
2. U: a list of matrices showing only survival-transition elements, in the same order as A.
3. $\mathbf{F}$ : a list of matrices showing only fecundity elements, in the same order as A .
4. hstages: a data frame showing the order of paired stages (given if matrices are historical, otherwise NA).
5. agestages: a data frame showing the order of age-stages (if matrices are age-by-stage, otherwise NA).
6. ahstages: the stageframe used in analysis, with stages reordered and edited as they occur in the matrix.
7. labels: a table showing the order of matrices, according to population, patch, and occasion in time $t$.
8. matrixqc: a short vector used in summary statements to describe the overall quality of each matrix.
9. dataqc: a short vector used in summary statements to describe key sampling aspects of the dataset (only in raw MPMs).
10. modelqc: a short vector used in summary statements to describe the vital rate models (only in function-based MPMs).

Calling particular values and elements within lefkoMat objects is not complicated, once you know the structure. Figure 4.2 illustrates how to call a particular element from one of the A matrices, for example.


Figure 4.2: Organization of a lefkoMat object, and how to call a specific element

Objects of class lefkoMat have their own summary () statements, which we can use to understand more about them.

```
summary(cypmatrix2rp)
>
> This ahistorical lefkoMat object contains 15 matrices.
> Each matrix is square with 11 rows and columns, and a total of 121 elements.
> A total of 266 survival transitions were estimated, with 17.733 per matrix.
> A total of 70 fecundity transitions were estimated, with 4.667 per matrix.
> This lefkoMat object covers 1 population, 3 patches, and 5 time steps.
> The dataset contains a total of }74\mathrm{ unique individuals and 320 unique transitions.
>
> Survival probability sum check (each matrix represented by column in order):
\(>\quad[, 1][, 2][, 3][, 4][, 5][, 6][, 7][, 8]\) [,9] [,10] [,11] [,12]
> Min. 0.000 0.000 0.000 0.000 0.000 0.000 0.050 0.050 0.000 0.050 0.000 0.000
> 1st Qu. 0.075 0.025 0.075 0.025 0.075 0.075 0.140 0.140 0.100 0.140 0.100 0.100
> Median 0.180 0.100 0.180 0.100 0.180 0.180 0.909 0.778 0.686 0.857 0.750 0.575
> Mean 0.457 0.361 0.471 0.328 0.417 0.464 0.631 0.611 0.530}00.631 0.562 0.523
> 3rd Qu. 0.955 0.769 1.000 0.592 0.781 1.000 1.000 1.000 0.955 1.000 1.000 1.000
> Max. 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000
> [,13] [,14] [,15]
> Min. 0.000 0.000 0.000
> 1st Qu. 0.075 0.075 0.100
> Median 0.180 0.180 0.750
```

```
> Mean 0.432 0.450 0.562
> 3rd Qu. 0.875 1.000 1.000
> Max. 1.000 1.000 1.000
```

We start off learning that 15 matrices were estimated, and we learn the dimensionality of those matrices. Next we see how many elements were actually estimated, both overall and per matrix (actually the number of non-zero elements), and the mean number of non-zero survival transitions and fecundity terms per matrix. This is followed by the number of populations, patches or subpopulations, and time steps covered by the MPM. Then we see the number of individuals and transitions the matrices are based on. It is typical for population ecologists to consider the total number of transitions in a dataset as a measure of the statistical power of a matrix, but the number of individuals used is just as important because each transition that an individual experiences is dependent on the other transitions that it also experiences. The final portion of the summary shows us the range of survival probabilities of stages in the matrices, where the survival probabilities are calculated as column sums of each $U$ matrix. Since there are 15 matrices, there are 15 survival summaries. It is important to check to see that no stage survives outside the realm of possibility (i.e. no probability should be greater than 1.0 or lower than 0.0 ). Unusual stage survival probabilities will result in a warning as a part of the summary() output.

The input for the rlefko2() function includes patch = "all" and year = "all", but can be set to focus on any set of patches / subpopulations or years included within the data. Package lefko3 includes a great deal of flexibility here, and can estimate many matrices covering all of the populations, patches, and years occurring in a specific dataset. For example, if we had wished to skip the patch divisions within the population and instead estimate only annual matrices at the population level, then we could have eliminated the patch and patchcol options altogether from the input, as below.

```
cypmatrix2r <- rlefko2(data = cypraw_v1, stageframe = cypframe_raw,
    year = "all", stages = c("stage3", "stage2"),
    size = c("size3added", "size2added"), supplement = cypsupp2_raw,
    yearcol = "year2", indivcol = "individ")
summary(cypmatrix2r)
>
> This ahistorical lefkoMat object contains 5 matrices.
>
> Each matrix is square with }11\mathrm{ rows and columns, and a total of 121 elements.
> A total of 120 survival transitions were estimated, with 24 per matrix.
> A total of 40 fecundity transitions were estimated, with }8\mathrm{ per matrix.
> This lefkoMat object covers 1 population, 1 patch, and 5 time steps.
>
> The dataset contains a total of 74 unique individuals and 320 unique transitions.
>
> Survival probability sum check (each matrix represented by column in order):
> [,1] [,2] [,3] [,4] [,5]
> Min. 0.000 0.050 0.050 0.000 0.050
> 1st Qu. 0.100 0.140 0.140 0.100 0.140
> Median 0.689 0.870 0.864 0.610 0.882
> Mean 0.552 0.629 0.629 0.528 0.627
> 3rd Qu. 1.000 1.000 1.000 0.960 1.000
> Max. 1.000 1.000 1.000 1.000 1.000
```

Notice the difference in the output. The first call to rlefko2() yielded 15 matrices, because there are three patches and a total of six years of data, yielding five transitions between years (also referred
to as time steps or periods). So, there are $3 \times 5=15$ matrices. But in the second call, we no longer recognize patches and so have only estimated one set of five matrices covering the whole population. A further impact of this change is that the mean number of elements estimated per matrix increased, because there are fewer missing transitions in the dataset per matrix when the data are not parsed by patch. We can also create MPMs for specific patches and specific sets of years, setting the options appropriately.

Let's try another exercise in this vein. This time, we will estimate matrices at the patch level, but only for the years 2006 and 2007. This will result in six matrices, since there are three patches and two monitoring occasions (note that focusing on years 2006 and 2007 means that we will only build matrices in which these are the years in time $t$ ).

```
cypmatrix2rp67 <- rlefko2(data = cypraw_v1, stageframe = cypframe_raw,
    year = c(2006, 2007), patch = "all", stages = c("stage3", "stage2"),
    size = c("size3added", "size2added"), supplement = cypsupp2_raw,
    yearcol = "year2", patchcol = "patchid", indivcol = "individ")
summary(cypmatrix2rp67)
>
> This ahistorical lefkoMat object contains 6 matrices.
>
> Each matrix is square with 11 rows and columns, and a total of 121 elements.
> A total of 111 survival transitions were estimated, with 18.5 per matrix.
> A total of 22 fecundity transitions were estimated, with 3.667 per matrix.
> This lefkoMat object covers 1 population, 3 patches, and 2 time steps.
>
> The dataset contains a total of 74 unique individuals and 320 unique transitions.
>
> Survival probability sum check (each matrix represented by column in order):
> [,1] [,2] [,3] [,4] [,5] [,6]
>Min. 0.000 0.000 0.050 0.000 0.000 0.000
> 1st Qu. 0.075 0.025 0.140 0.100 0.075 0.075
> Median 0.180 0.100 0.778 0.686 0.180 0.180
> Mean 0.471 0.328 0.611 0.530 0.432 0.450
> 3rd Qu. 1.000 0.592 1.000 0.955 0.875 1.000
> Max. 1.000 1.000 1.000 1.000 1.000 1.000
```

Finally, we will build matrices only for the years 2006 and 2007 in patch A. We expect only two matrices.

```
cypmatrix2rA67 <- rlefko2(data = cypraw_v1, stageframe = cypframe_raw,
    year = c(2006, 2007), patch = "A", stages = c("stage3", "stage2"),
    size = c("size3added", "size2added"), supplement = cypsupp2_raw,
    yearcol = "year2", patchcol = "patchid", indivcol = "individ")
summary(cypmatrix2rA67)
>
> This ahistorical lefkoMat object contains 2 matrices.
>
> Each matrix is square with 11 rows and columns, and a total of 121 elements.
> A total of 30 survival transitions were estimated, with }15\mathrm{ per matrix.
> A total of 0 fecundity transitions were estimated, with 0 per matrix.
> This lefkoMat object covers 1 population, 1 patch, and 2 time steps.
```

```
>
> The dataset contains a total of }74\mathrm{ unique individuals and 320 unique transitions.
>
> Survival probability sum check (each matrix represented by column in order):
> [,1] [,2]
> Min. 0.000 0.000
> 1st Qu. 0.075 0.025
> Median 0.180 0.100
> Mean 0.471 0.328
> 3rd Qu. 1.000 0.592
> Max. 1.000 1.000
```

This output is also interesting because it showcases one of the weaknesses of the raw MPM approach. We see that no individuals produced any offspring in patch A over these two time steps, so no non-zero fecundity terms were estimated. While this is a very accurate result, it can lead to problems if we think of the MPM as reflecting the underlying biology of the species. In this case, fecundity is missing most likely because there were too few individuals in this patch at these times and none of those present produced flowers. So, the luck of the draw left us with some biologically estimable elements not being estimated.

### 4.2 Ahistorical vs. historical matrix models

The matrices that we have built so far are all examples of raw ahistorical MPMs (ahMPMs). These are projection models in which the future stage of an individual is dependent only on its current stage, and not on previous stages. In other words, individual history is not incorporated into ahMPMs. It may seem odd that individual history is not incorporated into the matrices given that demographic datasets are composed of records of individual histories spanning several or even many monitoring occasions. However, construction of an ahMPM breaks up these individual histories into pairs of consecutive stages across time, with each stage-pair treated as independent of every other pair of consecutive stages. For example, if an individual is in stage $A$ in occasion 1 , stage $B$ in occasion 2 , stage $E$ in occasion 3 , and dead in occasion 4 , then its individual history is broken up into $A-B, B-E$, and $E$-Dead, with each stage pair assumed to be independent. The resulting projection matrix would be the same if these transitions originated from different individuals or the same one - their order and relationships do not have any impacts in ahMPM construction.

The independence of consecutive stage-pairs reflects a central assumption in ahMPM analysis: the stage of an individual in the next occasion is influenced only by its current stage. Conceptually, if an organism's stage in the next occasion is entirely determined by its current stage, then its previous states do not influence these transitions. Thus, standard ahMPMs are two dimensional and reflect only the current and next immediate stage of individuals, given by the columns and rows, respectively. This is ultimately an extension of the iad assumption in statistics - that the states of individuals are independent and originate from identically-distributed random variables. This assumption helped make matrix projection models relatively easy to estimate even before the advent of personal computers.

The first MPMs were produced in the 1940s, with Leslie's introduction of the age-structured model (Caswell, 2001; Leslie, 1945). We have never done a meta-analysis of all of these studies, but nonetheless it is safe to say that studies considering individual history are rare. The typical MPM study involves only ahMPMs, and so assumes independence of stage transitions across time even from the same individual. In fact, at the time of writing, we are aware of only five examples of studies breaking these assumptions and using a historical approach, in which some degree of individual history is incorporated into matrix estimation and analysis (Ehrlén, 2000; Shefferson et al., 2014, 2017, 2018; deVries and Caswell, 2018).

The historical MPM (hMPM) is an extension of the matrix projection model that incorporates
information on one previous occasion into the determination of vital rates. Thus, the expected survivaltransition probability of an individual in stage $j$ at occasion $t$ to stage $k$ at occasion $t+1$ depends not only on its stage in occasion $t$ but also on its stage in occasion $t-1$. Population ecologists considering this problem analytically might be inclined to add an extra dimension to the matrix to deal with this, thus creating a 3d array or cube. However, this is mathematically intractable, with many analyses becoming impossible (deVries and Caswell, 2018). Instead, we utilize the approach developed by Ehrlén (2000), in which rows and columns represent life history stages paired in consecutive occasions. Thus, columns now represent the From pair of stages (stages in occasions $t-1$ and $t$ ), and rows now represent the To pair of stages (stages in occasions $t$ and $t+1$ ), as in figure 4.3. This model is equivalent to the second-order model proposed by deVries and Caswell (2018), although the latter parameterizes history in transitions involving individuals just born differently (we explain the difference further in this vignette).

|  | $\begin{aligned} & \mathrm{S}_{\mathrm{t}-1} \\ & \mathrm{~S}_{\mathrm{t}} \end{aligned}$ | $\begin{gathered} \mathbf{S}_{t-1} \\ \mathbf{J}_{t} \end{gathered}$ | $\mathrm{J}_{t-1}$ | $\begin{aligned} & \mathrm{J}_{t-1} \\ & \mathrm{~V}_{t} \end{aligned}$ | $\begin{gathered} \mathrm{J}_{t-1} \\ \mathrm{D}_{\mathrm{t}} \end{gathered}$ | $\begin{gathered} \mathbf{J}_{t-1} \\ F_{t} \end{gathered}$ | $\begin{aligned} & v_{t-1} \\ & v_{t} \end{aligned}$ | $\begin{gathered} V_{t-1} \\ D_{t} \end{gathered}$ | $\begin{gathered} \mathbf{V}_{\mathrm{t}-1} \\ \mathrm{~F}_{\mathrm{t}} \end{gathered}$ | $\begin{aligned} & \mathrm{D}_{t-1} \\ & \mathrm{~V}_{t} \end{aligned}$ | $\begin{gathered} \mathrm{D}_{t-1} \\ \mathrm{D}_{\mathrm{t}} \end{gathered}$ | $\begin{gathered} D_{t-1} \\ F_{t} \end{gathered}$ | $\begin{aligned} & \mathbf{F}_{t-1} \\ & \mathbf{V}_{t} \end{aligned}$ | $\begin{gathered} \mathbf{F}_{t-1} \\ \mathbf{D}_{t} \end{gathered}$ | $\begin{gathered} F_{t-1} \\ F_{t} \end{gathered}$ |  | $\begin{aligned} & \mathbf{F}_{t-1} \\ & \mathbf{J}_{t} \end{aligned}$ |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| $\mathrm{S}_{t+1} \mathrm{~S}_{\mathrm{t}}$ | $\mathrm{S}_{\text {sss }}$ | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | $\mathrm{S}_{\text {SSF }}$ | 0 |
| $J_{t+1} S_{t}$ | $\mathrm{S}_{\text {SS }}$ | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | $\mathrm{S}_{\mathrm{JSF}}$ | 0 |
| $\mathrm{J}_{t+1} \mathrm{~J}_{t}$ | 0 | $\mathrm{S}_{\mathrm{JS}}$ | $\mathrm{S}_{\mathrm{\# J}}$ | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | $\mathrm{S}_{\mathrm{JJF}}$ |
| $\mathrm{V}_{t+1} \mathrm{~J}_{t}$ | 0 | $\mathrm{S}_{\mathrm{VJS}}$ | $\mathrm{S}_{\mathrm{VIJ}}$ | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | $\mathrm{S}_{\mathrm{VJF}}$ |
| $\mathrm{D}_{t+1} \mathrm{~J}_{t}$ | 0 | $\mathrm{S}_{\text {JJS }}$ | $\mathrm{S}_{\text {DJ }}$ | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | $\mathrm{S}_{\mathrm{DJF}}$ |
| $\mathrm{F}_{t+1} \mathrm{~J}_{t}$ | 0 | $\mathrm{S}_{\mathrm{FJS}}$ | $\mathrm{S}_{\mathrm{FJJ}}$ | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | $\mathrm{S}_{\mathrm{FJF}}$ |
| $\mathrm{V}_{t+1} \mathrm{~V}_{t}$ | 0 | 0 | 0 | $\mathrm{S}_{\mathrm{wvj}}$ | 0 | 0 | $\mathrm{S}_{\mathrm{vvv}}$ | 0 | 0 | $\mathrm{S}_{\mathrm{vvD}}$ | 0 | 0 | $\mathrm{S}_{\mathrm{VVF}}$ | 0 | 0 | 0 | 0 |
| $\mathrm{D}_{t+1} \mathrm{~V}_{t}$ | 0 | 0 | 0 | $\mathrm{S}_{\text {DVJ }}$ | 0 | 0 | $\mathrm{S}_{\text {DW }}$ | 0 | 0 | $\mathrm{S}_{\text {DVD }}$ | 0 | 0 | $\mathrm{S}_{\text {DVF }}$ | 0 | 0 | 0 | 0 |
| $\mathrm{F}_{t+1} \mathrm{~V}_{\mathrm{t}}$ | 0 | 0 | 0 | $\mathrm{S}_{\mathrm{FV} \text { J }}$ | 0 | 0 | $\mathrm{S}_{\mathrm{FWV}}$ | 0 | 0 | $\mathrm{S}_{\mathrm{FVD}}$ | 0 | 0 | $\mathrm{S}_{\mathrm{FVF}}$ | 0 | 0 | 0 | 0 |
| $\mathrm{V}_{t+1} \mathrm{D}_{t}$ | 0 | 0 | 0 | 0 | SVoj | 0 | 0 | Svov | 0 | 0 | SVDD | 0 | 0 | $\mathrm{S}_{\mathrm{VDF}}$ | 0 | 0 | 0 |
| $\mathrm{D}_{t+1} \mathrm{D}_{t}$ | 0 | 0 | 0 | 0 | $\mathrm{S}_{\text {DJJ }}$ | 0 | 0 | $\mathrm{S}_{\text {DDV }}$ | 0 | 0 | $S_{\text {DDD }}$ | 0 | 0 | $\mathrm{S}_{\text {DDF }}$ | 0 | 0 | 0 |
| $F_{t+1} \mathrm{D}_{\text {t }}$ | 0 | 0 | 0 | 0 | $\mathrm{S}_{\text {FJJ }}$ | 0 | 0 | $\mathrm{S}_{\mathrm{FDV}}$ | 0 | 0 | $\mathrm{S}_{\mathrm{FDD}}$ | 0 | 0 | $\mathrm{S}_{\mathrm{FDF}}$ | 0 | 0 | 0 |
| $v_{t+1} F_{t}$ | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | $\mathrm{S}_{\mathrm{VFV}}$ | 0 | 0 | SVFD | 0 | 0 | $\mathrm{S}_{\mathrm{VFF}}$ | 0 | 0 |
| $\mathrm{D}_{t+1} \mathrm{~F}_{\mathrm{t}}$ | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | $\mathrm{S}_{\text {DFV }}$ | 0 | 0 | $\mathrm{S}_{\text {DFD }}$ | 0 | 0 | $\mathrm{S}_{\text {DFF }}$ | 0 | 0 |
| $F_{t+1} F_{t}$ | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | $\mathrm{S}_{\mathrm{FFV}}$ | 0 | 0 | $\mathrm{S}_{\text {FFD }}$ | 0 | 0 | $\mathrm{S}_{\mathrm{FFF}}$ | 0 | 0 |
| $S_{t+1} F_{t}$ | 0 | 0 | 0 | 0 | 0 | $\mathrm{F}_{\text {SFJ }}$ | 0 | 0 | $\mathrm{F}_{\text {SFV }}$ | 0 | 0 | $\mathrm{F}_{\text {SFD }}$ | 0 | 0 | $\mathrm{F}_{\text {SFF }}$ | 0 | 0 |
| $J_{t+1} F_{t}$ | 0 | 0 | 0 | 0 | 0 | $\mathrm{F}_{\mathrm{JFJ}}$ | 0 | 0 | $\mathrm{F}_{\mathrm{JFV}}$ | 0 | 0 | $\mathrm{F}_{\mathrm{JFD}}$ | 0 | 0 | $\mathrm{F}_{\text {JFF }}$ | 0 | 0 |

Figure 4.3: Historical MPM for Cypripedium candidum, a North American herbaceous plant

Historical MPMs can be projected forward in the same way that ahMPMs can. In figure 4.4 we see an example of a projection one time step forward using the hMPM shown in figure 4.3.

Here, $n_{1, S S}$ is the number of individuals that were in stage $S$ in both occasions 0 and 1 , and $n_{2, D F}$ is the number of individuals that were in stage $D$ in occasion 2 and stage $F$ in occasion 1.


Figure 4.4: Historical MPM projection in Ehrlén format

Ahistorical matrices and historical matrices can be derived from the same datasets provided that individual fates are noted for three consecutive times each. We can refer to matrix elements as $a_{k j l}$, where the element represents the rate at which individuals transition to stage $k$ in occasion $t+1$ after having been in stage $j$ in occasion $t$ and in stage $l$ in occasion $t$-1. If $n_{k j l}$ is the number of individuals making this transition, $n_{. j l}$ is the total number of individuals in stage $j$ in occasion $t$ and stage $l$ in occasion $t-1$ regardless of stage or even status as alive or dead in occasion $t+1, m$ is the number of stages in the life history model, and $d$ is the number of stages plus death in the life history model (so that $d=m+1$ ), then we have:

$$
\begin{equation*}
a_{k j l}=\frac{n_{k j l}}{n_{. j l}}=\frac{n_{k j l}}{\sum_{i=1}^{d} n_{i j l}} \tag{4.4}
\end{equation*}
$$

Note that although we can use the actual numbers of individuals making historical transitions to estimate ahistorical MPMs, we cannot use the matrix elements in a historical MPM to calculate the associated ahistorical MPM elements, nor can we use the elements in an ahistorical MPM to calculate the elements in a historical MPM. This is due to the fact that historical transitions must be weighted by the numbers of individuals in each previous stage in occasions $t$ and $t-1\left(n_{. j l}\right)$ to produce the proper ahistorical transition rates. Historical matrix elements contain no information about these weights. Thus, we have the following relationships between ahistorical survival-transition terms and the historical datasets:

$$
\begin{equation*}
a_{k j}=\frac{\sum_{l=1}^{m} n_{k j l}}{\sum_{l=1}^{m} \sum_{i=1}^{d} n_{i j l}} \tag{4.5}
\end{equation*}
$$

Historical MPMs normally require a much larger number of elements to be parameterized than ahMPMs do. Figure 4.3 illustrates this issue. Typically, a historical matrix has dimensions equal to the number of stages squared (although this may sometimes be reduced, if certain conditions are met). However, most of the increase in matrix elements is in structural zeros. In an ahistorical matrix, the only true zeros will be those elements corresponding to biologically impossible transitions, or those elements without any individuals taking the respective transition in a given time step. However, most elements in a historical MPM are structural zeros, because transition elements in hMPMs are only estimable if stage at occasion $t$ is equal in the column and row stage pairs. For example, the transition probability between stage $A$ in occasion $t-1$ and stage $B$ in occasion $t$ (column stage-pair), to stage $C$ in occasion $t$ and stage $D$ in occasion $t+1$ (row stage-pair) equals 0 , because an organism cannot be in both stages $B$ and $C$ in occasion $t$. Increasing the number of stages in the life history model causes these structural zeros to increase at a faster rate than the rate at which the number of truly estimable elements increases. In fact, if there are $m$ stages in a life history, yielding $m^{2}$ elements in an ahistorical matrix, then although there will be $m^{4}$ elements in the historical matrix, only $m^{3}$ will be potentially estimable while $(m-1) m^{3}$ will be structural zeros (we say potentially because some of the
logically possible transitions may still be biologically impossible, or may have no data to yield values other than 0 ).

### 4.2.1 Historical MPMs in lefko3

Let's build our first historical MPM with the Cypripedium candidum data. As before, we will build our MPM to cover all years and all patches.

```
cypmatrix3rp <- rlefko3(data = cypraw_v1, stageframe = cypframe_raw,
    year = "all", patch = "all", stages = c("stage3", "stage2", "stage1"),
    size = c("size3added", "size2added", "size1added"), supplement = cypsupp3_raw,
    yearcol = "year2", patchcol = "patchid", indivcol = "individ")
summary(cypmatrix3rp)
>
> This historical lefkoMat object contains 12 matrices.
>
> Each matrix is square with 121 rows and columns, and a total of 14641 elements.
A total of 516 survival transitions were estimated, with 43 per matrix.
A total of }70\mathrm{ fecundity transitions were estimated, with 5.833 per matrix.
This lefkoMat object covers 1 population, 3 patches, and 4 time steps.
>
The dataset contains a total of 74 unique individuals and 320 unique transitions.
Survival probability sum check (each matrix represented by column in order):
> [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11]
Min. 0.000 0.0000 0.0000 0.000 0.000 0.000 0.00 0.000 0.000 0.0000 0.000
1st Qu. 0.000 0.0000 0.0000 0.000 0.000 0.000 0.00 0.000 0.000 0.0000 0.000
Median 0.000 0.0000 0.0000 0.000 0.000 0.000 0.00 0.000 0.000 0.0000 0.000
Mean 0.107 0.0945 0.0851 0.101 0.158 0.158 0.14 0.169 0.119 0.0851 0.119
3rd Qu. 0.000 0.0000 0.0000 0.000 0.100 0.100 0.05 0.100 0.000 0.0000 0.000
Max. 1.000 1.0000 1.0000 1.000 1.000 1.000 1.00 1.000 1.000 1.0000 1.000
[,12]
Min. 0.000
1st Qu. 0.000
Median 0.000
Mean 0.144
3rd Qu. 0.000
Max. 1.000
```

Quickly scanning this output shows a number of differences relative to the ahMPM. First, there are three fewer matrices here than in the ahistorical case. There are three patches that we are estimating matrices for, and six years of data for each patch, leading to five possible ahistorical time steps and 15 possible ahistorical matrices. Since historical matrices require three years of transition data, only four historical transitions are possible per patch, leading to 12 total historical matrices. Second, the size of the matrices in terms of the number of rows and columns is the square of the dimensions of the ahistorical matrices. This leads to vastly more matrix elements within each matrix, although it turns out that most of these matrix elements are structural zeros because they reflect impossible transitions. Indeed, in this case, although there are 14,641 elements in each matrix, on average only 48.83 are actually estimated to values greater than 0 .

Let's look at the first matrix, corresponding to the transition from 2004 and 2005 to 2006 in the first patch (patch A, according to the labels element). Because this is a huge matrix, we will only
look at the top corner, followed by a section toward the middle. Particularly note the sparseness most elements are zeros, because most transitions are actually impossible. These matrices may also be exported to Microsoft Excel, Apple Numbers, or another spreadsheet program to look over in greater detail.


As before, we can create matrices at the full population level by withholding the patch and patchcol options, as below.

```
cypmatrix3r <- rlefko3(data = cypraw_v1, stageframe = cypframe_raw,
    year = "all", stages = c("stage3", "stage2", "stage1"),
    size = c("size3added", "size2added", "size1added"), supplement = cypsupp3_raw,
    yearcol = "year2", indivcol = "individ")
summary(cypmatrix3r)
>
> This historical lefkoMat object contains 4 matrices.
>
> Each matrix is square with }121\mathrm{ rows and columns, and a total of 14641 elements.
> A total of 242 survival transitions were estimated, with 60.5 per matrix.
> A total of 54 fecundity transitions were estimated, with 13.5 per matrix.
> This lefkoMat object covers 1 population, 1 patch, and 4 time steps.
>
> The dataset contains a total of 74 unique individuals and 320 unique transitions.
>
> Survival probability sum check (each matrix represented by column in order):
> [,1] [,2] [,3] [,4]
Min. 0.000 0.000 0.000 0.000
1st Qu. 0.000 0.000 0.000 0.000
Median 0.000 0.000 0.000 0.000
Mean 0.173 0.179 0.166 0.198
3rd Qu. 0.100 0.100 0.100 0.100
Max. 1.000 1.000 1.000 1.000
```

We now have only four matrices, corresponding to the full population in each year at time $t$. Next, let's focus only on years 2006 and 2007 in all patches, as before.

```
cypmatrix3rp67 <- rlefko3(data = cypraw_v1, stageframe = cypframe_raw,
    year = c(2006, 2007), patch = "all", stages = c("stage3", "stage2", "stage1"),
    size = c("size3added", "size2added", "size1added"), supplement = cypsupp3_raw,
    yearcol = "year2", patchcol = "patchid", indivcol = "individ")
summary(cypmatrix3rp67)
>
> This historical lefkoMat object contains 6 matrices.
>
> Each matrix is square with }121\mathrm{ rows and columns, and a total of 14641 elements.
> A total of 260 survival transitions were estimated, with 43.333 per matrix.
> A total of 30 fecundity transitions were estimated, with 5 per matrix.
> This lefkoMat object covers 1 population, 3 patches, and 2 time steps.
>
> The dataset contains a total of 74 unique individuals and 320 unique transitions.
>
> Survival probability sum check (each matrix represented by column in order):
> [,1] [,2] [,3] [,4] [,5] [,6]
Min. 0.0000 0.0000 0.000 0.00 0.0000 0.000
1st Qu. 0.0000 0.0000 0.000 0.00 0.0000 0.000
Median 0.0000 0.0000 0.000 0.00 0.0000 0.000
Mean 0.0945 0.0851 0.158 0.14 0.0851 0.119
3rd Qu. 0.0000 0.0000 0.100 0.05 0.0000 0.000
Max. 1.0000 1.0000 1.000 1.00 1.0000 1.000
```

We find six matrices, corresponding to the years 2006 and 2007 for each of the three patches. And now let's focus in on years 2006 and 2007 in the first patch, patch A.

```
cypmatrix3rA67 <- rlefko3(data = cypraw_v1, stageframe = cypframe_raw,
    year = c(2006, 2007), patch = "A", stages = c("stage3", "stage2", "stage1"),
    size = c("size3added", "size2added", "size1added"), supplement = cypsupp3_raw,
    yearcol = "year2", patchcol = "patchid", indivcol = "individ")
summary(cypmatrix3rA67)
>
> This historical lefkoMat object contains 2 matrices.
>
> Each matrix is square with }121\mathrm{ rows and columns, and a total of 14641 elements.
> A total of 78 survival transitions were estimated, with 39 per matrix.
> A total of O fecundity transitions were estimated, with O per matrix.
> This lefkoMat object covers 1 population, 1 patch, and 2 time steps.
>
> The dataset contains a total of 74 unique individuals and 320 unique transitions.
>
> Survival probability sum check (each matrix represented by column in order):
> [,1] [,2]
> Min. 0.0000 0.0000
> 1st Qu. 0.0000 0.0000
> Median 0.0000 0.0000
> Mean 0.0945 0.0851
> 3rd Qu. 0.0000 0.0000
> Max. 1.0000 1.0000
```

As before, we find no fecundity terms estimated, as no individuals in patch A were reproductive in these two times.

### 4.3 Alternate parameterizations of historical MPMs

deVries and Caswell (2018) detail different approaches to the development of hMPMs, differing in how state or stage in occasion $t-1$ is incorporated into the matrix. Full prior stage dependence models deal with history by incorporating prior condition as the exact stage of an organism in occasion $t-1$, yielding a 2 d matrix showing stage pairs in occasion $t-1$ and $t$ along the columns of the matrix, and stage pairs in occasion $t$ and $t+1$ along the rows of the matrix. Prior condition models deal with history by making the transition from stage $t$ to stage $t+1$ a function of stage at occasion $t$ and condition in occasion $t-1$. Prior condition can be determined in the same way that current stage is determined (e.g. size classification), or a different measure of condition can be used, such as growth (i.e. the change in size between occasions $t-1$ and $t$ ).

One key feature proposed by deVries and Caswell (2018) is the addition of a stage to account for the prior status of newborn individuals. This reflects a different interpretation from Ehrlén Ehrlén (2000) of the historical transition. In Ehrlén (2000), all matrix elements simply reflect the order of the events in a single transition, including both fecundity and survival-transition events. However, in deVries and Caswell (2018), these matrix elements must also reflect the history of specific individuals. In the latter case, the fact that a newborn in occasion $t$ did not exist in occasion $t-1$ requires that a new prior stage is constructed and used in the matrix to account for that individual's prior lack of existence. This new stage only exists in the prior occasion, and is only used for newborns in the first survival transition from birth. So, for example, in a historical MPM where the first stage is the
newborn stage and only the third stage is reproductive, we add a fourth stage to the prior portion of the row and column. Thus, we may start with the following hMPM in Ehrlén (2000) format:


Figure 4.5: Example historical MPM projection in Ehrlén format

This hMPM becomes as follows in deVries and Caswell (2018) format, where the prior "pre-born" stage is shown as $P$.


Figure 4.6: Example historical MPM projection in deVries format

Package lefko3 generally implements the full prior stage dependence model approach, particularly in the development of raw matrices. However, in principle, function-based matrices developed with lefko3 can be seen as falling within the prior condition model approach where prior condition is determined in the same way as current stage. We also implement both the Ehrlén (2000) format and the deVries and Caswell (2018) format, and use the former as the default. We leave it to the user to decide which approach to use, and point out only that the literature suggests that maternal condition can have long-term effects on offspring survival, for example through maternal care and epigenetic influences (Descamps et al., 2008; Beamonte-Barrientos et al., 2010; Dogra and Dani, 2019). Thus, retaining maternal state as the prior condition of a newborn makes some logical sense and has a basis in the literature.

Let's build a de Vries format hMPM for the Cypripedium candidum dataset.

```
cypmatrix3rp_dev <- rlefko3(data = cypraw_v1, stageframe = cypframe_raw,
    year = "all", patch = "all", stages = c("stage3", "stage2", "stage1"),
    size = c("size3added", "size2added", "size1added"), supplement = cypsupp3_raw,
    yearcol = "year2", patchcol = "patchid", indivcol = "individ",
    format = "deVries")
summary(cypmatrix3rp_dev)
>
> This historical lefkoMat object contains }12\mathrm{ matrices.
>
> Each matrix is square with 132 rows and columns, and a total of 17424 elements.
> A total of }516\mathrm{ survival transitions were estimated, with 43 per matrix.
> A total of }70\mathrm{ fecundity transitions were estimated, with 5.833 per matrix.
> This lefkoMat object covers 1 population, 3 patches, and 4 time steps.
```

```
>
> The dataset contains a total of 74 unique individuals and 320 unique transitions.
>
> Survival probability sum check (each matrix represented by column in order):
\(>\quad[, 1] \quad[, 2][, 3][, 4][, 5][, 6][, 7]\) [,8] [,9] [,10] [,11]
Min. 0.0000 0.0000 0.000 0.0000 0.000 0.0000 0.000 0.0000 0.000 0.000 0.000
1st Qu. 0.0000 0.0000 0.000 0.0000 0.000 0.0000 0.000 0.0000 0.000 0.000 0.000
Median 0.0000 0.0000 0.000 0.0000 0.000 0.0000 0.000 0.0000 0.000 0.000 0.000
Mean 0.0981 0.0866 0.078 0.0923 0.144 0.1445 0.128 0.1548 0.109 0.078 0.109
3rd Qu. 0.0000 0.0000 0.000 0.0000 0.000 0.0125 0.000 0.0125 0.000 0.000 0.000
Max. 1.0000 1.0000 1.000 1.0000 1.000 1.0000 1.000 1.0000 1.000 1.000 1.000
> [,12]
> Min. 0.000
> 1st Qu. 0.000
Median 0.000
> Mean 0.132
3rd Qu. 0.000
Max. 1.000
```

Our new matrices have 11 more rows and columns, and 2,783 more elements. To see what is different, compare the hstages element in each on your own, and you will see the inclusion of some new stage pairs with the AlmostBorn stage. It is also possible for some transitions to be split, occasionally yielding more transitions than were estimated in the Ehrlén format case.

### 4.4 Sparse matrix format

Historical matrices can be huge. Unfortunately, the larger a matrix is, the longer it takes to create and to analyze, and the greater the share of memory needed to deal with it. In fact, some MPMs can take up many gigabytes of memory, and such situations risk not only extending analysis time into unreasonable territory, but also analysis termination due to memory shortage. To help deal with these situations, all matrix creation functions in package lefko3 can produce and analyze MPMs in sparse matrix format. As an example, let's create a sparse version of the deVries-formatted historical MPM.

```
cypmatrix3rp_dev_sparse <- rlefko3(data = cypraw_v1, stageframe = cypframe_raw,
    year = "all", patch = "all", stages = c("stage3", "stage2", "stage1"),
    size = c("size3added", "size2added", "size1added"), supplement = cypsupp3_raw,
    yearcol = "year2", patchcol = "patchid", indivcol = "individ",
    format = "deVries", sparse_output = TRUE)
summary(cypmatrix3rp_dev_sparse)
>
> This historical lefkoMat object contains 12 matrices.
>
> Each matrix is square with 132 rows and columns, and a total of 17424 elements.
A total of 516 survival transitions were estimated, with 43 per matrix.
A total of 70 fecundity transitions were estimated, with 5.833 per matrix.
This lefkoMat object covers 1 population, 3 patches, and 4 time steps.
>
The dataset contains a total of 74 unique individuals and 320 unique transitions.
>
Survival probability sum check (each matrix represented by column in order):
```

| > | [,1] | [,2] | [,3] | [,4] | [,5] | [,6] | [,7] | [,8] | [,9] | [,10] | [,11] |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| > Min. | 0.0000 | 0.0000 | 0.000 | 0.0000 | 0.000 | 0.0000 | 0.000 | 0.0000 | 0.000 | 0.000 | 0.000 |
| > 1st Qu. | 0.0000 | 0.0000 | 0.000 | 0.0000 | 0.000 | 0.0000 | 0.000 | 0.0000 | 0.000 | 0.000 | 0.000 |
| > Median | 0.0000 | 0.0000 | 0.000 | 0.0000 | 0.000 | 0.0000 | 0.000 | 0.0000 | 0.000 | 0.000 | 0.000 |
| > Mean | 0.0981 | 0.0866 | 0.078 | 0.0923 | 0.144 | 0.1445 | 0.128 | 0.1548 | 0.109 | 0.078 | 0.109 |
| > 3rdQu. | 0.0000 | 0.0000 | 0.000 | 0.0000 | 0.000 | 0.0125 | 0.000 | 0.0125 | 0.000 | 0.000 | 0.000 |
| > Max. | 1.0000 | 1.0000 | 1.000 | 1.0000 | 1.000 | 1.0000 | 1.000 | 1.0000 | 1.000 | 1.000 | 1.000 |
| > | [,12] |  |  |  |  |  |  |  |  |  |  |
| > Min. | 0.000 |  |  |  |  |  |  |  |  |  |  |
| $>1$ st Qu. | 0.000 |  |  |  |  |  |  |  |  |  |  |
| > Median | 0.000 |  |  |  |  |  |  |  |  |  |  |
| > Mean | 0.132 |  |  |  |  |  |  |  |  |  |  |
| > 3rd Qu. | 0.000 |  |  |  |  |  |  |  |  |  |  |
| > Max. | 1.000 |  |  |  |  |  |  |  |  |  |  |

Looking over the output, it is clear that we have the same MPM. However, we see a difference in formatting when we actually zoom in to take a look at a specific matrix.

```
cypmatrix3rp_dev_sparse$A[[1]]
> 132 x 132 sparse Matrix of class "dgCMatrix"
>
> [1,] 0.08
> [2,] 0.10
> [3,]
> [4,]
> [5,]
> [6,]
[7,]
    [8,]
    [9,]
[10,]
[11,]
[12,]
> [13,]
> [14,] . 0.1
> [15,]
>
> ................................
> .......suppressing 99 columns and 102 rows in show(); maybe adjust 'options(max.print= *, width =
> ...............................
>
> [118,]
> [119,]
> [120,]
> [121,]
> [122,]
> [123,]
> [124,]
> [125,]
> [126,]
> [127,]
> [128,]
```

```
> [129,]
> [130,]
> [131,]
> [132,]
```

Users with recent versions of R Studio can see the amount of memory taken up by each of these MPMs. However, let's see how much each object is taking up using the pryr package, as below (please install package plyr if it is not installed).

```
pryr::object_size(cypmatrix3rp_dev)
> 5.04 MB
pryr::object_size(cypmatrix3rp_dev_sparse)
> 80.24 kB
```

The standard matrix format version of the above MPM takes up roughly 5 MB on the author's computer, while the sparse format MPM takes up only 80 kB . Clearly we have a serious savings in memory usage with the sparse format for historical MPMs.

The dgCMatrix format is described in detail in package Matrix, and is a reliable sparse matrix format that retains compatibility with many other programming languages. We generally advocate the use of sparse format whenever a historical MPM with more than 100 stages is used, as well as in age-by-stage matrices in which the number of rows and columns is greater than 100 each. However, sparse format should not be used for most ahistorical IPMs, since they are inherently dense and will take up more memory than the equivalent IPM in standard matrix format.

### 4.5 Matrix reduction

One important issue that arises in raw MPM and hMPM estimation is that some rows and columns may have only zeroes associated with them. This is particularly the case with raw hMPMs, since these are quite data-hungry MPMs. Sometimes all of the rows and columns associated with a particular stage pair are completely full of zeros, because individuals were not observed to take particular transitions related to those stage pairs. In these situations, matrices can be reduced by eliminating the stage-pair or stage altogether. The result is a set of smaller matrices, which is useful considering that historical MPMs are generally large and can take up a great deal of memory. We can tell R to develop reduced MPMs by using the reduce $=$ TRUE option in the matrix generating function that we are using. Note that this option is available in all matrix generating functions, even for those creating ahistorical MPMs, and that rows and columns will be reduced only if all matrices within the lefkoMat object being produced have empty rows and columns associated with a particular stage or stage-pair.

```
cypmatrix3rp_red <- rlefko3(data = cypraw_v1, stageframe = cypframe_raw,
    year = "all", patch = "all", stages = c("stage3", "stage2", "stage1"),
    size = c("size3added", "size2added", "size1added"), supplement = cypsupp3_raw,
    yearcol = "year2", patchcol = "patchid", indivcol = "individ",
    reduce = TRUE)
summary(cypmatrix3rp_red)
>
> This historical lefkoMat object contains 12 matrices.
>
> Each matrix is square with 43 rows and columns, and a total of 1849 elements.
> A total of 516 survival transitions were estimated, with 43 per matrix.
> A total of 70 fecundity transitions were estimated, with 5.833 per matrix.
```

```
> This lefkoMat object covers 1 population, 3 patches, and 4 time steps.
>
> The dataset contains a total of 74 unique individuals and 320 unique transitions.
>
> Survival probability sum check (each matrix represented by column in order):
> [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11] [,12]
> Min. 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000
> 1st Qu. 0.000 0.000 0.000 0.000 0.025 0.075 0.000 0.075 0.000 0.000 0.000 0.000
> Median 0.100 0.100 0.100 0.100 0.200 0.200 0.200 0.200 0.100 0.100 0.100 0.180
> Mean 0.301 0.266 0.239 0.283 0.443 0.443 0.393 0.475 0.336 0.239 0.336 0.405
> 3rd Qu. 0.650 0.200 0.200 0.475 1.000 1.000 1.000 1.000 0.875 0.200 0.875 1.000
>Max. 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000
```

Our new hMPM has matrices with 43 rows and columns, while the original unreduced hMPM has matrices with 121 rows and columns. This is a reduction of 78 rows and columns, with an overall reduction in the size of the matrix by $14641-1849=12792$ elements. This should speed up later calculations greatly.

### 4.6 Arithmetic mean matrices

Now that we have created our MPMs, we might wish to create element-wise arithmetic mean matrices to aid inference and further analysis. For example, we might be interested in developing patch-level means and an overall population mean, but one in which the element means weight each patch and each year equally. For this purpose, we can use the lmean() function. Let's take a look at the element-wise mean raw ahMPM first.

```
cyp2rp_mean <- lmean(cypmatrix2rp)
cyp2rp_mean
> $A
> $A[[1]]
> [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8]
> [1,] 0.08 0.0 0.0 0.00 0.00000000 0.00000000 62.50000000 422.22222222
> [2,] 0.10 0.0 0.0 0.00 0.00000000 0.00000000 62.50000000 422.22222222
> [3,] 0.00 0.1 0.0 0.00 0.00000000 0.00000000 0.00000000 0.00000000
> [4,] 0.00 0.0 0.1 0.00 0.00000000 0.00000000 0.00000000}00.0000000
> [5,] 0.00 0.0 0.0 0.05 0.05000000 0.00000000}00.00000000 0.0000000
> [6,] 0.00 0.0 0.0}00.000.04083333 0.00000000 0.05833333 0.04444444
> [7,] 0.00 0.0 0.0 0.00 0.41964646 0.06666667 0.59949495 0.19555556
> [8,] 0.00 0.0 0.0 0.00}00.14707071 0.33333333 0.21010101 0.59555556
> [9,] 0.00 0.0 0.0 0.00 0.00000000 0.00000000 0.00000000}00.08444444
> [10,] 0.00 0.0 0.0 0.00 0.00000000 0.00000000 0.00000000 0.00000000
> [11,] 0.00 0.0 0.0 0.00 0.00000000 0.00000000 0.00000000 0.00000000
> [,9] [,10] [,11]
> [1,] 1.583333e+03 0.0 0
> [2,] 1.583333e+03 0.0 0
> [3,] 0.000000e+00 0.0 0
> [4,] 0.000000e+00 0.0 0
> [5,] 0.000000e+00 0.0 0
> [6,] 0.000000e+00 0.0 0
> [7,] 6.666667e-02 0.0 0
> [8,] 2.500000e-01 0.2 0
```

```
> [9,] 4.833333e-01 0.0 0
> [10,] 2.000000e-01 0.0 0
> [11,] 0.000000e+00 0.0 0
>
> $A[[2]]
> [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8]
> [1,] 0.08 0.0 0.0 0.00 0.00000000 0.00000000 55.55555556 572.91666667
> [2,] 0.10 0.0 0.0 0.00 0.00000000 0.00000000 55.55555556 572.91666667
> [3,] 0.00 0.1 0.0 0.00 0.00000000 0.00000000 0.00000000 0.00000000
> [4,] 0.00 0.0 0.1 0.00 0.00000000 0.00000000 0.00000000 0.00000000
> [5,] 0.00 0.0 0.0 0.05 0.05000000 0.00000000 0.00000000}00.0000000
> [6,] 0.00 0.0 0.0 0.00 0.05373737 0.06666667 0.07676768 0.05416667
```



```
> [8,] 0.00 0.0 0.0 0.00 0.21676768 0.20000000 0.30966811
> [9,] 0.00 0.0 0.0 0.00 0.00000000 0.00000000 0.02222222 
> [10,] 0.00 0.0 0.0 0.00 0.00000000 0.06666667 0.02222222 0.01250000
> [11,] 0.00 0.0 0.0 0.00 0.00000000 0.00000000 0.00000000 0.00000000
> [,9] [,10] [,11]
> [1,] 2392.8571429 2875.0 6e+03
> [2,] 2392.8571429 2875.0 6e+03
> [3,] 0.0000000 0.0 0e+00
> [4,] 0.0000000 0.0 0e+00
> [5,] 0.0000000 0.0 0e+00
> [6,] 0.0000000 0.0 0e+00
> [7,] 0.0500000 0.0 0e+00
> [8,] 0.4000000 0.1 0e+00
> [9,] 0.3428571 0.2 0e+00
> [10,] 0.2071429 0.6 2e-01
> [11,] 0.0000000 0.1 4e-01
>
> $A[[3]]
> [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9]
> [1,] 0.08 0.0 0.0 0.00 0.00000000 0.0 0.00000000 422.61904762 1625.0000000
> [2,] 0.10 0.0 0.0 0.00 0.00000000 0.0 0.00000000 422.61904762 1625.0000000
> [3,] 0.00 0.1 0.0 0.00 0.00000000 0.0 0.00000000 0.00000000 0.0000000
> [4,] 0.00 0.0}00.1 0.00 0.00000000 0.0 0.00000000 0.00000000 0.0000000
> [5,] 0.00 0.0 0.0 0.05 0.05000000 0.0 0.00000000 0.00000000 0.0000000
> [6,] 0.00 0.0 0.0 0.00 0.04666667 0.0 0.06666667 
> [7,] 0.00 0.0 0.0 0.00 0.35000000 0.1 0.50000000 0.11666667 0.1333333
> [8,] 0.00 0.0 0.0 0.00 0.18666667
> [9,] 0.00 0.0 0.0 0.00 0.00000000 0.0 0.06666667 0.22380952 0.5333333
> [10,] 0.00 0.0 0.0 0.00 0.00000000 0.0 0.00000000 0.07857143 0.0000000
> [11,] 0.00 0.0 0.0 0.00 0.00000000 0.0 0.00000000 0.00000000 0.0000000
> [,10] [,11]
> [1,] 750.0 2500.0
> [2,] 750.0 2500.0
> [3,] 0.0 0.0
> [4,] 0.0 0.0
> [5,] 0.0 0.0
> [6,] 0.0 0.0
> [7,] 0.0 0.0
```

```
> [8,] 0.1 0.0
> [9,] 0.2 0.0
> [10,] 0.4 0.2
> [11,] 0.1 0.4
>
> $A[[4]]
> [,1] [,2] [,3] [,4] [,5] [,6] [,7] 8]
> [1,] 0.08 0.0 0.0 0.00 0.00000000 0.00000000 39.351851852 472.58597884
> [2,] 0.10 0.0 0.0 0.00 0.00000000 0.00000000 39.351851852 472.58597884
> [3,] 0.00 0.1 0.0 0.00 0.00000000 0.00000000 0.000000000 0.00000000
> [4,] 0.00 0.0 0.1 0.00 0.00000000 0.00000000 0.000000000 0.00000000
> [5,] 0.00 0.0 0.0 0.05 0.05000000 0.00000000 0.000000000 0.00000000
> [6,] 0.00 0.0 0.0 0.00 0.04707912 0.02222222 0.067255892 0.04398148
> [7,] 0.00 0.0 0.0 0.00 0.36382155 0.21111111 0.519745070 0.18101852
> [8,] 0.00 0.0 0.0 0.00 0.18350168 0.21111111 
> [9,] 0.00 0.0 0.0 0.00 0.00000000 0.00000000 0.029629630
> [10,] 0.00 0.0 0.0 0.00 0.00000000 0.02222222 0.007407407 0.03035714
> [11,] 0.00 0.0 0.0 0.00 0.00000000 0.00000000 0.000000000 0.00000000
> [,9] [,10] [,11]
> [1,] 1.867063e+03 1.208333e+03 2833.3333333
> [2,] 1.867063e+03 1.208333e+03 2833.3333333
> [3,] 0.000000e+00 0.000000e+00 0.0000000
> [4,] 0.000000e+00 0.000000e+00 0.0000000
> [5,] 0.000000e+00 0.000000e+00 0.0000000
> [6,] 0.000000e+00 0.000000e+00 0.0000000
> [7,] 8.333333e-02 0.000000e+00 0.0000000
> [8,] 3.277778e-01 1.333333e-01 0.0000000
> [9,] 4.531746e-01 1.333333e-01 0.0000000
> [10,] 1.357143e-01 3.333333e-01 0.1333333
> [11,] 0.000000e+00 6.666667e-02 0.2666667
>
>
> $U
> $U[[1]]
> [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8]
> [1,] 0.08 0.0 0.0 0.00 0.00000000 0.00000000 0.00000000 0.00000000
> [2,] 0.10}0.0 0.0 0.00 0.00000000 0.00000000 0.0000000000.00000000
> [3,] 0.00 0.1 0.0 0.00 0.00000000 0.00000000 0.00000000 0.00000000
> [4,] 0.00 0.0}0.0.1 0.00 0.00000000 0.00000000 0.000000000.00000000
> [5,] 0.00 0.0 0.0 0.05 0.05000000 0.00000000 0.00000000 0.00000000
> [6,] 0.00 0.0 0.0 0.00 0.04083333 0.00000000 0.05833333 0.044444444
> [7,] 0.00 0.0 0.0 0.00 0.41964646 0.06666667 0.59949495 0.19555556
> [8,] 0.00 0.0 0.0 0.00 0.14707071 0.33333333 0.21010101 0.59555556
> [9,] 0.00 0.0 0.0 0.00 0.00000000 0.00000000 0.00000000 0.084444444
> [10,] 0.00 0.0 0.0 0.00 0.00000000 0.00000000 0.00000000 0.00000000
> [11,] 0.00 0.0 0.0 0.00 0.00000000 0.00000000 0.00000000 0.00000000
> [,9] [,10] [,11]
> [1,] 0.00000000 0.0 0
> [2,] 0.00000000 0.0 0
> [3,] 0.00000000 0.0 0
> [4,] 0.00000000 0.0 0
```

```
> [5,] 0.00000000 0.0 0
> [6,] 0.00000000 0.0 0
> [7,] 0.06666667 0.0 0
> [8,] 0.25000000 0.2 0
> [9,] 0.48333333 0.0 0
> [10,] 0.20000000 0.0 0
> [11,] 0.00000000 0.0 0
>
> $U[[2]]
> [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9]
> [1,] 0.08 0.0 0.0 0.00 0.00000000 0.00000000 0.00000000 0.00000000 0.0000000
> [2,] 0.10 0.0 0.0 0.00 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000
> [3,] 0.00 0.1 0.0 0.00 0.00000000 0.00000000 0.00000000 0.00000000 0.0000000
> [4,] 0.00 0.0
> [5,] 0.00 0.0 0.0 0.05 0.05000000 0.00000000 0.00000000 0.00000000 0.0000000
> [6,] 0.00 0.0 0.0 0.00 0.05373737 0.06666667 0.07676768 0.05416667 0.0000000
> [7,] 0.00 0.0 0.0 0.00 0.32181818 0.46666667 0.45974026 0.23083333 0.0500000
> [8,] 0.00 0.0 0.0 0.00 0.21676768 0.20000000 0.30966811 0.50166667 0.4000000
> [9,] 0.00 0.0 0.0 0.00 0.00000000 0.00000000 0.02222222 0.14416667 0.3428571
> [10,] 0.00 0.0 0.0 0.00 0.00000000 0.06666667 0.02222222 0.01250000 0.2071429
> [11,] 0.00 0.0 0.0 0.00 0.00000000 0.00000000 0.00000000 0.00000000 0.0000000
> [,10] [,11]
> [1,] 0.0 0.0
> [2,] 0.0 0.0
> [3,] 0.0 0.0
> [4,] 0.0 0.0
> [5,] 0.0 0.0
> [6,] 0.0 0.0
> [7,] 0.0 0.0
> [8,] 0.1 0.0
> [9,] 0.2 0.0
> [10,] 0.6 0.2
> [11,] 0.1 0.4
>
> $U[[3]]
> [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
> [1,] 0.08 0.0 0.0 0.00 0.00000000 0.0 0.00000000 0.00000000 0.0000000 0.0
> [2,] 0.10 0.0
> [3,] 0.00 0.1 0.0 0.00 0.00000000 0.0 0.00000000 0.00000000 0.0000000 0.0
> [4,] 0.00 0.0 0.1 0.00 0.00000000 0.0 0.00000000 0.00000000 0.0000000 0.0
> [5,] 0.00 0.0 0.0 0.05 0.05000000 0.0 0.00000000 0.00000000 0.0000000 0.0
> [6,] 0.00 0.0 0.0 0.00 0.04666667 0.0 0.06666667 0.03333333 0.0000000 0.0
> [7,] 0.00 0.0 0.0 0.00 0.35000000 0.1 0.50000000 0.11666667 0.1333333 0.0
> [8,] 0.00 0.0 0.0 0.00 0.18666667 0.1 0.26666667 0.54761905 0.3333333 
> [9,] 0.00 0.0 0.0 0.00 0.00000000 0.0 0.06666667 0.22380952 0.5333333 0.2
> [10,] 0.00 0.0 0.0 0.00 0.00000000 0.0 0.00000000 0.07857143 0.0000000 0.4
> [11,] 0.00 0.0 0.0 0.00 0.00000000 0.0 0.00000000 0.00000000 0.0000000 0.1
> [,11]
> [1,] 0.0
> [2,] 0.0
> [3,] 0.0
```

```
> [4,] 0.0
> [5,] 0.0
> [6,] 0.0
> [7,] 0.0
> [8,] 0.0
> [9,] 0.0
> [10,] 0.2
> [11,] 0.4
>
> $U[[4]]
> [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8]
> [1,] 0.08 0.0}00.00.00 0.00000000 0.0000000000.000000000 0.00000000
> [2,] 0.10}00.0 0.0 0.00 0.00000000 0.00000000 0.000000000 0.00000000
> [3,] 0.00 0.1 0.0 0.00 0.00000000 0.00000000 0.000000000 0.00000000
> [4,] 0.00 0.0
> [5,] 0.00 0.0 0.0 0.05 0.05000000 0.00000000 0.000000000 0.00000000
> [6,] 0.00 0.0 0.0 0.00 0.04707912 0.02222222 0.067255892 0.04398148
> [7,] 0.00 0.0 0.0 0.00 0.36382155 0.211111111 0.519745070 0.18101852
> [8,] 0.00 0.0 0.0 0.00 0.18350168 0.211111111 0.262145262 0.54828042
> [9,] 0.00 0.0 0.0 0.00 0.00000000 0.00000000 0.029629630}0.1508068
> [10,] 0.00 0.0 0.0 0.00 0.00000000 0.02222222 0.007407407 0.03035714
> [11,] 0.00 0.0 0.0 0.00 0.00000000 0.00000000 0.000000000 0.00000000
> [,9] [,10] [,11]
> [1,] 0.00000000 0.00000000 0.0000000
> [2,] 0.00000000 0.00000000 0.0000000
> [3,] 0.00000000 0.00000000 0.0000000
> [4,] 0.00000000 0.00000000 0.0000000
> [5,] 0.00000000 0.00000000 0.0000000
> [6,] 0.00000000 0.00000000 0.0000000
> [7,] 0.08333333 0.00000000 0.0000000
> [8,] 0.32777778 0.13333333 0.0000000
> [9,] 0.45317460 0.13333333 0.0000000
> [10,] 0.13571429 0.33333333 0.1333333
> [11,] 0.00000000 0.06666667 0.2666667
>
>
> $F
> $F[[1]]
> [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11]
> [1,] 0
> [2,] 0
> [3,] 0
> [4,] }0
> [5,] 
> [6,] 
> [8,] 0
> [9,] 0
>[10,] 
```

$>$




A quick scan through our output shows that we have four matrices. The labels element shows us their order. There is no time term in the labels element, because all matrices are temporal means. Instead, we see that the first three matrices are the patch-level means for patches A, B, and C. This is followed by the overall population mean matrix, listed as patch 0 . It also pays to look at the summary.

```
summary(cyp2rp_mean)
>
> This ahistorical lefkoMat object contains 4 matrices.
>
> Each matrix is square with }11\mathrm{ rows and columns, and a total of 121 elements.
> A total of 118 survival transitions were estimated, with 29.5 per matrix.
> A total of 34 fecundity transitions were estimated, with 8.5 per matrix.
> This lefkoMat object covers 1 population, 4 patches, and O time steps.
```

```
>
> The dataset contains a total of 74 unique individuals and 320 unique transitions.
>
> Survival probability sum check (each matrix represented by column in order):
> [,1] [,2] [,3] [,4]
> Min. 0.000 0.050 0.050 0.050
> 1st Qu. 0.100 0.140 0.140 0.140
> Median 0.200 0.642 0.600 0.467
> Mean 0.407 0.573 0.506 0.495
> 3rd Qu. 0.763 0.917 0.850 0.776
> Max. 1.000 1.000 1.000 1.000
```

We now have four matrices, and we see that these matrices have slightly more elements estimated, on average, than in the raw ahMPM. This is because some of the zeros in the original raw ahMPM were zeros only because of a sampling issue - no individuals actually transitioned through a particular transition in a particular year, but may have transitioned in other years, yielding higher numbers of non-zero elements in the arithmetic mean matrices than in the original raw matrices. Unfortunately these zeros will nonetheless drag these mean element values down artificially, potentially impacting our analyses.

Now let's create the final sets of arithmetic mean matrices for the hMPMs.

```
cyp3rp_mean <- lmean(cypmatrix3rp)
summary(cyp3rp_mean)
>
> This historical lefkoMat object contains 4 matrices.
>
> Each matrix is square with }121\mathrm{ rows and columns, and a total of 14641 elements.
> A total of 309 survival transitions were estimated, with 77.25 per matrix.
> A total of }74\mathrm{ fecundity transitions were estimated, with 18.5 per matrix.
> This lefkoMat object covers 1 population, 4 patches, and 0 time steps.
>
> The dataset contains a total of 74 unique individuals and 320 unique transitions.
>
> Survival probability sum check (each matrix represented by column in order):
> [,1] [,2] [,3] [,4]
>Min. 0.0000 0.000 0.000 0.000
> 1st Qu. 0.0000 0.000 0.000 0.000
> Median 0.0000 0.000 0.000 0.000
> Mean 0.0968 0.156 0.117 0.123
> 3rd Qu. 0.0500 0.200 0.100 0.167
> Max. 1.0000 1.000 1.000 0.991
```

We see many more elements, and many more estimated elements, the latter reflecting the fact that our estimated transitions now reflect different starting stages in time $t$ - 1 .

Let's now look at function-based MPMs in the next chapter.

### 4.7 Points to remember

1. Raw, or empirical, MPMs are created by calculating probability transitions directly from the dataset as proportions of individuals actually transitioning to each stage in the next time out of all individuals alive at some time. Fecundity is calculated in a number of ways, but often involves
direct calculation based either on actual offspring produced or counts / measures of fecundity proxies, such as flowers.
2. Ahistorical MPMs include transitions from each stage at time $t$ to each stage in time $t+1$. Historical MPMs include transitions from each stage in times $t$ and $t-1$, to each stage in time $t+1$. Therefore, the latter include information on one time of individual history, while the former do not.
3. Historical MPMs are sparse and vastly larger than their ahistorical counterparts.
4. Historical MPMs exist in two formats - Ehrlén and deVries. The latter treats survival transitions from a reproductive step in time $t-1$ as originating from a prior unborn stage for the newborn, while the former does not separate maternal fecundity from the fate of the newborn.
5. Use functions rlefko2() and rlefko3() to produce raw ahistorical and historical matrices, respectively.

## Chapter 5

## Matrix Models II: Function-based MPMs

"There is a simple rule here, a rule of legislation, a rule of business, a rule of life: beyond a certain point, complexity is fraud."

- P.J. O'Rourke

Matrix projection models, whether historical or ahistorical, can be categorized as either 1) raw, also called empirical, or 2) function-based, also sometimes inaccurately called complex IPMs (see chapter 7 for details on the differences). The oldest and most common in the literature is the raw MPM, and chapter 4 discusses that approach in detail. In this chapter, we will cover the function-based approach.

Function-based MPMs are created differently than raw MPMs. Although they can look the same to the untrained eye, and although the matrix elements themselves still represent survival-transition probabilities and fecundity rates, matrix elements are calculated differently. Rather than estimating matrix elements as proportions of individuals moving between stages at consecutive occasions or as products of average offspring production from the original dataset, matrix elements in function-based MPMs are estimated using mathematical functions that represent the vital rates or demographic processes underlying the dynamics of the population. These functions are linked together via kernels that treat them typically as conditional rates and probabilities.

How are these vital rate functions estimated? Let's consider a simple function-based size-classified model. In this model, the survival-transition probability is estimated as the product of the probability that an individual in stage $j$ at occasion $t$ survives to occasion $t+1$, and the probability that that same individual becomes stage $k$ in time $t+1$ assuming survival to that time. The latter probability is conditional on the first. Thus, we have the following product, where $\sigma_{j}$ is the survival probability of an individual in stage $j$ at time $t$ to time $t+1$, and $\psi_{k j}$ is the probability of transition from stage $j$ in time $t$ to stage $k$ in time $t+1$ assuming survival to time $t+1$.

$$
\begin{equation*}
a_{k j}=\sigma_{j} \psi_{k j} \tag{5.1}
\end{equation*}
$$

Continuing with this simple case, fecundity elements assuming a pre-breeding model are given as

$$
\begin{equation*}
a_{k j}=f_{k j} \sigma_{k} \tag{5.2}
\end{equation*}
$$

and assuming a post-breeding model are given as

$$
\begin{equation*}
a_{k j}=\sigma_{. j} f_{k j} \tag{5.3}
\end{equation*}
$$

where $f_{k j}$ is the production of stage $k$ offspring by a reproductive individual in stage $j$ in time $t, \sigma_{k}$. is the survival probability of newborns to age 1 at time $t+1$ when they are first censused (assuming
a pre-breeding model), and $\sigma_{. j}$ is the survival of the parent from the occasion preceding reproduction to the time of reproduction (assuming a post-breeding model). Note that reproductive stages are defined differently in the post-breeding model than in the pre-breeding model to account for the need to incorporate parental survival in the former but not in the latter (see Chapter 2 for further details).

Complications in the life history model generally increase the terms to be multiplied in each product to produce the corresponding element. For example, if we are dealing with an herbaceous plant that undergoes vegetative dormancy, then we might change the survival-transition kernel to two different forms, with the equation used dependent on whether the stage we are transitioning to in time $t+1$ is dormancy or not. A dormant plant has no aboveground size, and so the equation used for dormancy would be the following instead of equation 5.1 , where $\rho$. is the probability of sprouting in time $t+1$ assuming survival from time $t$ and stage $D$ is the dormant stage.

$$
\begin{equation*}
a_{D j}=\sigma_{j}\left(1-\rho_{.}\right) \tag{5.4}
\end{equation*}
$$

The probability of transitioning to a non-dormant stage $k$ is given as

$$
\begin{equation*}
a_{k j}=\sigma_{j} \rho . \psi_{k j} \tag{5.5}
\end{equation*}
$$

where $\psi_{k j}$ is the transition probability from stage $j$ in time $t$ to stage $k$ in time $t+1$ assuming both survival to time $t+1$ and sprouting in time $t+1$. This makes $\psi_{k j}$ here a little different than $\psi_{k j}$ in equation 5.1, where we assumed a life history that did not involve the possibility of unobservable life stages.

Size in function-based MPMs is handled using size classes, which often have size bins set to standardized widths. Such widths may be set to 1.0 , as might be typical of count variables, but they can also be set narrower or wider. The function-based approach is very flexible, and allows stages and their associated size classes to be cut quite finely by using different vital rate functions on a case-by-case basis. For example, we may wish to create two sets of stages that have the same size classes, but differ in reproductive status. In the herbaceous perennial plant case that we have been working with, such a matrix would be composed of survival-transition probabilities determined by three functions. The first would be given by equation 5.4 , for the dormant plant case. The second would be the following equation, corresponding to the transition to non-reproductive stage $k$, where $\gamma_{k}$ is the probability of becoming reproductive in time $t+1$ of an individual in size class $k$ in that time.

$$
\begin{equation*}
a_{k j}=\sigma_{j} \rho . \psi_{k j}\left(1-\gamma_{k}\right) \tag{5.6}
\end{equation*}
$$

Finally, we have the following equation giving the survival-transition probability to reproductive stage $k$.

$$
\begin{equation*}
a_{k j}=\sigma_{j} \rho . \psi_{k j} \gamma_{k} \tag{5.7}
\end{equation*}
$$

The functions used in these equations can be developed in several ways. In lefko3, we utilize the linear modeling approach. In this approach, we use $h f v$-formatted datasets to analyze the main vital rates via either generalized linear models or generalized linear mixed models. The best-fit models are then used as inputs into general functions created for the job, with a great deal of flexibility provided for the assumptions inherent in the system being analyzed. Next, we will describe this approach, but also note that linear vital rate models may also be imported (see Chapter 13 for more information).

### 5.1 Developing vital rate models

Package lefko3 employs two methods to estimate the vital rates used to create function-based MPMs. The main method is automated vital rate modeling. Alternatively, users may create their own vital rate models separately and incorporate them into their MPMs. In lefko3, the function modelsearch() is the main tool provided to estimate vital rate functions via the former method. It also allows users to
explore whether history influences component vital rates within a rigorous statistical framework. The results can be used not only to decide whether a historical MPM is justified, but also to develop functionbased hMPMs and ahMPMs, including age-by-stage MPMs and even integral projection models (IPMs). Package lefko3 can estimate linear models for up to 14 different vital rates:

1. Survival probability - This is the probability of surviving from occasion $t$ to occasion $t+1$, given that the individual is in stage $j$ in occasion $t$ (and, if historical, in stage $l$ in occasion $t-1$ ). In lefko3, this parameter may be modeled as a function of up to three size metrics, reproductive status, patch, year, age, and individual identity, and a number of individual or environmental covariates in occasions $t$ and $t$-1. This parameter is required in all function-based matrices.
2. Observation probability - This is the probability of observation in occasion $t+1$ of an individual in stage $k$ given survival from occasion $t$ to occasion $t+1$. This parameter is only used when at least one stage is technically not observable. For example, some plants are capable of vegetative dormancy, in which case they are alive but do not necessarily sprout in all years. In these cases, the probability of sprouting may be estimated as the observation probability. Note that this probability does not refer to observer effort, and so should only be used to differentiate completely unobservable stages where the observation status refers to an important biological phenomenon, such as when individuals may be alive but have a size of zero. In lefko3, this parameter may be modeled as a function of up to three size metrics, reproductive status, patch, year, age, and individual identity, and a number of individual or environmental covariates in occasions $t$ and $t-1$.
3. Primary size transition probability - This is the probability of becoming size $k$ in occasion $t+1$ assuming survival from occasion $t$ to occasion $t+1$ and observation in that time. If multiple size metrics are used, then this refers only to the first of these, which we may refer to as the primary size variable. In lefko3, this parameter may be modeled as a function of up to three size metrics, reproductive status, patch, year, age, and individual identity, and a number of individual or environmental covariates in occasions $t$ and $t-1$. This parameter is required in all function-based size-classified matrices.
4. Secondary size transition probability - This is the probability of becoming size $k$ in occasion $t+1$ assuming survival from occasion $t$ to occasion $t+1$ and observation in that time, within a second size metric used for classification in addition to the primary metric. In lefko3, this parameter may be modeled as a function of up to three size metrics, reproductive status, patch, year, age, and individual identity, and a number of individual or environmental covariates in occasions $t$ and $t-1$.
5. Tertiary size transition probability - This is the probability of becoming size $k$ in occasion $t+1$ assuming survival from occasion $t$ to occasion $t+1$ and observation in that time, within a third size metric used for classification in addition to the primary and secondary metrics. In lefko3, this parameter may be modeled as a function of up to three size metrics, reproductive status, patch, year, age, and individual identity, and a number of individual or environmental covariates in occasions $t$ and $t-1$.
6. Reproduction probability - This is the probability of becoming reproductive in occasion $t+1$ given survival from occasion $t$ to occasion $t+1$, and observation in that time. Note that this should be used only if the researcher wishes to separate breeding from non-breeding mature stages. If all adult stages are potentially reproductive and no separation of reproducing from non-reproducing adults is required by the life history model, then this parameter should not be estimated. In lefko3, this parameter may be modeled as a function of up to three size metrics, reproductive status, patch, year, age, and individual identity, and a number of individual or environmental covariates in occasions $t$ and $t-1$.
7. Fecundity rate - This refers to the rate of production of new individuals into stages in time $t+1$ as offspring from reproduction events happening in time $t$. Under the default setting, this is the rate of successful production of offspring in occasion $t$ by individuals alive, observable, and reproductive in that time, and, if assuming a pre-breeding model and sufficient information is
provided in the dataset, the survival of those offspring into occasion $t+1$ in whatever juvenile class is possible. Thus, the fecundity rate of seed-producing plants might be split into seedlings, which are plants that germinated within a year of seed production, and dormant seeds. Alternatively, it may be given only as produced fruits or seeds, with the survival and germination of seeds provided elsewhere in the MPM development process, such as within a supplement table. An additional setting allows fecundity rate to be estimated using data provided for occasion $t+1$ instead of occasion $t$. In lefko3, this parameter may be modeled as a function of up to three size metrics, reproductive status, patch, year, age, and individual identity, and a number of individual or environmental covariates in occasions $t$ and $t-1$.
8. Juvenile survival probability - This is the probability of surviving from juvenile stage $j$ in occasion $t$ to occasion $t+1$. It is used only when the user wishes to model vital rates for a single size-unclassified juvenile period separately from adults. In lefko3, this parameter may be modeled as a function of up to three size metrics, patch, year, and individual identity, and a number of individual or environmental covariates in occasions $t$ and $t-1$.
9. Juvenile observation probability - This is the probability of observation in occasion $t+1$ of an individual in juvenile stage $j$ in occasion $t$ given survival from occasion $t$ to occasion $t+1$. It is used only when the user wishes to model vital rates for a single size-unclassified juvenile period separately from adults. In lefko3, this parameter may be modeled as a function of up to three size metrics, patch, year, and individual identity, and a number of individual or environmental covariates in occasions $t$ and $t-1$.
10. Juvenile primary size transition probability - This is the probability of becoming stage $k$ in occasion $t+1$ assuming survival from juvenile stage $j$ in occasion $t$ to occasion $t+1$ and observation in that time. It is in terms of a single size metric. It is used only when the user wishes to model vital rates for a single size-unclassified juvenile period separately from adults. In lefko3, this parameter may be modeled as a function of up to three size metrics, patch, year, and a number of individual or environmental covariates in occasions $t$ and $t-1$, and individual identity.
11. Juvenile secondary size transition probability - This is the probability of becoming stage $k$ in occasion $t+1$ assuming survival from juvenile stage $j$ in occasion $t$ to occasion $t+1$ and observation in that time, in a secondary size metric in addition to the primary size metric. It is used only when the user wishes to model vital rates for a single size-unclassified juvenile period separately from adults. In lefko3, this parameter may be modeled as a function of up to three size metrics, patch, year, and individual identity, and a number of individual or environmental covariates in occasions $t$ and $t-1$.
12. Juvenile tertiary size transition probability - This is the probability of becoming stage $k$ in occasion $t+1$ assuming survival from juvenile stage $j$ in occasion $t$ to occasion $t+1$ and observation in that time, in a tertiary size metric in addition to the primary and secondary size metrics. It is used only when the user wishes to model vital rates for a single size-unclassified juvenile period separately from adults. In lefko3, this parameter may be modeled as a function of up to three size metrics, patch, year, and individual identity, and a number of individual or environmental covariates in occasions $t$ and $t-1$.
13. Juvenile reproduction probability - This is the probability of reproducing in mature stage $k$ in occasion $t+1$ given survival from juvenile stage $j$ in occasion $t$ to occasion $t+1$, and observation in that time. It is used only when the user wishes to model vital rates for a single size-unclassified juvenile period separately from adults. In lefko3, this parameter may be modeled as a function of up to three size metrics, patch, year, and individual identity, and a number of individual or environmental covariates in occasions $t$ and $t-1$.
14. Juvenile maturity probability - This is the probability of becoming mature in occasion $t+1$ given survival from juvenile stage $j$ in occasion $t$ to occasion $t+1$. It is used only when the user wishes to model vital rates for a single size-unclassified juvenile period separately from adults. In lefko3, this parameter may be modeled as a function of up to three size metrics, patch, year, and individual identity, and a number of individual or environmental covariates in occasions $t$
and $t-1$. Note that this parameter denotes transition to maturity.

Of these fourteen vital rates, most users will estimate at least parameters (1) survival probability, (3) primary size transition probability, and (7) fecundity rate. These three are the default set for function modelsearch(). Parameters (2) observation probability and (6) reproduction probability may be used when some stages are included that are completely unobservable (and so do not have any size), or that are mature but non-reproductive, respectively. Parameters (4) secondary size transition and (5) tertiary size transition should only be used when size classification involves more than one size variable. Parameters (8) through (14) should only be added if the dataset contains juvenile individuals transitioning to maturity, and these juveniles live essentially as a single juvenile stage for some amount of time before transitioning to maturity, or before transitioning to a stage that is size-classified in the same manner as adult stages are. If juveniles can be classified by size similarly to adults (or at least on the same scale), then only vital rates (1) through (7) should be used and stage groups can be used with supplement tables to disallow transitions back to juvenile stages. If multiple juvenile stages exist on a different size classification system than adults, then stage groups may also be included as categorical variables in linear vital rate modeling in rates (1) through (7) to stratify vital rate models properly.

### 5.1.1 Options in modeling vital rates with modelsearch()

Function modelsearch() handles the entire modeling process using an exhaustive model building process combined with information theoretical model selection using model AICc and the number of estimated parameters in each model. It first develops global models, and then develops all nested models via exhaustive model building. Finally, it determines and selects the best-fit models. Users may provide this function with information about the following:

1. Individual history - Are the matrices to be built historical or ahistorical? If the former, then the state of the individual in occasion $t-1$ will be included in modeling.
2. Modeling approach - Should the models be estimated as generalized linear models (GLMs) or as generalized linear mixed models (GLMMs, the default)? While most function-based matrix models are estimated via the former, the latter approach can account for repeated observations of the same individual by including individual identity as a random factor. Mixed models also allow time and patch to be treated as random variables, which allows for broader and more theoretically sound inference.
3. Suite of factors - Should both size and reproductive status be tested as causal factors? Should two-way interactions be included? Should only the y-intercept be estimated? Should age be tested exclusively?
4. Suite of vital rates - Which adult demographic parameters should be estimated? The defaults are (1) survival, (3) primary size, and (7) fecundity. Should (2) observation status, (4) secondary size, (5) tertiary size, or (6) reproductive status also be modeled?
5. Juvenile vital rate estimation - Should juvenile parameters (8) through (14) also be modeled?
6. Best-fit criterion - If a model with fewer parameters exists with $\Delta A I C c \leq 2.0$, then should this model be used as the best-fit model (the default), or should the model with the minimum $A I C c$ always be chosen?
7. Size distribution - Should size be modeled as a continuous variable under a Gaussian or a gamma distribution, or as a count variable under either a Poisson or negative binomial distribution? If as a count variable, then should the distribution be zero-inflated to account for excess zeros, zero-truncated to account for a lack of zeros, or left unaltered?
8. Fecundity distribution - Should fecundity be modeled as a continuous variable under a Gaussian or a gamma distribution, or as a count variable under either a Poisson or negative binomial distribution? If a count variable, then should the distribution be zero-inflated to account for excess zeros, zero-truncated to account for a lack of zeros, or left unaltered?
9. Timing of fecundity - Although fecundity refers to the production of offspring in stages alive in time $t+1$, fecundity may be measured differently in different studies. Function modelsearch() assumes that linear models of fecundity use a metric counted or measured in occasion $t$ as the response. This applies well with most herbaceous plant datasets, where flowers or seeds produced in one year might be the fecundity response measured. However, users not wishing to follow this default behavior can use the fectime option to stipulate a fecundity metric measured in occasion $t+1$.
10. Age - Is an age-classified (Leslie) or age-by-stage MPM the main goal?
11. Individual covariates and environmental state variables - Should the effects of any individual or environmental covariates on vital rates also be tested? If so, should they be treated as fixed, quantitative terms or as random, categorical terms?
12. Density - Should spatial density be included as a factor in modeling? Note that this refers to local, spatial density with respect to each individual, rather than to the population size.
13. Stage groups - If stage groups are noted in the stageframe, then should they also be used as fixed, independent categorical predictors in modeling?
14. Censoring - Should data points marked as questionable be used in or excluded from modeling?
15. Variable names - The names of all relevant variables in the dataset need to be specified. Note that the default behavior assumes variable names produced via the historicalize3() or verticalize3() functions, which produce standardized historically-formatted vertical (hfv format) datasets.
16. Global model only - Should a full exhaustive model building and selection exercise be performed (the default), or should only the global model be built and parameterized?
17. Accuracy - Should the accuracy of binomial models and simple $R^{2}$ of size and fecundity models be evaluated (the default)?

Once all decisions have been made and associated input terms have been provided, modelsearch() goes to work. The result is a lefkoMod object, which is a list in which the first 14 elements are the best-fit models developed for each vital rate. These are followed by an equivalent number of elements showing the full model tables developed and tested, followed by a table of parameter names (object paramnames), an element detailing the best-fit criterion used (object criterion), and ending on quality control data showing the number of individuals and the number of unique transitions used in the estimation of each model, and the accuracy or simple $R^{2}$ of each model (object qc). Depending on user choices, linear modeling is handled through the $\operatorname{lm}()$ and $\operatorname{glm}()$ functions in the stats package, the 1 mer () and glmer () functions in 1 me 4 , the $\operatorname{glmmTMB()}$ function in $\operatorname{glmmTMB}$, the glm.nb() function in package MASS, the vglm () function in package VGAM, or the zeroinfl() function in package pscl. Exhaustive model building proceeds through the dredge() function in package MuMIn (Bartoń, 2014). Model selection is handled through assessment of $A I C c$ and the degrees of freedom ( $k$ ) estimated per model by default (see 6. Best-fit criterion above).

If modelsearch() is set for historical analysis (historical = TRUE, the default), then the decision of whether to develop a historical MPM can be made on the basis of whether any best-fit vital rate model includes size or reproductive status in occasion $t$-1. If at least one vital rate does, then a historical MPM is justified. Otherwise, it is not. Regardless, the output can be used to create a function-based MPM in the next step.

### 5.2 Assessing vital rates in Cypripedium candidum

Let's now work on the Cypripedium candidum case study. We introduced the model and code for for our stageframe and supplements in Chapter 2. However, as it has likely been a while since we have viewed that chapter, let's take a look again at the life history model that we will use (note that this is the same as figure 2.3).


Figure 5.1: Life history model of Cypripedium candidum for use in function-based MPMs

This is a fairly large life history model with two adult life stages for each number of sprouts - a reproductive stage that flowers and a non-reproductive stage that does not flower. Let's load the raw dataset first. Then we will set up our stageframe and standardize the data into a historically formatted vertical ( $h f v$ ) data frame.

```
data(cypdata)
stagevector <- c("SD", "P1", "P2", "P3", "SL", "D", "V1", "V2", "V3", "V4",
    "V5", "V6", "V7", "V8", "V9", "V10", "V11", "V12", "V13", "V14", "V15", "V16",
    "V17", "V18", "V19", "V20", "V21", "V22", "V23", "V24", "F1", "F2", "F3",
    "F4", "F5", "F6", "F7", "F8", "F9", "F10", "F11", "F12", "F13", "F14", "F15",
    "F16", "F17", "F18", "F19", "F20", "F21", "F22", "F23", "F24")
indataset <- c(0, 0, 0, 0, 0, rep(1, 49))
sizevector <- c(0, 0, 0, 0, 0, seq(from = 0, t = 24), seq(from = 1, to = 24))
repvector <- c(0, 0, 0, 0, 0, rep(0, 25), rep(1, 24))
obsvector <- c(0, 0, 0, 0, 0, 0, rep(1, 48))
matvector <- c(0, 0, 0, 0, 0, rep(1, 49))
immvector <- c(0, 1, 1, 1, 1, rep(0, 49))
propvector <- c(1, rep(0, 53))
comments <- c("Dormant seed", "Yr1 protocorm", "Yr2 protocorm", "Yr3 protocorm",
    "Seedling", "Veg dorm", "Veg adult 1 stem", "Veg adult 2 stems",
    "Veg adult 3 stems", "Veg adult 4 stems", "Veg adult 5 stems",
    "Veg adult 6 stems", "Veg adult 7 stems", "Veg adult 8 stems",
    "Veg adult 9 stems", "Veg adult }10\mathrm{ stems", "Veg adult 11 stems",
    "Veg adult 12 stems", "Veg adult 13 stems", "Veg adult 14 stems",
    "Veg adult 15 stems", "Veg adult 16 stems", "Veg adult 17 stems",
    "Veg adult 18 stems", "Veg adult 19 stems", "Veg adult 20 stems",
```

```
    "Veg adult 21 stems", "Veg adult 22 stems", "Veg adult 23 stems",
    "Veg adult 24 stems", "Flo adult 1 stem", "Flo adult 2 stems",
    "Flo adult 3 stems", "Flo adult 4 stems", "Flo adult 5 stems",
    "Flo adult 6 stems", "Flo adult 7 stems", "Flo adult 8 stems",
    "Flo adult 9 stems", "Flo adult 10 stems", "Flo adult 11 stems",
    "Flo adult 12 stems", "Flo adult 13 stems", "Flo adult 14 stems",
    "Flo adult 15 stems", "Flo adult 16 stems", "Flo adult 17 stems",
    "Flo adult 18 stems", "Flo adult 19 stems", "Flo adult 20 stems",
    "Flo adult 21 stems", "Flo adult 22 stems", "Flo adult 23 stems",
    "Flo adult 24 stems")
cypframe_fb <- sf_create(sizes = sizevector, stagenames = stagevector,
    repstatus = repvector, obsstatus = obsvector, matstatus = matvector,
    propstatus = propvector, immstatus = immvector, indataset = indataset,
    comments = comments)
cypfb_v1 <- verticalize3(data = cypdata, noyears = 6, firstyear = 2004,
    patchidcol = "patch", individcol = "plantid", blocksize = 4,
    sizeacol = "Inf2.04", sizebcol = "Inf.04", sizeccol = "Veg.04",
    repstracol = "Inf.04", repstrbcol = "Inf2.04", fecacol = "Pod.04",
    stageassign = cypframe_fb, stagesize = "sizeadded", NAas0 = TRUE,
    age_offset = 4)
```

Now we will load the supplement tables that we will use for the ahistorical and historical MPMs. Note that these are slightly different from the previous chapter (chapter 4), because of the different life history model that we are using.

```
seeds_per_fruit <- 5000
sl_mult <- 0.7
cypsupp2_fb <- supplemental(stage3 = c("SD", "P1", "P2", "P3", "SL", "SL", "D",
    "V1", "V2", "V3", "SD", "P1"),
    stage2 = c("SD", "SD", "P1", "P2", "P3", "SL", "SL", "SL", "SL", "SL", "rep",
        "rep"),
    eststage3 = c(NA, NA, NA, NA, NA, NA, "D", "V1", "V2", "V3", NA, NA),
    eststage2 = c(NA, NA, NA, NA, NA, NA, "D", "D", "D", "D", NA, NA),
    givenrate = c(0.08, 0.1, 0.1, 0.1, 0.05, 0.05, NA, NA, NA, NA, NA, NA),
    multiplier = c(NA, NA, NA, NA, NA, NA, sl_mult, sl_mult, sl_mult, sl_mult,
        0.5 * seeds_per_fruit, 0.5 * seeds_per_fruit),
    type = c("S", "S", "S", "S", "S", "S", "S", "S", "S", "S", "R", "R"),
    stageframe = cypframe_fb, historical = FALSE)
cypsupp3_fb <- supplemental(stage3 = c("SD", "SD", "P1", "P1", "P2", "P3", "SL",
        "SL", "SL", "D", "V1", "V2", "V3", "D", "V1", "V2", "V3", "mat", "mat",
        "mat", "mat", "SD", "P1"),
    stage2 = c("SD", "SD", "SD", "SD", "P1", "P2", "P3", "SL", "SL", "SL", "SL",
        "SL", "SL", "SL", "SL", "SL", "SL", "D", "V1", "V2", "V3", "rep", "rep"),
    stage1 = c("SD", "rep", "SD", "rep", "SD", "P1", "P2", "P3", "SL", "P3", "P3",
        "P3", "P3", "SL", "SL", "SL", "SL", "SL", "SL", "SL", "SL", "mat", "mat"),
    eststage3 = c(NA, NA, NA, NA, NA, NA, NA, NA, NA, "D", "V1", "V2", "V3", "D",
        "V1", "V2", "V3", "mat", "mat", "mat", "mat", NA, NA),
    eststage2 = c(NA, NA, NA, NA, NA, NA, NA, NA, NA, "D", "D", "D", "D", "D",
```

```
    "D", "D", "D", "D", "V1", "V2", "V3", NA, NA),
eststage1 = c(NA, NA, NA, NA, NA, NA, NA, NA, NA, "D", "D", "D", "D", "D",
    "D", "D", "D", "V1", "V1", "V1", "V1", NA, NA),
givenrate \(=c(0.08,0.08,0.1,0.1,0.1,0.1,0.1,0.05,0.05, N A, N A, N A, N A\),
    NA, NA, NA, NA, NA, NA, NA, NA, NA, NA),
multiplier \(=c\left(N A, N A, N A, N A, N A, N A, N A, N A, N A, s l_{-} m u l t, s l \_m u l t\right.\),
    sl_mult, sl_mult, sl_mult, sl_mult, sl_mult, sl_mult, 1, 1, 1, 1,
        0.5 * seeds_per_fruit, 0.5 * seeds_per_fruit),
type = c ("S", "S", "S", "S", "S", "S", "S", "S", "S", "S", "S", "S", "S", "S",
    "S", "S", "S", "S", "S", "S", "S", "R", "R"),
type_t12 = c("S", "F", "S", "F", "S", "S", "S", "S", "S", "S", "S", "S", "S",
    "S", "S", "S", "S", "S", "S", "S", "S", "S", "S"),
stageframe = cypframe_fb, historical = TRUE)
```

Before we can develop our vital rate models, we need to determine the proper distributions for size and fecundity. As a reminder, size in the Cypripedium dataset is given as a series of numbers of sprouts, and our proxy for size can be either the number of flowers or the number of fruits produced by an individual. So, size and fecundity in this dataset are all non-negative integers, or count variables. Count variables generally break the assumptions of a Gaussian distribution because the mean and variance are strongly related, possible values are discrete, and distributions are often dramatically skewed. Additionally, because we have absorbed individuals with a size of zero into the dormant stage, there are no zeros left for the size model and so we will need a zero-truncated distribution. Fecundity, in contrast, is a count variable that includes zeros and so we should test whether we need a zero-inflated model.

Package lefko3 includes a function that can help in determining which distributions to use, as well as to explore the quality of the data with respect to the linear models that we wish to develop: hfv_qc(). Let's try this function out. Note that we need to specify which vital rate models we will be modeling (the vitalrates field below), because the data frame that we will input will be subset according to this decision. We also need to enter the names of some of the variables we are using, here in particular for size, since we are not using the default variables.

```
hfv_qc(cypfb_v1, vitalrates = c("surv", "obs", "size", "repst", "fec"),
    patch = "patchid", size = c("size3added", "size2added", "size1added"))
> Survival:
>
> Data subset has 58 variables and 320 transitions.
>
> Variable alive3 has O missing values.
> Variable alive3 is a binomial variable.
>
>
> Observation status:
>
> Data subset has 58 variables and 303 transitions.
>
> Variable obsstatus3 has 0 missing values.
> Variable obsstatus3 is a binomial variable.
>
>
> Primary size:
>
> Data subset has }58\mathrm{ variables and 288 transitions.
```

```
>
> Variable size3added has O missing values.
> Variable size3added appears to be an integer variable.
Variable size3added is fully positive, lacking even Os.
Overdispersion test:
    Mean size3added is 3.653
    The variance in size3added is 13.41
    The probability of this dispersion level by chance assuming that
    the true mean size3added = variance in size3added,
    and an alternative hypothesis of overdispersion, is 3.721e-138
    Variable size3added is significantly overdispersed.
Zero-inflation and truncation tests:
    Mean lambda in size3added is 0.02592
    The actual number of 0s in size3added is 0
    The expected number of 0s in size3added under the null hypothesis is 7.465
    The probability of this deviation in Os from expectation by chance is 0.9964
    Variable size3added is not significantly zero-inflated.
    Variable size3added does not include Os, suggesting that a zero-truncated distribution may be
Reproductive status:
    Data subset has 58 variables and 288 transitions.
    Variable repstatus3 has 0 missing values.
    Variable repstatus3 is a binomial variable.
Fecundity:
    Data subset has 58 variables and }118\mathrm{ transitions.
    Variable feca2 has 0 missing values.
    Variable feca2 appears to be an integer variable.
    Variable feca2 is fully non-negative.
    Overdispersion test:
        Mean feca2 is 0.7881
        The variance in feca2 is 1.536
        The probability of this dispersion level by chance assuming that
        the true mean feca2 = variance in feca2,
        and an alternative hypothesis of overdispersion, is 0.1193
        Dispersion level in feca2 matches expectation.
    Zero-inflation and truncation tests:
        Mean lambda in feca2 is 0.4547
```

```
> The actual number of 0s in feca2 is 68
> The expected number of 0s in feca2 under the null hypothesis is 53.65
> The probability of this deviation in 0s from expectation by chance is 5.904e-06
> Variable feca2 is significantly zero-inflated.
```

The results show that size is significantly overdispersed and zero-truncated. This means that we cannot use the Poisson distribution, and should instead use the zero-truncated negative binomial distribution. Fecundity is not significantly overdispersed, but it is significantly zero-inflated, so we will require a zero-inflated Poisson distribution. Looking over the other variables, we see that they match the binomial distribution (required for probabilities), and we also see the changing sample size used for the different data subsets that will be used to conduct the linear modeling.

### 5.3 Setting up function modelsearch ()

Next, we will create a full suite of vital rate models for the Cypripedium candidum dataset using function modelsearch(). Before proceeding, we need to decide on the linear model building strategy, the correct vital rates to model, the proper statistical distributions for estimated vital rates, the proper parameterization for each vital rate, and the strategy for determination of the best-fit models.

First, we must determine the model building strategy. In most cases, the best procedure will be through mixed linear models in which monitoring occasion and individual identity are random terms. We will set monitoring occasion as random because we wish to make inferences for the population as a whole and do not wish to restrict ourselves to inference only for the years monitored (i.e. our distribution of monitoring occasions sampled is itself a sample of the population in time). We will set individual identity as random because many or most of the individuals that we have sampled to produce our dataset yield multiple observation data points across time. Thus, we will set approach = "mixed". To make sure that time and individual identity are treated as random, we will set the proper variable names for indiv and year, corresponding to individual identity (individ by default), and to occasion $t$ (year2 by default). The year .as.random option is set to random by default. Setting year.as.random to FALSE would make time a fixed categorical variable in all cases. The patch.as.random option functions similarly with respect to patches / subpopulations.

The mixed modeling approach is usually preferable to the GLM approach because of its treatment of repeat samples. Indeed, treating each transition as independent of every other leads to pseudoreplication if transitions can come from the same individual. Mixed modeling deals with this problem by treating the individual as a random factor. It also yields truer statistical power while the GLM approach can inflate Type I error. Users of package lefko3 wishing to use a standard generalized linear modeling strategy should set approach = "glm". In this case, individual identity is not used, time is a fixed categorical factor (as is patch, if used), and all observed transitions are treated as independent.

Next, we must determine which vital rates to model. The default settings for modelsearch() estimate 1) survival probability, 3) primary size distribution, and 7) fecundity, which are the minimum three vital rates required for a full MPM. Observation probability (option obs in vitalrates) should only be included when a life history stage or size exists that cannot be observed. For example, in the case of a plant with vegetative dormancy, the observation probability can be thought of as the sprouting probability, which is a biologically meaningful vital rate (Shefferson et al., 2001). Further, reproduction status (option repst in vitalrates) should only be modeled if size classification needs to be stratified by the ability to reproduce, as when some individuals with no fecundity occur within stages that also include individuals producing offspring. Since Cypripedium candidum is capable of long bouts of vegetative dormancy, since we wish to stratify the population into reproductive and non-reproductive stages of the same size classes, and since we have no data derived from juvenile individuals, we will set vitalrates = c("surv", "obs", "size", "repst", "fec") and we will not set juvestimate and juvsize.

Third, we need to set the proper statistical distribution for each parameter. Survival probability, observation probability, reproductive status, and maturity status are all modeled as binomial variables, and this cannot be changed. In the case of this population of Cypripedium candidum, size was measured as the number of stems and so is a count variable. Likewise, fecundity is estimated as the number of fruits produced per plant, and so is also a count variable. We have already performed tests for overdispersion and zero-inflation, and we are also aware that size in observed stages cannot be zero, requiring zero-truncation in that parameter. So we will set size to the zero-truncated negative binomial distribution, and fecundity to the zero-inflated Poisson distribution.

Fourth, we need the proper model parameterizations for each vital rate, using the suite option. The default, suite = "main", under the mixed model setting (approach = "mixed") starts with the estimation of global models that include size and reproductive status in occasions $t$ and $t-1$ as fixed factors, with individual identity and time in occasion $t$ (year $t$ ) set as random categorical terms. Setting suite = "full" will yield global models that also include all two-way interactions among fixed terms ("full" is the only setting with interaction terms). If the population is not stratified by reproductive status, then suite = "size" will eliminate reproductive status terms and use all others in the global model. If size is not important, then suite = "rep" will eliminate size but keep reproductive status and all other terms. Finally, suite $=$ "cons" will result in a global model in which neither reproductive status nor size are considered. Other terms can be specified, including individual covariates and age.

Finally, we should determine the proper strategy for the determination of the best-fit model. Model building proceeds through the dredge() function in package MuMIn (Bartoń, 2014), and each model has an associated AICc value. The default setting in lefko3 (bestfit = "AICc\&k") will compare all models within 2.0 AICc units of the model with $\triangle A I C c=0$, and choose the one with the lowest degrees of freedom. This approach is generally better than the alternative, which simply uses the model with $\Delta A I C c=0$ (bestfit $=$ "AICc"), as all models within 2.0 AICc units of that model are equally parsimonious and so fewer degrees of freedom result from fewer parameters estimated (Burnham and Anderson, 2002).

In the model building exercise below, we will use the suite $=$ "main" option to run all main effects only. Normally we would set to suite = "full", but running all effects including their twoway interactions will likely tie up our computers for a little too long for this tutorial. To keep tabs on what is going on, we will set quiet = "partial", which will yield messages showing where the modeling is currently running as it goes. While this runs, feel free to make yourself a coffee or a cappuccino, or perhaps something stronger (no judgement here...).

```
cypmodels3p <- modelsearch(cypfb_v1, historical = TRUE, approach = "mixed",
    vitalrates = c("surv", "obs", "size", "repst", "fec"), patch = "patchid",
    sizedist = "negbin", size.trunc = TRUE, fecdist = "poisson", fec.zero = TRUE,
    suite = "main", size = c("size3added", "size2added", "size1added"),
    quiet = "partial")
```

Once done, we will summarize the output with the summary () function.

```
summary(cypmodels3p)
> This LefkoMod object includes 5 linear models.
> Best-fit model criterion used: aicc&k
>
>
>
> Survival model:
> Generalized linear mixed model fit by maximum likelihood (Laplace
    Approximation) [glmerMod]
    Family: binomial ( logit )
    Formula: alive3 ~ size2added + (1 | year2) + (1 | patchid) + (1 | individ)
```

```
> Data: subdata
> AIC BIC logLik deviance df.resid
130.1321 148.9737-60.0660 120.1321 315
Random effects:
Groups Name Std.Dev.
    individ (Intercept) 1.199e+00
    year2 (Intercept) 5.117e-05
    patchid (Intercept) 1.172e-05
Number of obs: 320, groups: individ, 74; year2, 5; patchid, 3
Fixed Effects:
(Intercept) size2added
> 2.0356 0.6343
optimizer (Nelder_Mead) convergence code: O (OK) ; 0 optimizer warnings; 1 lme4 warnings
>
>
>
Observation model:
Generalized linear mixed model fit by maximum likelihood (Laplace
        Approximation) [glmerMod]
    Family: binomial ( logit )
Formula: obsstatus3 ~ size2added + (1 | year2) + (1 | patchid) + (1 |
        individ)
        Data: subdata
            AIC BIC logLik deviance df.resid
120.2567 138.8254 -55.1284 110.2567 298
Random effects:
    Groups Name Std.Dev.
    individ (Intercept) 0.0000
    year2 (Intercept) 0.8776
    patchid (Intercept) 0.0000
Number of obs: 303, groups: individ, 70; year2, 5; patchid, 3
Fixed Effects:
(Intercept) size2added
        2.4904 0.3134
optimizer (Nelder_Mead) convergence code: O (OK) ; O optimizer warnings; 1 lme4 warnings
>
>
>
Size model:
Formula: size3added ~ (1 | year2) + (1 | patchid) + (1 | individ)
Data: subdata
    AIC BIC logLik df.resid
1009.9750 1028.2898-499.9875 283
Random-effects (co)variances:
Conditional model:
    Groups Name Std.Dev.
    year2 (Intercept) 0.1133
    patchid (Intercept) 0.2118
    individ (Intercept) 1.0320
>
```

```
> Number of obs: 288 / Conditional model: year2, 5; patchid, 3; individ, 70
>
> Dispersion parameter for truncated_nbinom2 family (): 2.73e+07
>
Fixed Effects:
>
Conditional model:
> (Intercept)
> 0.587
>
>
>
Secondary size model:
[1] 1
>
>
>
Tertiary size model:
[1] 1
>
>
>
Reproductive status model:
Generalized linear mixed model fit by maximum likelihood (Laplace
        Approximation) [glmerMod]
Family: binomial ( logit )
Formula: repstatus3 ~ repstatus2 + size2added + (1 | year2) + (1 | patchid) +
            (1 | individ)
        Data: subdata
            AIC BIC logLik deviance df.resid
333.4037 355.3815 -160.7019 321.4037 282
Random effects:
Groups Name Std.Dev.
individ (Intercept) 0.1776
year2 (Intercept) 0.6636
patchid (Intercept) 0.3501
Number of obs: 288, groups: individ, 70; year2, 5; patchid, 3
Fixed Effects:
(Intercept) repstatus2 size2added
-1.3836 1.5543 0.1788
>
>
>
Fecundity model:
Formula:
feca2 ~ size2added + (1 | year2) + (1 | patchid) + (1 | individ)
Zero inflation:
~size2added + (1 | year2) + (1 | patchid) + (1 | individ)
Data: subdata
    AIC BIC logLik df.resid
251.4551 279.1619 -115.7275 108
```

```
Random-effects (co)variances:
Conditional model:
Groups Name Std.Dev.
    year2 (Intercept) 5.610e-01
    patchid (Intercept) 2.283e-01
    individ (Intercept) 4.630e-08
Zero-inflation model:
Groups Name Std.Dev.
year2 (Intercept) 3.340e-07
    patchid (Intercept) 1.724e-12
    individ (Intercept) 2.057e-04
Number of obs: 118 / Conditional model: year2, 5; patchid, 3; individ, 51 / Zero-inflation model:
>
Fixed Effects:
Conditional model:
(Intercept) size2added
        -0.56501 0.06247
Zero-inflation model:
(Intercept) size2added
            3.840 -1.588
>
>
Juvenile survival model:
[1] 1
>
>
>
> Juvenile observation model:
[1] 1
>
>
>
> Juvenile size model:
[1] 1
>
>
>
> Juvenile secondary size model:
[1] 1
>
>
>
> Juvenile tertiary size model:
[1] 1
>
>
```

```
>
Juvenile reproduction model:
[1] 1
>
>
>
> Juvenile maturity model:
[1] 1
>
>
>
>
>
Number of models in survival table: 16
Number of models in observation table: 16
Number of models in size table: 16
Number of models in secondary size table: 1
Number of models in tertiary size table: 1
Number of models in reproduction status table: 16
Number of models in fecundity table: 241
Number of models in juvenile survival table: 1
Number of models in juvenile observation table: 1
Number of models in juvenile size table: 1
Number of models in juvenile secondary size table: 1
Number of models in juvenile tertiary size table: 1
Number of models in juvenile reproduction table: 1
Number of models in juvenile maturity table: 1
>
>
>
>
>
> General model parameter names (column 1), and
specific names used in these models (column 2):
    parameter_names mainparams
1
> 2
> 3
```

```
>4 alive in time t+1
alive in time t+1 surv3
>5 observed in time t+1 obs3
>6 sizea in time t+1 size3
> 7 sizeb in time t+1 sizeb3
> sizec in time t+1 sizec3
> 9 reproductive status in time t+1 repst3
10 fecundity in time t+1 fec3
11 fecundity in time t fec2
12 sizea in time t size2
13 sizea in time t-1 size1
14 sizeb in time t sizeb2
15 sizeb in time t-1 sizeb1
> 16 sizec in time t sizec2
> 17 sizec in time t-1 sizec1
> 18 reproductive status in time t repst2
> 19 reproductive status in time t-1 repst1
> 20 maturity status in time t+1 matst3
> 21 maturity status in time t matst2
    22 age in time t age
    23 density in time t density
    24 individual covariate a in time t indcova2
> 25 individual covariate a in time t-1 indcova1
> 26 individual covariate b in time t indcovb2
> 27 individual covariate b in time t-1 indcovb1
> 28 individual covariate c in time t indcovc2
> 29 individual covariate c in time t-1 indcovc1
> 30 stage group in time t group2
>31 stage group in time t-1 group1
>
>
>
>
>
> Quality control:
>
> Survival model estimated with 74 individuals and 320 individual transitions.
> Survival model accuracy is 0.947.
> Observation status model estimated with 70 individuals and 303 individual transitions.
> Observation status model accuracy is 0.95.
> Primary size model estimated with 70 individuals and 288 individual transitions.
> Primary size model R-squared is 0.82.
> Secondary size model not estimated.
> Tertiary size model not estimated.
> Reproductive status model estimated with 70 individuals and 288 individual transitions.
> Reproductive status model accuracy is 0.74.
> Fecundity model estimated with 51 individuals and }118\mathrm{ individual transitions.
> Fecundity model R-squared is 0.535.
> Juvenile survival model not estimated.
> Juvenile observation status model not estimated.
> Juvenile primary size model not estimated.
> Juvenile secondary size model not estimated.
```

```
> Juvenile tertiary size model not estimated.
> Juvenile reproductive status model not estimated.
> Juvenile maturity status model not estimated.
```

The above code may take some time to run, depending on how powerful a computer it is run on. It is sufficiently complicated that R might crash if it is run while the user does other things in R or R Studio, so please leave your R or R Studio session alone while it runs. As the function runs, it will likely produce some text related to the modeling. Some of the text is meant to provide guideposts to where the modeling stands. Since the vital rates run in order, the user can get a handle on which vital rate best-fit models are finished and which are still waiting to be developed. Other text relates to notes, warnings, and other messages from the modeling functions used, which are found in other CRAN-based packages. We have set quiet = "partial" to prevent this text from getting very long (quiet = TRUE eliminates just about all of it).

The final object is a class lefkoMod object. This is an S3 object, meaning that it is a list, and it is composed of 31 elements. The first 14 elements are the best-fit models, and are in the formats of the functions used to develop them. The next 14 elements are the model tables, showing all of the models tested for each vital rate in order of $\triangle A I C c$ ( $\triangle A I C c$ is given as the difference between the $A I C c$ values of the current model and the model with the lowest $A I C c$ ). The final three elements are a data frame describing the parameters tested, the criterion used for determination of the best-fit model, and quality control data used in summary() calls.

The summary of this lefkoMod object is quite informative. We start off by seeing the best-fit model chosen for each vital rate. In the case of vital rates that are not estimated, they are set to constants, either 1 or 0 . Below that, we see how many models were built and compared for each vital rate. The next section is something of a crib sheet, showing us the names of different parameters used in the modeling. Finally, we see a section on quality control. This last section includes information on the numbers of individuals as well as individual transitions used to parameterize each model, and also shows their accuracy or simple $R^{2}$ values. We see very high accuracy in survival and observation status, and a high $R^{2}$ for primary size, but poorer accuracy in reproduction status and low $R^{2}$ for fecundity.

Looking over the models, we can also see from the summary that historical size and reproductive status are not included in any best-fit models. This suggests that an ahistorical MPM is a more parsimonious choice of MPM than an ahistorical MPM. Regardless, in order to compare MPMs for educational purposes, we will also create a purely ahistorical model set. We will set quiet = "partial" to see less text output as the function works, but you may remove that option to view the guidepost text. Note that a vital rate model set that includes historical terms should never be used to make an ahistorical MPM.

```
cypmodels2p <- modelsearch(cypfb_v1, historical = FALSE, approach = "mixed",
    vitalrates = c("surv", "obs", "size", "repst", "fec"), patch = "patchid",
    sizedist = "negbin", size.trunc = TRUE, fecdist = "poisson", fec.zero = TRUE,
    suite = "main", size = c("size3added", "size2added", "size1added"),
    quiet = "partial")
```

Let's see a summary of this lefkoMod object.

```
summary(cypmodels2p)
> This LefkoMod object includes 5 linear models.
> Best-fit model criterion used: aicc&k
>
>
>
```

```
Survival model:
Generalized linear mixed model fit by maximum likelihood (Laplace
        Approximation) [glmerMod]
    Family: binomial ( logit )
Formula: alive3 ~ size2added + (1 | year2) + (1 | patchid) + (1 | individ)
        Data: subdata
            AIC BIC logLik deviance df.resid
130.1321 148.9737-60.0660 120.1321 315
Random effects:
    Groups Name Std.Dev.
    individ (Intercept) 1.199e+00
    year2 (Intercept) 5.117e-05
    patchid (Intercept) 1.172e-05
    Number of obs: 320, groups: individ, 74; year2, 5; patchid, 3
    Fixed Effects:
    (Intercept) size2added
    2.0356 0.6343 (Nelder_Mead) convergence code: O (OK) ; 0 optimizer warnings; 1 lme4 warnings
>
>
>
Observation model:
Generalized linear mixed model fit by maximum likelihood (Laplace
        Approximation) [glmerMod]
    Family: binomial ( logit )
    Formula: obsstatus3 ~ size2added + (1 | year2) + (1 | patchid) + (1 |
            individ)
        Data: subdata
            AIC BIC logLik deviance df.resid
120.2567 138.8254 -55.1284 110.2567 298
Random effects:
    Groups Name Std.Dev.
    individ (Intercept) 0.0000
    year2 (Intercept) 0.8776
    patchid (Intercept) 0.0000
    Number of obs: 303, groups: individ, 70; year2, 5; patchid, 3
    Fixed Effects:
    (Intercept) size2added
    2.4904 0.3134
optimizer (Nelder_Mead) convergence code: O (OK) ; O optimizer warnings; 1 lme4 warnings
>
>
>
Size model:
Formula: size3added ~ (1 | year2) + (1 | patchid) + (1 | individ)
Data: subdata
    AIC BIC logLik df.resid
1009.9750 1028.2898-499.9875 283
Random-effects (co)variances:
Conditional model:
```

```
Groups Name Std.Dev.
year2 (Intercept) 0.1133
patchid (Intercept) 0.2118
individ (Intercept) 1.0320
Number of obs: 288 / Conditional model: year2, 5; patchid, 3; individ, 70
Dispersion parameter for truncated_nbinom2 family (): 2.73e+07
Fixed Effects:
Conditional model:
(Intercept)
            0.587
Secondary size model:
[1] 1
Tertiary size model:
[1] 1
Reproductive status model:
Generalized linear mixed model fit by maximum likelihood (Laplace
    Approximation) [glmerMod]
    Family: binomial ( logit )
Formula: repstatus3 ~ repstatus2 + size2added + (1 | year2) + (1 | patchid) +
            (1 | individ)
        Data: subdata
            AIC BIC logLik deviance df.resid
    333.4037 355.3815 -160.7019 321.4037 282
Random effects:
    Groups Name Std.Dev.
    individ (Intercept) 0.1776
    year2 (Intercept) 0.6636
    patchid (Intercept) 0.3501
Number of obs: 288, groups: individ, 70; year2, 5; patchid, 3
Fixed Effects:
(Intercept) repstatus2 size2added
    -1.3836 1.5543 0.1788
Fecundity model:
Formula:
feca2 ~ size2added + (1 | year2) + (1 | patchid) + (1 | individ)
```

$>$
$>$
$>$

```
> Zero inflation:
> ~size2added + (1 | year2) + (1 | patchid) + (1 | individ)
Data: subdata
AIC BIC logLik df.resid
    251.4551 279.1619 -115.7275 108
Random-effects (co)variances:
Conditional model:
Groups Name Std.Dev.
year2 (Intercept) 5.610e-01
    patchid (Intercept) 2.283e-01
    individ (Intercept) 4.630e-08
    Zero-inflation model:
    Groups Name Std.Dev.
    year2 (Intercept) 3.340e-07
    patchid (Intercept) 1.724e-12
    individ (Intercept) 2.057e-04
Number of obs: 118 / Conditional model: year2, 5; patchid, 3; individ, 51 / Zero-inflation model: y
Fixed Effects:
>
Conditional model:
(Intercept) size2added
-0.56501 0.06247
Zero-inflation model:
(Intercept) size2added
            3.840 -1.588
>
>
Juvenile survival model:
[1] 1
>
>
>
> Juvenile observation model:
[1] 1
Juvenile size model:
[1] 1
>
>
>
> Juvenile secondary size model:
[1] 1
>
>
```

```
>
Juvenile tertiary size model:
[1] 1
>
>
>
Juvenile reproduction model:
[1] 1
>
>
>
Juvenile maturity model:
[1] 1
>
>
>
>
>
Number of models in survival table: 4
Number of models in observation table: 4
Number of models in size table: 4
Number of models in secondary size table: 1
Number of models in tertiary size table: 1
Number of models in reproduction status table: 4
Number of models in fecundity table: 14
Number of models in juvenile survival table: 1
Number of models in juvenile observation table: 1
Number of models in juvenile size table: 1
Number of models in juvenile secondary size table: 1
Number of models in juvenile tertiary size table: 1
Number of models in juvenile reproduction table: 1
Number of models in juvenile maturity table: 1
>
>
>
>
>
General model parameter names (column 1), and
```

```
> specific names used in these models (column 2):
> parameter_names mainparams
> 1 time t year2
> 2 individual individ
> 3 patch patch
>4 alive in time t+1 surv3
> 5 observed in time t+1 obs3
> 6 sizea in time t+1 size3
> 7 sizeb in time t+1 sizeb3
8 sizec in time t+1 sizec3
> 9 reproductive status in time t+1 repst3
> 10 fecundity in time t+1 fec3
> 11 fecundity in time t fec2
> 12 sizea in time t size2
> 13 sizea in time t-1 size1
> 14 sizeb in time t sizeb2
> 15 sizeb in time t-1 sizeb1
> 16 sizec in time t sizec2
> 17 sizec in time t-1 sizec1
> 18 reproductive status in time t repst2
> 19 reproductive status in time t-1 repst1
> 20 maturity status in time t+1 matst3
> 21 maturity status in time t matst2
22 age in time t age
density in time t density
> 24 individual covariate a in time t indcova2
> 25 individual covariate a in time t-1 indcova1
> 26 individual covariate b in time t indcovb2
> 27 individual covariate b in time t-1 indcovb1
> 28 individual covariate c in time t indcovc2
> 29 individual covariate c in time t-1 indcovc1
30 stage group in time t group2
>31 stage group in time t-1 group1
>
>
>
>
>
> Quality control:
>
> Survival model estimated with 74 individuals and 320 individual transitions.
> Survival model accuracy is 0.947.
> Observation status model estimated with 70 individuals and 303 individual transitions.
> Observation status model accuracy is 0.95.
> Primary size model estimated with 70 individuals and 288 individual transitions.
> Primary size model R-squared is 0.82.
> Secondary size model not estimated.
> Tertiary size model not estimated.
> Reproductive status model estimated with 70 individuals and 288 individual transitions.
> Reproductive status model accuracy is 0.74.
> Fecundity model estimated with 51 individuals and 118 individual transitions.
```

```
> Fecundity model R-squared is 0.535.
> Juvenile survival model not estimated.
> Juvenile observation status model not estimated.
> Juvenile primary size model not estimated.
> Juvenile secondary size model not estimated.
> Juvenile tertiary size model not estimated.
> Juvenile reproductive status model not estimated.
> Juvenile maturity status model not estimated.
```

The ahistorical set of models is the same as the historical case, since historical terms dropped out of the historical analysis. However, this is never assured, and care should be taken to test individual history properly and choose the proper model set for MPM creation.

Function modelsearch() has many options that can lead to very complicated global models and searches for vital rate models. For example, up to three size metrics may be used for classification, and these may assume different distributions. Also, up to three individual/environmental covariates coded in the $h f v$ dataset may also be included in models, and these covariates may be quantitative (used as fixed factors), or categorical and either fixed or random (options indcova, indcovb, and indcovc, with logical settings random.indcova, random.indcovb, and random.indcovc). If stage groups are used and were coded into the stageframe, then stage group can be included in models as a categorical fixed factor (logical setting test.group). Model tables may be eliminated from the lefkoMod output using show.model.tables = FALSE, and model selection can be eliminated with the global model alone being estimated with the option global.only = TRUE.

### 5.4 Size and fecundity distributions

Before we create MPMs, let's discuss response distributions a bit further. The probabilities of survival, observation, and reproductive status are automatically set to the binomial distribution in function modelsearch(), and this cannot be altered. However, the probability of size transition and the fecundity rate can be set to the Gaussian, gamma, Poisson, or negative binomial distributions, with zero-inflated and zero-truncated versions of the Poisson and negative binomial also available. If size or fecundity rate is a continuous variable (i.e., not an integer or count variable), then it should be set to the Gaussian distribution if reasonably symmetric. If there is a strong right skew, then a gamma distribution may be warranted. This can be explored by using the plot() function in conjunction with density(). For example, here we look at the primary size distribution used in the Cypripedium candidum case (figure 5.2). Please note that there is no established test to determine whether a Gaussian distribution is better than the gamma, so this decision should be made at the user's discretion.
plot(density(cypfb_v1\$size2added), main = "", xlab = "Sprout number", bty = "n")

If size or fecundity is a count variable, then it should be set to the Poisson distribution if the mean equals the variance, and this can be tested with the $h f v_{\mathrm{v}} \mathrm{qc}()$ function. The negative binomial distribution is provided in cases where the assumption that the mean equals the variance is clearly broken. We do not encourage the use of the negative binomial except in such cases, as the extra parameters estimated for the negative binomial distribution reduce the power of the modeling exercises conducted.

The Poisson and the negative binomial distributions both predict specific numbers of zeros in the response variable. If excess zeros occur within the dataset even after including the observation status and reproductive status as vital rates to absorb zeros, then a zero-inflated Poisson or negative binomial distribution may be used. These modeling approaches work by parameterizing two separate models. The first is a binomial model, typically assuming a logit link, to predict excess zero responses. The second model is called a conditional model, and it is the Poisson or negative binomial model that is


Figure 5.2: Density distribution of size in Cypripedium candidum
used to predict non-zero and only some zero responses. In effect, a zero-inflated model is actually two models in which zeros are assumed to be predicted under potentially different processes than the remaining counts. Because an extra model is built to cover excess zeros, zero-inflated models are much more complex and can include many more parameters than their non-inflated counterparts. The principle of parsimony suggests that they should only be used when there are significantly more zeros than expected.

Cases may arise in which zeros do not occur in either size or fecundity. For these situations, we provide zero-truncated distributions. This may occur in size if all cases of size $=0$ are absorbed by observation status, leaving only positive integers for the size of observed individuals. For example, if an unobservable stage such as vegetative dormancy occurs and absorbs all cases of size $=0$, then a zero-truncated Poisson or negative binomial distribution will be more appropriate than the equivalent distribution without zero-truncation. It can also occur if all cases of fecundity $=0$ are absorbed by reproductive status. Such distributions only involve the estimation of single, conditional models, and so are simpler than zero-inflated models.

We have assumed that users will only use a single size variable in classification. However, cases may arise in which stages are classified under two or three size variables, and these two or three size variables are also used in vital rate modeling. These size variables can be treated as independent for the purposes of modeling, meaning that they can also have completely different distributions.

Let's now build some function-based MPMs.

### 5.5 Using lefkoMod objects to create function-based MPMs

MPM creation from a demographic dataset can be accomplished with nine different functions:

1. rlefko2() - Creates raw ahistorical MPMs given a dataset, a stageframe, and either a supplement table or a reproductive matrix and an overwrite table.
2. rlefko3() - Creates raw historical MPMs given a dataset, a stageframe, and either a supplement table or a reproductive matrix and an overwrite table.
3. arlefko2() - Creates raw ahistorical age x stage MPMs given a dataset, a stageframe, and either a supplement table or a reproductive matrix and an overwrite table.
4. rleslie() - Creates raw age-based (Leslie) MPMs given a dataset.
5. flefko2() - Creates function-based ahistorical MPMs given a dataset, a set of models, a stageframe, and either a supplement table or a reproductive matrix and an overwrite table.
6. flefko3() - Creates function-based historical MPMs given a dataset, a set of models, a stageframe, and either a supplement table or a reproductive matrix and an overwrite table.
7. aflefko2() - Creates function-based ahistorical age $x$ stage MPMs given a dataset, a set of models, a stageframe, and either a supplement table or a reproductive matrix and an overwrite table.
8. fleslie() - Creates function-based age-based (Leslie) MPMs given a dataset and a set of models.
9. mpm_create() - A general, all-purpose MPM creation function. All other MPM creation functions are actually wrappers for this function.

These functions incorporate binary kernels developed to handle the estimation of matrix elements quickly and efficiently. A single run of flefko3(), for example, should be able to yield all annual historical matrices for all patches for the Cypripedium candidum dataset provided with lefko3 in under a minute on most machines (14s or so on the author's 2019 MacBook Pro with 2.3 GHz 8 Core Intel Core i9), while a single run of flefko2() should typically run in a fraction of a second. Parallel computing should not be necessary even with the slowest of current machines, provided that the machine is current enough to handle at least R 3.6.3.

The output for each of these functions is a lefkoMat object, which is an S3 object and is described in detail in Chapter 4. These lists include elements A, U, and F, which are themselves lists of complete projection matrices, survival-transition matrices, and fecundity matrices, respectively. For example, code such as matobject\$A[[1]] would access the first complete projection matrix in a lefkoMat object named matobject. They are followed by elements referred to as ahstages, hstages, and agestages, which provide data frames describing the stages (technically showing the edited stageframe), the historical stage-pairs, and the age-stage pairs shown in the order in which they occur within the matrices, respectively. The labels element is a data frame giving a description of each matrix in the order in which it occurs within the A, U, and F elements, including the population, patch, and monitoring time designations. The final elements are quality control outputs that vary in content depending on whether the output matrices are raw or function-based, but nonetheless always include at least the numbers of individuals and individual-transitions used for estimation.

Let's create function-based MPMs with the lefkoMod objects that we created in this chapter. We will start with the ahistorical function-based MPM. We will also look at a summary of the resulting lefkoMat object, described in section 1.7.1.

```
cypmatrix2fp <- flefko2(stageframe = cypframe_fb, supplement = cypsupp2_fb,
    modelsuite = cypmodels2p, data = cypfb_v1)
summary(cypmatrix2fp)
>
> This ahistorical lefkoMat object contains 15 matrices.
>
> Each matrix is square with }54\mathrm{ rows and columns, and a total of 2916 elements.
> A total of 36165 survival transitions were estimated, with 2411 per matrix.
> A total of }720\mathrm{ fecundity transitions were estimated, with 48 per matrix.
> This lefkoMat object covers 1 population, 3 patches, and 5 time steps.
>
> The dataset contains a total of 74 unique individuals and 320 unique transitions.
>
> Vital rate modeling quality control:
>
> Survival estimated with 74 individuals and 320 individual transitions.
> Observation estimated with 70 individuals and 303 individual transitions.
> Primary size estimated with 70 individuals and 288 individual transitions.
```

```
> Secondary size transition not estimated.
> Tertiary size transition not estimated.
> Reproductive status estimated with 70 individuals and 288 individual transitions.
> Fecundity estimated with 51 individuals and 118 individual transitions.
> Juvenile survival not estimated.
> Juvenile observation probability not estimated.
> Juvenile primary size transition not estimated.
> Juvenile secondary size transition not estimated.
> Juvenile tertiary size transition not estimated.
> Juvenile reproduction probability not estimated.
> Juvenile maturity transition probability not estimated.
>
> Survival probability sum check (each matrix represented by column in order):
> [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11] [,12]
> Min. 0.050 0.050 0.050 0.050 0.050 0.050 0.050 0.050 0.050 0.050 0.050 0.050
> 1st Qu. 0.991 0.991 0.991 0.991 0.991 0.991 0.991 0.991 0.991 0.991 0.991 0.991
> Median 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000
> Mean 0.916 0.917 0.917 0.917 0.914 0.916 0.917 0.917 0.917 0.914 0.914 0.916
> 3rd Qu. 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000
> Max. 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000
> [,13] [,14] [,15]
> Min. 0.050 0.050 0.050
> 1st Qu. 0.991 0.991 0.991
> Median 1.000 1.000 1.000
> Mean 0.916 0.916 0.913
> 3rd Qu. 1.000 1.000 1.000
> Max. 1.000 1.000 1.000
```

Here we have fifteen matrices. Since there are six years of data, there are a total of five pairs of years with which to estimate transitions. There are also three patches, yielding fifteen total matrices. We also have $54^{2}=2916$ elements per matrix, of which 2459 elements are estimated (the rest are zeros). Let's look at the first function-based matrix.

```
print(cypmatrix2fp$A[[1]], digits = 3)
> [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
> [1,] 0.08 0.0 0.0 0.00 0.0000 0.00e+00 0.00e+00 0.00e+00 0.00e+00 0.00e+00
> [2,] 0.10}0.0.0 0.0 0.00 0.0000 0.00e+00 0.00e+00 0.00e+00 0.00e+00 0.00e+0
> [3,] 0.00 0.1 0.0 0.00 0.0000 0.00e+00 0.00e+00 0.00e+00 0.00e+00 0.00e+00
> [4,] 0.00 0.0 0.1 0.00 0.0000 0.00e+00 0.00e+00 0.00e+00 0.00e+00 0.00e+00
> [5,] 0.00 0.0 0.0 0.05 0.0500 0.00e+00 0.00e+00 0.00e+00 0.00e+00 0.00e+00
> [6,] 0.00 0.0 0.0 0.00 0.0329 4.70e-02 3.69e-02 2.81e-02 2.10e-02 1.56e-02
> [7,] 0.00 0.0 0.0 0.00 0.1628 2.33e-01 2.38e-01 2.36e-01 2.28e-01 2.17e-01
> [8,] 0.00 0.0 0.0 0.00 0.1035 1.48e-01 1.52e-01 1.50e-01 1.45e-01 1.38e-01
> [9,] 0.00 0.0 0.0 0.00 0.0658 9.40e-02 9.64e-02 9.54e-02 9.23e-02 8.77e-02
> [10,] 0.00 0.0 0.0 0.00 0.0000 5.98e-02 6.13e-02 6.07e-02 5.87e-02 5.58e-02
> [11,] 0.00 0.0 0.0 0.00 0.0000 3.80e-02 3.90e-02 3.86e-02 3.73e-02 3.55e-02
> [12,] 0.00 0.0 0.0 0.00 0.0000 2.42e-02 2.48e-02 2.45e-02 2.37e-02 2.25e-02
> [13,] 0.00 0.0 0.0 0.00 0.0000 1.54e-02 1.58e-02 1.56e-02 1.51e-02 1.43e-02
> [14,] 0.00 0.0 0.0 0.00 0.0000 9.78e-03 1.00e-02 9.92e-03 9.59e-03 9.12e-03
> [15,] 0.00 0.0 0.0 0.00 0.0000 6.22e-03 6.38e-03 6.31e-03 6.10e-03 5.80e-03
> [16,] 0.00 0.0 0.0 0.00 0.0000 3.96e-03 4.05e-03 4.01e-03 3.88e-03 3.69e-03
> [17,] 0.00 0.0 0.0 0.00 0.0000 2.52e-03 2.58e-03 2.55e-03 2.47e-03 2.34e-03
```

$>[18] \quad 0.00 \quad 0.0 \quad 0.0 \quad 0.00 \quad 0.00001 .60 \mathrm{e}-,031.64 \mathrm{e}-031.62 \mathrm{e}-03 \quad 1.57 \mathrm{e}-03 \quad 1.49 \mathrm{e}-03$
$>\quad[, 11] \quad[, 12] \quad[, 13] \quad[, 14] \quad[, 15] \quad[, 16] \quad[, 17] \quad[, 18]$
$>\quad[1] 0.00 \mathrm{e}+,000.00 \mathrm{e}+00 \quad 0.00 \mathrm{e}+00 \quad 0.00 \mathrm{e}+000.00 \mathrm{e}+00 \quad 0.00 \mathrm{e}+00 \quad 0.00 \mathrm{e}+000.00 \mathrm{e}+00$
$>\quad[2] 0.00 \mathrm{e}+,000.00 \mathrm{e}+000.00 \mathrm{e}+000.00 \mathrm{e}+000.00 \mathrm{e}+00 \quad 0.00 \mathrm{e}+00 \quad 0.00 \mathrm{e}+00 \quad 0.00 \mathrm{e}+00$
$>\quad[3] 0.00 \mathrm{e}+,000.00 \mathrm{e}+000.00 \mathrm{e}+00 \quad 0.00 \mathrm{e}+000.00 \mathrm{e}+00 \quad 0.00 \mathrm{e}+00 \quad 0.00 \mathrm{e}+00 \quad 0.00 \mathrm{e}+00$
$>$ [4,] $0.00 \mathrm{e}+000.00 \mathrm{e}+000.00 \mathrm{e}+000.00 \mathrm{e}+00 \quad 0.00 \mathrm{e}+00 \quad 0.00 \mathrm{e}+00 \quad 0.00 \mathrm{e}+00 \quad 0.00 \mathrm{e}+00$
$>\quad[5] 0.00 \mathrm{e}+,000.00 \mathrm{e}+00 \quad 0.00 \mathrm{e}+000.00 \mathrm{e}+000.00 \mathrm{e}+00 \quad 0.00 \mathrm{e}+00 \quad 0.00 \mathrm{e}+00 \quad 0.00 \mathrm{e}+00$
$>\quad[6] 1.15 \mathrm{e}-,028.47 \mathrm{e}-036.21 \mathrm{e}-034.55 \mathrm{e}-033.33 \mathrm{e}-032.44 \mathrm{e}-031.78 \mathrm{e}-031.30 \mathrm{e}-03$
$>$ [7,] 2.03e-01 1.89e-01 1.73e-01 1.57e-01 1.42e-01 1.27e-01 1.13e-01 9.94e-02
$>\quad[8] 1.29 \mathrm{e}-,011.20 \mathrm{e}-011.10 \mathrm{e}-011.00 \mathrm{e}-019.03 \mathrm{e}-028.08 \mathrm{e}-027.17 \mathrm{e}-026.32 \mathrm{e}-02$
$>\quad[9] 8.22 \mathrm{e}-,027.62 \mathrm{e}-027.00 \mathrm{e}-026.37 \mathrm{e}-025.74 \mathrm{e}-025.14 \mathrm{e}-024.56 \mathrm{e}-024.02 \mathrm{e}-02$
$>$ [10,] 5.23e-02 $4.85 \mathrm{e}-024.45 \mathrm{e}-024.05 \mathrm{e}-023.65 \mathrm{e}-023.27 \mathrm{e}-022.90 \mathrm{e}-022.56 \mathrm{e}-02$
$>[11] 3.32 \mathrm{e}-,023.08 \mathrm{e}-022.83 \mathrm{e}-022.57 \mathrm{e}-022.32 \mathrm{e}-022.08 \mathrm{e}-021.84 \mathrm{e}-021.63 \mathrm{e}-02$
$>$ [12,] 2.11e-02 1.96e-02 1.80e-02 $1.64 \mathrm{e}-021.48 \mathrm{e}-021.32 \mathrm{e}-02 \quad 1.17 \mathrm{e}-02 \quad 1.03 \mathrm{e}-02$
$>[13] 1.34 \mathrm{e}-,021.25 \mathrm{e}-021.14 \mathrm{e}-021.04 \mathrm{e}-029.39 \mathrm{e}-03 \quad 8.40 \mathrm{e}-03 \quad 7.46 \mathrm{e}-036.57 \mathrm{e}-03$
$>$ [14,] 8.55e-03 7.93e-03 7.28e-03 6.62e-03 5.97e-03 5.34e-03 4.74e-03 4.18e-03
$>[15] 5.44 \mathrm{e}-,035.04 \mathrm{e}-034.63 \mathrm{e}-034.21 \mathrm{e}-033.80 \mathrm{e}-033.40 \mathrm{e}-033.02 \mathrm{e}-032.66 \mathrm{e}-03$
$>[16] 3.46 \mathrm{e}-,033.21 \mathrm{e}-032.94 \mathrm{e}-032.68 \mathrm{e}-032.41 \mathrm{e}-032.16 \mathrm{e}-031.92 \mathrm{e}-031.69 \mathrm{e}-03$
$>$ [17,] 2.20e-03 2.04e-03 1.87e-03 1.70e-03 1.54e-03 1.37e-03 1.22e-03 1.08e-03
$>[18] 1.40 \mathrm{e}-,031.30 \mathrm{e}-031.19 \mathrm{e}-031.08 \mathrm{e}-039.77 \mathrm{e}-048.74 \mathrm{e}-047.76 \mathrm{e}-046.84 \mathrm{e}-04$
$>$ [,19] [,20] [,21] [,22] [,23] [,24] [,25] [,26]
$>$ [1,] $0.00 \mathrm{e}+000.00 \mathrm{e}+000.00 \mathrm{e}+000.00 \mathrm{e}+00 \quad 0.00 \mathrm{e}+00 \quad 0.00 \mathrm{e}+00 \quad 0.00 \mathrm{e}+00 \quad 0.00 \mathrm{e}+00$
$>$ [2,] $0.00 \mathrm{e}+000.00 \mathrm{e}+000.00 \mathrm{e}+000.00 \mathrm{e}+00 \quad 0.00 \mathrm{e}+000.00 \mathrm{e}+00 \quad 0.00 \mathrm{e}+00 \quad 0.00 \mathrm{e}+00$
$>$ [3,] $0.00 \mathrm{e}+000.00 \mathrm{e}+000.00 \mathrm{e}+000.00 \mathrm{e}+000.00 \mathrm{e}+000.00 \mathrm{e}+000.00 \mathrm{e}+000.00 \mathrm{e}+00$
$>$ [4,] $0.00 \mathrm{e}+000.00 \mathrm{e}+000.00 \mathrm{e}+000.00 \mathrm{e}+000.00 \mathrm{e}+000.00 \mathrm{e}+00 \quad 0.00 \mathrm{e}+000.00 \mathrm{e}+00$
$>\quad[5] 0.00 \mathrm{e}+,000.00 \mathrm{e}+000.00 \mathrm{e}+000.00 \mathrm{e}+000.00 \mathrm{e}+00 \quad 0.00 \mathrm{e}+00 \quad 0.00 \mathrm{e}+00 \quad 0.00 \mathrm{e}+00$
$>\quad[6] 9.54 \mathrm{e}-,046.97 \mathrm{e}-04 \quad 5.10 \mathrm{e}-04 \quad 3.73 \mathrm{e}-04 \quad 2.72 \mathrm{e}-04 \quad 1.99 \mathrm{e}-04 \quad 1.46 \mathrm{e}-04 \quad 1.06 \mathrm{e}-04$
$>$ [7,] 8.70e-02 7.58e-02 6.56e-02 5.66e-02 4.85e-02 4.15e-02 3.54e-02 3.01e-02
$>\quad[8] 5.54 \mathrm{e}-,024.82 \mathrm{e}-024.17 \mathrm{e}-023.60 \mathrm{e}-023.09 \mathrm{e}-022.64 \mathrm{e}-022.25 \mathrm{e}-021.91 \mathrm{e}-02$
$>$ [9,] 3.52e-02 3.06e-02 2.65e-02 2.29e-02 1.96e-02 1.68e-02 1.43e-02 1.22e-02
$>[10] 2.24 \mathrm{e}-,021.95 \mathrm{e}-021.69 \mathrm{e}-021.45 \mathrm{e}-021.25 \mathrm{e}-021.07 \mathrm{e}-029.10 \mathrm{e}-037.73 \mathrm{e}-03$
$>[11] 1.42 \mathrm{e}-,021.24 \mathrm{e}-021.07 \mathrm{e}-029.25 \mathrm{e}-037.94 \mathrm{e}-036.79 \mathrm{e}-035.79 \mathrm{e}-034.92 \mathrm{e}-03$
> [12,] 9.05e-03 7.88e-03 6.83e-03 5.88e-03 5.05e-03 4.32e-03 3.68e-03 3.13e-03
$>[13] 5.76 \mathrm{e}-,035.01 \mathrm{e}-034.34 \mathrm{e}-033.74 \mathrm{e}-033.21 \mathrm{e}-032.75 \mathrm{e}-032.34 \mathrm{e}-031.99 \mathrm{e}-03$
$>[14] 3.66 \mathrm{e}-,033.19 \mathrm{e}-032.76 \mathrm{e}-032.38 \mathrm{e}-032.04 \mathrm{e}-031.75 \mathrm{e}-031.49 \mathrm{e}-031.26 \mathrm{e}-03$
> [15,] 2.33e-03 2.03e-03 1.76e-03 1.51e-03 1.30e-03 1.11e-03 9.46e-04 8.04e-04
$>[16] 1.48 \mathrm{e}-,031.29 \mathrm{e}-031.12 \mathrm{e}-03 \quad 9.62 \mathrm{e}-04 \quad 8.26 \mathrm{e}-047.06 \mathrm{e}-04 \quad 6.02 \mathrm{e}-04 \quad 5.11 \mathrm{e}-04$
$>[17] 9.42 \mathrm{e}-,048.20 \mathrm{e}-047.10 \mathrm{e}-04 \quad 6.12 \mathrm{e}-04 \quad 5.25 \mathrm{e}-04 \quad 4.49 \mathrm{e}-04 \quad 3.83 \mathrm{e}-04 \quad 3.25 \mathrm{e}-04$
$>$ [18,] 5.99e-04 5.21e-04 4.51e-04 3.89e-04 3.34e-04 2.86e-04 2.43e-04 2.07e-04
> [,27] [,28] [,29] [,30] [,31] [,32] [,33] [,34]
$>\quad[1] 0.00 \mathrm{e}+,000.00 \mathrm{e}+000.00 \mathrm{e}+000.00 \mathrm{e}+002.26 \mathrm{e}+031.65 \mathrm{e}+03 \quad 7.10 \mathrm{e}+021.88 \mathrm{e}+02$
$>\quad[2] 0.00 \mathrm{e}+,000.00 \mathrm{e}+00 \quad 0.00 \mathrm{e}+00 \quad 0.00 \mathrm{e}+002.26 \mathrm{e}+031.65 \mathrm{e}+037.10 \mathrm{e}+021.88 \mathrm{e}+02$
$>\quad[3] 0.00 \mathrm{e}+,000.00 \mathrm{e}+000.00 \mathrm{e}+000.00 \mathrm{e}+000.00 \mathrm{e}+00 \quad 0.00 \mathrm{e}+00 \quad 0.00 \mathrm{e}+000.00 \mathrm{e}+00$
$>[4] 0.00 \mathrm{e}+,000.00 \mathrm{e}+000.00 \mathrm{e}+000.00 \mathrm{e}+000.00 \mathrm{e}+000.00 \mathrm{e}+00 \quad 0.00 \mathrm{e}+00 \quad 0.00 \mathrm{e}+00$
$>[5] 0.00 \mathrm{e}+,000.00 \mathrm{e}+000.00 \mathrm{e}+000.00 \mathrm{e}+000.00 \mathrm{e}+000.00 \mathrm{e}+000.00 \mathrm{e}+000.00 \mathrm{e}+00$
$>$ [6,] 7.78e-05 5.69e-05 4.16e-05 3.04e-05 3.69e-02 2.81e-02 2.10e-02 1.56e-02
$>$ [7,] 2.55e-02 $2.16 \mathrm{e}-021.82 \mathrm{e}-021.54 \mathrm{e}-021.18 \mathrm{e}-011.10 \mathrm{e}-01 \quad 9.94 \mathrm{e}-028.85 \mathrm{e}-02$
$>\quad[8] 1.62 \mathrm{e}-,021.37 \mathrm{e}-021.16 \mathrm{e}-029.77 \mathrm{e}-037.54 \mathrm{e}-026.98 \mathrm{e}-026.32 \mathrm{e}-025.63 \mathrm{e}-02$
$>\quad[9] 1.03 \mathrm{e}-,028.72 \mathrm{e}-037.36 \mathrm{e}-036.21 \mathrm{e}-034.79 \mathrm{e}-024.44 \mathrm{e}-024.02 \mathrm{e}-023.58 \mathrm{e}-02$
$>[10] 6.55 \mathrm{e}-,035.55 \mathrm{e}-034.68 \mathrm{e}-033.95 \mathrm{e}-033.05 \mathrm{e}-022.82 \mathrm{e}-022.56 \mathrm{e}-022.27 \mathrm{e}-02$
$>[11] 4.17 \mathrm{e}-,033.53 \mathrm{e}-032.98 \mathrm{e}-032.51 \mathrm{e}-031.94 \mathrm{e}-02 \quad 1.80 \mathrm{e}-02 \quad 1.63 \mathrm{e}-02 \quad 1.45 \mathrm{e}-02$

```
> [6,] 7.78e-05 5.69e-05 4.16e-05 3.04e-05
> [7,] 5.70e-03 4.78e-03 4.01e-03 3.36e-03
> [8,] 3.63e-03 3.04e-03 2.55e-03 2.13e-03
> [9,] 2.31e-03 1.93e-03 1.62e-03 1.36e-03
> [10,] 1.47e-03 1.23e-03 1.03e-03 8.63e-04
> [11,] 9.32e-04 7.82e-04 6.55e-04 5.49e-04
> [12,] 5.93e-04 4.97e-04 4.17e-04 3.49e-04
> [13,] 3.77e-04 3.16e-04 2.65e-04 2.22e-04
> [14,] 2.40e-04 2.01e-04 1.69e-04 1.41e-04
> [15,] 1.53e-04 1.28e-04 1.07e-04 8.98e-05
> [16,] 9.70e-05 8.13e-05 6.82e-05 5.71e-05
> [17,] 6.17e-05 5.17e-05 4.33e-05 3.63e-05
> [18,] 3.92e-05 3.29e-05 2.76e-05 2.31e-05
> [ reached getOption("max.print") -- omitted 36 rows ]
```

Notice that this matrix is bigger and more full of non-zero values than the associated raw matrix. This is due to the fact that function-based matrices use kernels to estimate every estimable element, while raw matrices are populated only by proportions of individuals making actual transitions. Thus, if no individuals make a particular transition, then a raw matrix will assign that survival-transition probability a zero, while a function-based matrix will provide a non-zero estimate based on the chosen vital rate linear models.

Let's now estimate the function-based historical MPM. Because this will have huge dimensions, we will set sparse_output $=$ TRUE to make sure that it does not take up too much memory (standard matrix format should yield a lefkoMat object taking up over three gigabytes, while the sparse matrix format should yield a lefkoMat object taking up only a little over 40 megabytes).

```
cypmatrix3fp <- flefko3(stageframe = cypframe_fb, supplement = cypsupp3_fb,
    modelsuite = cypmodels3p, data = cypfb_v1, sparse_output = TRUE)
summary(cypmatrix3fp)
>
> This historical lefkoMat object contains 15 matrices.
>
> Each matrix is square with }2916\mathrm{ rows and columns, and a total of }8503056\mathrm{ elements.
> A total of 1768620 survival transitions were estimated, with 117908 per matrix.
> A total of 35280 fecundity transitions were estimated, with }2352\mathrm{ per matrix.
> This lefkoMat object covers 1 population, 3 patches, and 5 time steps.
>
> The dataset contains a total of 74 unique individuals and 320 unique transitions.
>
> Vital rate modeling quality control:
>
> Survival estimated with 74 individuals and 320 individual transitions.
> Observation estimated with 70 individuals and 303 individual transitions.
Primary size estimated with 70 individuals and 288 individual transitions.
Secondary size transition not estimated.
Tertiary size transition not estimated.
Reproductive status estimated with 70 individuals and 288 individual transitions.
Fecundity estimated with 51 individuals and 118 individual transitions.
Juvenile survival not estimated.
Juvenile observation probability not estimated.
Juvenile primary size transition not estimated.
```

```
> Juvenile secondary size transition not estimated.
> Juvenile tertiary size transition not estimated.
> Juvenile reproduction probability not estimated.
> Juvenile maturity transition probability not estimated.
>
> Survival probability sum check (each matrix represented by column in order):
> [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11] [,12]
> Min. 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000
> 1st Qu. 0.965 0.965 0.965 0.965 0.965 0.964 0.965 0.965 0.965 0.964 0.964 0.965
> Median 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000
> Mean 0.820 0.820 0.820 0.820 0.820 0.820 0.820 0.820 0.820 0.820 0.820 0.820
> 3rd Qu. 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000
> Max. 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000
> [,13] [,14] [,15]
> Min. 0.000 0.000 0.000
> 1st Qu. 0.965 0.964 0.964
> Median 1.000 1.000 1.000
> Mean 0.820 0.820 0.820
> 3rd Qu. 1.000 1.000 1.000
> Max. 1.000 1.000 1.000
```

Note the similarities and differences with the ahistorical MPMs. First, we see 15 matrices produced again. However, in the raw hMPM, we only saw 12 matrices because of the fact that we need sets of three consecutive monitoring occasions of data to parameterize each raw transition (six monitoring occasions total means four sets of three monitoring occasions, or four total time steps, in the historical case). Since this is a function-based MPM, we can use our functions to estimate transitions in the first year even without a full set of three years of data. The result is five time steps that we can estimate transitions for.

Second, these matrices are utterly huge. They have $54^{2}=2916$ rows and columns, yielding $2916^{2}=$ $8,503,056$ total elements per matrix. And we have 120,260 elements estimated per matrix. This is vastly more than in the ahistorical case. However, it is also just a small fraction of the total number of elements in the matrix. In fact, in an unreduced hMPM estimated in Ehrlén format, the total number of elements that can be estimated is equal to $\frac{r \times c}{m}$, where $r$ and $c$ are the numbers of rows and columns, respectively, and $m$ is the number of stages in the stageframe. For example, if there are 10 stages in the stageframe, then only 1000 of the total 10,000 elements in the hMPM are estimable $(10 \%$ of the elements).

We can also make deVries format hMPMs. Here, we just add a single option regarding format. The resulting summary shows bigger matrices reflecting the addition of the prior stage for newborns.

```
cypmatrix3fp_deV <- flefko3(stageframe = cypframe_fb, supplement = cypsupp3_fb,
    modelsuite = cypmodels3p, data = cypfb_v1, format = "deVries",
    sparse_output = TRUE)
summary(cypmatrix3fp_deV)
>
> This historical lefkoMat object contains 15 matrices.
>
> Each matrix is square with 2970 rows and columns, and a total of 8820900 elements.
> A total of 1767900 survival transitions were estimated, with }117860\mathrm{ per matrix.
> A total of 35280 fecundity transitions were estimated, with }2352\mathrm{ per matrix.
> This lefkoMat object covers 1 population, 3 patches, and 5 time steps.
>
```

```
> The dataset contains a total of 74 unique individuals and 320 unique transitions.
>
> Vital rate modeling quality control:
>
> Survival estimated with 74 individuals and 320 individual transitions.
> Observation estimated with 70 individuals and 303 individual transitions.
> Primary size estimated with 70 individuals and 288 individual transitions.
> Secondary size transition not estimated.
> Tertiary size transition not estimated.
> Reproductive status estimated with 70 individuals and 288 individual transitions.
> Fecundity estimated with 51 individuals and }118\mathrm{ individual transitions.
> Juvenile survival not estimated.
> Juvenile observation probability not estimated.
> Juvenile primary size transition not estimated.
> Juvenile secondary size transition not estimated.
> Juvenile tertiary size transition not estimated.
> Juvenile reproduction probability not estimated.
> Juvenile maturity transition probability not estimated.
>
> Survival probability sum check (each matrix represented by column in order):
> [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11] [,12]
> Min. 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000
> 1st Qu. 0.965 0.965 0.965 0.965 0.965 0.964 0.965 0.965 0.965 0.964 0.964 0.965
> Median 1.000 1.000 1.000 1.000 1.000 0.999 1.000 1.000 1.000 0.999 0.999 1.000
> Mean 0.804 0.804 0.804 0.804 0.803 0.803 0.804 0.804 0.804 0.803 0.803 0.804
> 3rd Qu. 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000
> Max. 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000
> [,13] [,14] [,15]
> Min. 0.000 0.000 0.000
> 1st Qu. 0.965 0.964 0.964
> Median 0.999 0.999 0.999
> Mean 0.804 0.803 0.803
> 3rd Qu. 1.000 1.000 1.000
> Max. 1.000 1.000 1.000
```

Now that we have created our MPMs, we might wish to create element-wise arithmetic mean matrices to aid inference and further analysis. For example, we might be interested in developing patch-level means and an overall population mean, but one in which the element means treat each patch and each year as equal in proportional effect. For this purpose, we can use the lmean() function.

```
cyp2fp_mean <- lmean(cypmatrix2fp)
cyp3fp_mean <- lmean(cypmatrix3fp)
summary(cyp2fp_mean)
>
> This ahistorical lefkoMat object contains 4 matrices.
>
> Each matrix is square with }54\mathrm{ rows and columns, and a total of 2916 elements.
> A total of 9644 survival transitions were estimated, with }2411\mathrm{ per matrix.
> A total of }192\mathrm{ fecundity transitions were estimated, with 48 per matrix.
> This lefkoMat object covers 1 population, 4 patches, and 0 time steps.
>
```

```
> The dataset contains a total of }74\mathrm{ unique individuals and 320 unique transitions.
>
> Vital rate modeling quality control:
>
> Survival estimated with 74 individuals and 320 individual transitions.
> Observation estimated with 70 individuals and 303 individual transitions.
> Primary size estimated with 70 individuals and 288 individual transitions.
> Secondary size transition not estimated.
> Tertiary size transition not estimated.
> Reproductive status estimated with 70 individuals and 288 individual transitions.
> Fecundity estimated with 51 individuals and }118\mathrm{ individual transitions.
> Juvenile survival not estimated.
> Juvenile observation probability not estimated.
> Juvenile primary size transition not estimated.
> Juvenile secondary size transition not estimated.
> Juvenile tertiary size transition not estimated.
> Juvenile reproduction probability not estimated.
> Juvenile maturity transition probability not estimated.
>
> Survival probability sum check (each matrix represented by column in order):
> [,1] [,2] [,3] [,4]
> Min. 0.050 0.050 0.050 0.050
> 1st Qu. 0.991 0.991 0.991 0.991
> Median 1.000 1.000 1.000 1.000
> Mean 0.916 0.916 0.915 0.916
> 3rd Qu. 1.000 1.000 1.000 1.000
> Max. 1.000 1.000 1.000 1.000
summary(cyp3fp_mean)
>
> This historical lefkoMat object contains 4 matrices.
>
> Each matrix is square with 2916 rows and columns, and a total of 8503056 elements.
> A total of 471632 survival transitions were estimated, with }117908\mathrm{ per matrix.
> A total of 9408 fecundity transitions were estimated, with }2352\mathrm{ per matrix.
> This lefkoMat object covers 1 population, 4 patches, and O time steps.
>
> The dataset contains a total of }74\mathrm{ unique individuals and 320 unique transitions.
>
> Vital rate modeling quality control:
>
> Survival estimated with 74 individuals and 320 individual transitions.
> Observation estimated with 70 individuals and 303 individual transitions.
> Primary size estimated with 70 individuals and 288 individual transitions.
> Secondary size transition not estimated.
> Tertiary size transition not estimated.
> Reproductive status estimated with 70 individuals and 288 individual transitions.
> Fecundity estimated with 51 individuals and 118 individual transitions.
Juvenile survival not estimated.
Juvenile observation probability not estimated.
Juvenile primary size transition not estimated.
Juvenile secondary size transition not estimated.
```

```
> Juvenile tertiary size transition not estimated.
> Juvenile reproduction probability not estimated.
> Juvenile maturity transition probability not estimated.
>
> Survival probability sum check (each matrix represented by column in order):
> [,1] [,2] [,3] [,4]
> Min. 0.000 0.000 0.000 0.000
> 1st Qu. 0.965 0.965 0.964 0.965
> Median 1.000 1.000 1.000 1.000
> Mean 0.820 0.820 0.820 0.820
> 3rd Qu. 1.000 1.000 1.000 1.000
> Max. 1.000 1.000 1.000 1.000
```

The function-based mean matrices have the same number of estimated transitions as their constituent matrices, because all elements that are estimable are actually estimated using the vital rate models supplied, meaning that each matrix used per mean has the same number of estimated elements. We have four mean matrices in each case - the first three are patch-level means for each of three patches, and the final matrix in each case is the population mean matrix. The labels element in each lefkoMat object shows this order.

```
cyp2fp_mean$labels
> pop patch
> 1 1 A
> 2 1 B
> 3 1 C
>4 1 0
```

In the preceding vital rate model sets, we created best-fit models that necessarily included a patch term. This will result necessarily in the creation of matrices at the patch level. To create matrices for the population level only, we remove the patch option from the preceding function calls, as below.

```
cypmodels3 <- modelsearch(cypfb_v1, historical = TRUE, approach = "mixed",
    vitalrates = c("surv", "obs", "size", "repst", "fec"),
    sizedist = "negbin", size.trunc = TRUE, fecdist = "poisson", fec.zero = TRUE,
    suite = "main", size = c("size3added", "size2added", "size1added"),
    quiet = "partial")
cypmodels2 <- modelsearch(cypfb_v1, historical = FALSE, approach = "mixed",
    vitalrates = c("surv", "obs", "size", "repst", "fec"),
    sizedist = "negbin", size.trunc = TRUE, fecdist = "poisson", fec.zero = TRUE,
    suite = "main", size = c("size3added", "size2added"),
    quiet = "partial")
```

Let's see the model suite summaries. First the historical set.

```
summary(cypmodels3)
> This LefkoMod object includes 5 linear models.
> Best-fit model criterion used: aicc&k
>
>
>
> Survival model:
```

```
Generalized linear mixed model fit by maximum likelihood (Laplace
    Approximation) [glmerMod]
    Family: binomial ( logit )
    Formula: alive3 ~ size2added + (1 | year2) + (1 | individ)
        Data: subdata
            AIC BIC logLik deviance df.resid
128.1324 143.2057-60.0662 120.1324 316
Random effects:
Groups Name Std.Dev.
individ (Intercept) 1.198381
year2 (Intercept) 0.008826
Number of obs: 320, groups: individ, 74; year2, 5
Fixed Effects:
(Intercept) size2added
> 2.0352 0.6344
optimizer (Nelder_Mead) convergence code: O (OK) ; O optimizer warnings; 1 lme4 warnings
>
>
Observation model:
Generalized linear mixed model fit by maximum likelihood (Laplace
    Approximation) [glmerMod]
    Family: binomial ( logit )
Formula: obsstatus3 ~ size2added + (1 | year2) + (1 | individ)
            Data: subdata
            AIC BIC logLik deviance df.resid
118.2567 133.1117 -55.1284 110.2567 299
Random effects:
    Groups Name Std.Dev.
    individ (Intercept) 1.078e-05
    year2 (Intercept) 8.776e-01
    Number of obs: 303, groups: individ, 70; year2, 5
    Fixed Effects:
    (Intercept) size2added
        2.4904 0.3134
optimizer (Nelder_Mead) convergence code: O (OK) ; O optimizer warnings; 1 lme4 warnings
>
>
>
Size model:
Formula: size3added ~ (1 | year2) + (1 | individ)
Data: subdata
AIC BIC logLik df.resid
1008.2763 1022.9282-500.1382 284
Random-effects (co)variances:
Conditional model:
    Groups Name Std.Dev.
    year2 (Intercept) 0.1109
    individ (Intercept) 1.0561
```

```
> Number of obs: 288 / Conditional model: year2, 5; individ, 70
>
> Dispersion parameter for truncated_nbinom2 family (): 2.59e+07
>
Fixed Effects:
>
Conditional model:
> (Intercept)
> 0.5761
>
>
>
Secondary size model:
[1] 1
>
>
>
> Tertiary size model:
[1] 1
>
>
>
Reproductive status model:
Generalized linear mixed model fit by maximum likelihood (Laplace
        Approximation) [glmerMod]
    Family: binomial ( logit )
Formula: repstatus3 ~ repstatus2 + size2added + (1 | year2) + (1 | individ)
            Data: subdata
            AIC BIC logLik deviance df.resid
    333.6176 351.9324 -161.8088 323.6176 283
Random effects:
Groups Name Std.Dev.
individ (Intercept) 0.1829
year2 (Intercept) 0.6250
Number of obs: 288, groups: individ, 70; year2, 5
Fixed Effects:
(Intercept) repstatus2 size2added
    -1.4630 1.6457 0.1715
>
>
>
Fecundity model:
Formula: feca2 ~ size2added + (1 | year2) + (1 | individ)
Zero inflation: ~size2added + (1 | year2) + (1 | individ)
Data: subdata
    AIC BIC logLik df.resid
248.8609 271.0264-116.4305 110
Random-effects (co)variances:
Conditional model:
Groups Name Std.Dev.
```

```
year2 (Intercept) 0.5760
individ (Intercept) 0.1639
Zero-inflation model:
Groups Name Std.Dev.
year2 (Intercept) 1.642e-06
individ (Intercept) 3.089e-04
Number of obs: 118 / Conditional model: year2, 5; individ, 51 / Zero-inflation model: year2, 5; inc
Fixed Effects:
Conditional model:
(Intercept) size2added
-0.54014 0.06174
Zero-inflation model:
(Intercept) size2added
    3.865 -1.574
Juvenile survival model:
[1] 1
> Juvenile observation model:
[1] 1
Juvenile size model:
[1] 1
>
>
> Juvenile secondary size model:
[1] 1
>
>
> Juvenile tertiary size model:
[1] 1
>
> Juvenile reproduction model:
[1] 1
>
```

$>$
$>$
$>$
$>$
$>$
$>$
$>$
$>$
$>$

```
Juvenile maturity model:
[1] 1
>
>
>
>
>
> Number of models in survival table: 16
Number of models in observation table: 16
Number of models in size table: 16
Number of models in secondary size table: 1
Number of models in tertiary size table: 1
Number of models in reproduction status table: 16
Number of models in fecundity table: 238
Number of models in juvenile survival table: 1
Number of models in juvenile observation table: 1
Number of models in juvenile size table: 1
Number of models in juvenile secondary size table: 1
Number of models in juvenile tertiary size table: 1
Number of models in juvenile reproduction table: 1
Number of models in juvenile maturity table: 1
General model parameter names (column 1), and
specific names used in these models (column 2):
            parameter_names mainparams
```

> 9 reproductive status in time $t+1$ repst3

```
> 10 fecundity in time t+1 fec3
> 11 fecundity in time t fec2
> 12 sizea in time t size2
> 13 sizea in time t-1 size1
> 14 sizeb in time t sizeb2
> 15 sizeb in time t-1 sizeb1
16 sizec in time t sizec2
        sizec in time t-1 sizec1
        reproductive status in time t repst2
        reproductive status in time t-1 repst1
        maturity status in time t+1 matst3
            maturity status in time t matst2
                age in time t age
            density in time t density
        individual covariate a in time t indcova2
        individual covariate a in time t-1 indcova1
        individual covariate b in time t indcovb2
        individual covariate b in time t-1 indcovb1
        individual covariate c in time t indcovc2
        individual covariate c in time t-1 indcovc1
            stage group in time t group2
            stage group in time t-1 group1
>
>
>
>
>
> Quality control:
>
> Survival model estimated with 74 individuals and 320 individual transitions.
> Survival model accuracy is 0.947.
> Observation status model estimated with 70 individuals and 303 individual transitions.
> Observation status model accuracy is 0.95.
> Primary size model estimated with 70 individuals and 288 individual transitions.
> Primary size model R-squared is 0.822.
> Secondary size model not estimated.
> Tertiary size model not estimated.
> Reproductive status model estimated with 70 individuals and 288 individual transitions.
> Reproductive status model accuracy is 0.715.
> Fecundity model estimated with }51\mathrm{ individuals and }118\mathrm{ individual transitions.
> Fecundity model R-squared is 0.562.
> Juvenile survival model not estimated.
> Juvenile observation status model not estimated.
> Juvenile primary size model not estimated.
> Juvenile secondary size model not estimated.
> Juvenile tertiary size model not estimated.
> Juvenile reproductive status model not estimated.
> Juvenile maturity status model not estimated.
```

Now the ahistorical set.

```
summary(cypmodels2)
> This LefkoMod object includes 5 linear models.
Best-fit model criterion used: aicc&k
>
>
>
Survival model:
Generalized linear mixed model fit by maximum likelihood (Laplace
    Approximation) [glmerMod]
    Family: binomial ( logit )
Formula: alive3 ~ size2added + (1 | year2) + (1 | individ)
            Data: subdata
            AIC BIC logLik deviance df.resid
128.1324 143.2057-60.0662 120.1324 316
Random effects:
    Groups Name Std.Dev.
    individ (Intercept) 1.198381
    year2 (Intercept) 0.008826
Number of obs: 320, groups: individ, 74; year2, 5
Fixed Effects:
(Intercept) size2added
    2.0352 0.6344
optimizer (Nelder_Mead) convergence code: O (OK) ; O optimizer warnings; 1 lme4 warnings
Observation model:
Generalized linear mixed model fit by maximum likelihood (Laplace
    Approximation) [glmerMod]
    Family: binomial ( logit )
Formula: obsstatus3 ~ size2added + (1 | year2) + (1 | individ)
            Data: subdata
            AIC BIC logLik deviance df.resid
118.2567 133.1117 -55.1284 110.2567 299
Random effects:
Groups Name Std.Dev.
individ (Intercept) 1.078e-05
year2 (Intercept) 8.776e-01
Number of obs: 303, groups: individ, 70; year2, 5
Fixed Effects:
(Intercept) size2added
    2.4904 0.3134
optimizer (Nelder_Mead) convergence code: O (OK) ; O optimizer warnings; 1 lme4 warnings
>
>
>
Size model:
Formula: size3added ~ (1 | year2) + (1 | individ)
Data: subdata
    AIC BIC logLik df.resid
1008.2763 1022.9282-500.1382 284
```

```
Random-effects (co)variances:
Conditional model:
Groups Name Std.Dev.
    year2 (Intercept) 0.1109
    individ (Intercept) 1.0561
Number of obs: 288 / Conditional model: year2, 5; individ, 70
Dispersion parameter for truncated_nbinom2 family (): 2.59e+07
Fixed Effects:
Conditional model:
(Intercept)
> 0.5761
>
>
>
Secondary size model:
[1] 1
>
>
>
Tertiary size model:
[1] 1
>
>
>
Reproductive status model:
Generalized linear mixed model fit by maximum likelihood (Laplace
        Approximation) [glmerMod]
    Family: binomial ( logit )
Formula: repstatus3 ~ repstatus2 + size2added + (1 | year2) + (1 | individ)
            Data: subdata
                AIC BIC logLik deviance df.resid
    333.6176 351.9324 -161.8088 323.6176 283
Random effects:
    Groups Name Std.Dev.
    individ (Intercept) 0.1829
    year2 (Intercept) 0.6250
Number of obs: 288, groups: individ, 70; year2, 5
Fixed Effects:
(Intercept) repstatus2 size2added
            -1.4630 1.6457 0.1715
>
>
>
> Fecundity model:
Formula: feca2 ~ size2added + (1 | year2) + (1 | individ)
Zero inflation: ~size2added + (1 | year2) + (1 | individ)
```

```
> Data: subdata
> AIC BIC logLik df.resid
248.8609 271.0264-116.4305 110
Random-effects (co)variances:
>
> Conditional model:
> Groups Name Std.Dev.
> year2 (Intercept) 0.5760
individ (Intercept) 0.1639
>
> Zero-inflation model:
> Groups Name Std.Dev.
> year2 (Intercept) 1.642e-06
individ (Intercept) 3.089e-04
>
Number of obs: 118 / Conditional model: year2, 5; individ, 51 / Zero-inflation model: year2, 5; ind
Fixed Effects:
>
> Conditional model:
> (Intercept) size2added
        -0.54014 0.06174
Zero-inflation model:
(Intercept) size2added
3.865 -1.574
>
>
> Juvenile survival model:
[1] 1
>
>
>
> Juvenile observation model:
[1] 1
>
>
>
> Juvenile size model:
[1] 1
>
>
>
> Juvenile secondary size model:
[1] 1
>
>
>
> Juvenile tertiary size model:
[1] 1
>
```

```
>
>
Juvenile reproduction model:
[1] 1
>
>
>
Juvenile maturity model:
[1] 1
>
>
>
>
>
Number of models in survival table: 4
Number of models in observation table: 4
Number of models in size table: 4
Number of models in secondary size table: 1
Number of models in tertiary size table: 1
Number of models in reproduction status table: 4
Number of models in fecundity table: 15
Number of models in juvenile survival table: 1
Number of models in juvenile observation table: 1
Number of models in juvenile size table: 1
Number of models in juvenile secondary size table: 1
Number of models in juvenile tertiary size table: 1
Number of models in juvenile reproduction table: 1
Number of models in juvenile maturity table: 1
>
>
>
>
> General model parameter names (column 1), and
specific names used in these models (column 2):
parameter_names mainparams
                                    time t year2
                                    individual individ
```

```
> 3
>4
> 5
> 6
>}
```



```
reproductive status in time t+1
> 10 fecundity in time t+1 fec3
fecundity in time t fec2
12
> 13
> 14
> 15
> 16
    > 17
    > 18
    > }1
    > 20
    > 21
    > 22
    > 23
    24
    individual covariate a in time t-1 indcova1
    individual covariate b in time t indcovb2
    individual covariate b in time t-1 indcovb1
        individual covariate c in time t indcovc2
        individual covariate c in time t-1 indcovc1
            stage group in time t group2
            stage group in time t-1 group1
>
>
>
>
>
> Quality control:
>
> Survival model estimated with 74 individuals and 320 individual transitions.
> Survival model accuracy is 0.947.
> Observation status model estimated with 70 individuals and 303 individual transitions.
> Observation status model accuracy is 0.95.
> Primary size model estimated with 70 individuals and 288 individual transitions.
> Primary size model R-squared is 0.822.
> Secondary size model not estimated.
> Tertiary size model not estimated.
> Reproductive status model estimated with 70 individuals and 288 individual transitions.
> Reproductive status model accuracy is 0.715.
> Fecundity model estimated with 51 individuals and 118 individual transitions.
> Fecundity model R-squared is 0.562.
> Juvenile survival model not estimated.
> Juvenile observation status model not estimated.
> Juvenile primary size model not estimated.
```

```
> Juvenile secondary size model not estimated.
> Juvenile tertiary size model not estimated.
> Juvenile reproductive status model not estimated.
> Juvenile maturity status model not estimated.
```

A close look at the model output shows that the best-fit models have changed in structure. Particularly, patch is no longer included - it had been included previously as a random factor in all models. The lack of patch as a random factor has also led to many other changes in the best-fit models. Let's use these models to create new MPMs. First the ahistorical MPM.

```
cypmatrix2f <- flefko2(stageframe = cypframe_fb, supplement = cypsupp2_fb,
    modelsuite = cypmodels2, data = cypfb_v1)
summary(cypmatrix2f)
>
> This ahistorical lefkoMat object contains 5 matrices.
>
> Each matrix is square with }54\mathrm{ rows and columns, and a total of 2916 elements.
> A total of }12055\mathrm{ survival transitions were estimated, with }2411\mathrm{ per matrix.
> A total of 240 fecundity transitions were estimated, with 48 per matrix.
> This lefkoMat object covers 1 population, 1 patch, and 5 time steps.
>
> The dataset contains a total of }74\mathrm{ unique individuals and 320 unique transitions.
>
> Vital rate modeling quality control:
>
> Survival estimated with 74 individuals and 320 individual transitions.
> Observation estimated with 70 individuals and 303 individual transitions.
> Primary size estimated with 70 individuals and 288 individual transitions.
> Secondary size transition not estimated.
> Tertiary size transition not estimated.
> Reproductive status estimated with 70 individuals and 288 individual transitions.
> Fecundity estimated with 51 individuals and 118 individual transitions.
> Juvenile survival not estimated.
> Juvenile observation probability not estimated.
> Juvenile primary size transition not estimated.
> Juvenile secondary size transition not estimated.
> Juvenile tertiary size transition not estimated.
> Juvenile reproduction probability not estimated.
> Juvenile maturity transition probability not estimated.
>
> Survival probability sum check (each matrix represented by column in order):
> [,1] [,2] [,3] [,4] [,5]
> Min. 0.050 0.050 0.050 0.050 0.050
> 1st Qu. 0.991 0.991 0.991 0.991 0.991
> Median 1.000 1.000 1.000 1.000 1.000
> Mean 0.916 0.917 0.917 0.917 0.914
> 3rd Qu. 1.000 1.000 1.000 1.000 1.000
> Max. 1.000 1.000 1.000 1.000 1.000
```

Now the historical MPM.

```
cypmatrix3f <- flefko3(stageframe = cypframe_fb, supplement = cypsupp3_fb,
    modelsuite = cypmodels3, data = cypfb_v1, sparse_output = TRUE)
summary(cypmatrix3f)
>
> This historical lefkoMat object contains 5 matrices.
>
> Each matrix is square with }2916\mathrm{ rows and columns, and a total of }8503056\mathrm{ elements.
> A total of 589540 survival transitions were estimated, with }117908\mathrm{ per matrix.
> A total of }11760\mathrm{ fecundity transitions were estimated, with }2352\mathrm{ per matrix.
> This lefkoMat object covers 1 population, 1 patch, and 5 time steps.
>
> The dataset contains a total of }74\mathrm{ unique individuals and 320 unique transitions.
>
> Vital rate modeling quality control:
>
> Survival estimated with 74 individuals and 320 individual transitions.
> Observation estimated with 70 individuals and 303 individual transitions.
> Primary size estimated with 70 individuals and 288 individual transitions.
> Secondary size transition not estimated.
> Tertiary size transition not estimated.
> Reproductive status estimated with 70 individuals and 288 individual transitions.
> Fecundity estimated with 51 individuals and 118 individual transitions.
> Juvenile survival not estimated.
> Juvenile observation probability not estimated.
> Juvenile primary size transition not estimated.
> Juvenile secondary size transition not estimated.
> Juvenile tertiary size transition not estimated.
> Juvenile reproduction probability not estimated.
> Juvenile maturity transition probability not estimated.
>
> Survival probability sum check (each matrix represented by column in order):
> [,1] [,2] [,3] [,4] [,5]
> Min. 0.000 0.000 0.000 0.000 0.000
> 1st Qu. 0.965 0.965 0.965 0.965 0.965
> Median 1.000 1.000 1.000 1.000 1.000
> Mean 0.820 0.820 0.820 0.820 0.820
> 3rd Qu. 1.000 1.000 1.000 1.000 1.000
> Max. 1.000 1.000 1.000 1.000 1.000
```

We see that the main difference here is that there are only five matrices in each MPM, since the matrices are now annual matrices representing the whole population rather than each patch. The elements are NOT simple averages of the patch matrices. Since we are not differentiating between individuals in different patches, patches with greater numbers of individuals have a greater impact on the population-level matrices that we now created. So, these matrices are fundamentally different from those of the previous case.

### 5.6 Using stage groups for complex MPMs

Let's imagine a more complex life history. For example, imagine that our study organism had a life history in which the juvenile morphology inevitably led to stages that were non-reproductive but otherwise just like the reproductive adult stages, and that once the organism became reproductive it
stayed so until death. The resulting life history model might look like this (figure 5.3).


Figure 5.3: Imaginary life history model with permanent transitions from non-reproductive to reproductive adult stages

We can alter our code to account for this subtle but nonetheless major change with just two steps. First, we will add stage groups to our stage frame. We will assign the dormant seed stage, three protocorm stages, and seedling stage to a juvenile stage group; the dormant and non-reproductive adult stages to an adult stage group; and the reproductive adult stages to a second adult stage group. This only requires adding one further vector coding these groups. Please bear in mind that stage groups must be numbered sequentially, and the first stage group must be group 0 .

```
groupvec <- c(rep(0, 5), rep(1, 25), rep(2, 24))
cypframe_fb_group <- sf_create(sizes = sizevector, stagenames = stagevector,
    repstatus = repvector, obsstatus = obsvector, matstatus = matvector,
    propstatus = propvector, immstatus = immvector, indataset = indataset,
    group = groupvec, comments = comments)
cypframe_fb_group
> stage size size_b size_c min_age max_age repstatus obsstatus propstatus
>1 SD 0 NA NA NA NA NA
>2 P1 P1 0 NA NA NA NA NA 
>3 3
>4 
> 5 NL 
> 7 V1 1 NA NA NA NA NA N
> 8 N-N2 
> 9 V3 3 3 NA 
```





```
\begin{tabular}{|c|c|c|c|}
\hline > 23 & NA & NA & 1 Veg adult 17 stems \\
\hline > 24 & NA & NA & 1 Veg adult 18 stems \\
\hline > 25 & NA & NA & 1 Veg adult 19 stems \\
\hline > 26 & NA & NA & 1 Veg adult 20 stems \\
\hline > 27 & NA & NA & 1 Veg adult 21 stems \\
\hline > 28 & NA & NA & 1 Veg adult 22 stems \\
\hline > 29 & NA & NA & 1 Veg adult 23 stems \\
\hline > 30 & NA & NA & 1 Veg adult 24 stems \\
\hline > 31 & NA & NA & 2 Flo adult 1 stem \\
\hline > 32 & NA & NA & 2 Flo adult 2 stems \\
\hline > 33 & NA & NA & 2 Flo adult 3 stems \\
\hline > 34 & NA & NA & 2 Flo adult 4 stems \\
\hline
\end{tabular}
> [ reached 'max' / getOption("max.print") -- omitted 20 rows ]
```

Next, we will add an extra line to the supplement table to set transitions from group 2 to group 1 to a value of 0.0 . This effectively disallows transitions back from reproductive to non-reproductive adult stages. We will do this with our ahistorical supplement table and create only ahistorical MPMs, but this may also be done with the historical supplement table to create historical MPMs.

```
cypsupp2_fb_group <- supplemental(stage3 = c("SD", "P1", "P2", "P3", "SL", "SL",
    "D", "V1", "V2", "V3", "SD", "P1", "group1"),
    stage2 = c("SD", "SD", "P1", "P2", "P3", "SL", "SL", "SL", "SL", "SL", "rep",
    "rep", "group2"),
    eststage3 = c(NA, NA, NA, NA, NA, NA, "D", "V1", "V2", "V3", NA, NA, NA),
    eststage2 = c(NA, NA, NA, NA, NA, NA, "D", "D", "D", "D", NA, NA, NA),
    givenrate = c(0.08, 0.1, 0.1, 0.1, 0.05, 0.05, NA, NA, NA, NA, NA, NA, 0.0),
    multiplier = c(NA, NA, NA, NA, NA, NA, sl_mult, sl_mult, sl_mult, sl_mult,
        0.5 * seeds_per_fruit, 0.5 * seeds_per_fruit, NA),
    type = c("S", "S", "S", "S", "S", "S", "S", "S", "S", "S", "R", "R", "S"),
    stageframe = cypframe_fb_group, historical = FALSE)
```

cypsupp2_fb_group


| $>$ | 6 | NA | 1 |
| ---: | ---: | ---: | ---: |
| $>7$ | 0.7 | 1 | 1 |
| $>8$ | 0.7 | 1 | 1 |
| $>9$ | 0.7 | 1 | 1 |
| $>10$ | 0.7 | 1 | 1 |
| $>11$ | 2500.0 | 3 | 1 |
| $>12$ | 2500.0 | 3 | 1 |
| $>13$ | NA | 1 | 1 |
| $>$ |  |  | 1 |

Finally, let's create our new ahistorical MPM, and take a look at the first matrix.

| cypmatrix2fp_group\$A[[1]] |  |  |  |  |  |  |  |  |  |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| > |  | [,1] | [,2] | [,3] | [,4] | [,5] | [,6] | [,7] | [,8] |
| > | [1,] | 0.08 | 0.0 | 0.0 | 0.00 | 0.00000000 | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |
| > | [2,] | 0.10 | 0.0 | 0.0 | 0.00 | 0.00000000 | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |
| $>$ | [3,] | 0.00 | 0.1 | 0.0 | 0.00 | 0.00000000 | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |
| > | [4, ] | 0.00 | 0.0 | 0.1 | 0.00 | 0.00000000 | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |
| > | [5,] | 0.00 | 0.0 | 0.0 | 0.05 | 0.05000000 | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |
| > | [6,] | 0.00 | 0.0 | 0.0 | 0.00 | 0.03291481 | $4.702115 \mathrm{e}-02$ | $3.686951 \mathrm{e}-02$ | $2.809355 \mathrm{e}-02$ |
|  | [7, ] | 0.00 | 0.0 | 0.0 | 0.00 | 0.16278272 | $2.325467 e-01$ | $2.383837 \mathrm{e}-01$ | $2.359754 \mathrm{e}-01$ |
|  | [8,] | 0.00 | 0.0 | 0.0 | 0.00 | 0.10351783 | $1.478826 \mathrm{e}-01$ | $1.515945 \mathrm{e}-01$ | $1.500630 \mathrm{e}-01$ |
| $>$ | [9,] | 0.00 | 0.0 | 0.0 | 0.00 | 0.06582972 | $9.404245 \mathrm{e}-02$ | 9.640294e-02 | 9.542901e-02 |
| > | [10,] | 0.00 | 0.0 | 0.0 | 0.00 | 0.00000000 | 5.980408e-02 | 6.130517e-02 | 6.068583e-02 |
|  | [11,] | 0.00 | 0.0 | 0.0 | 0.00 | 0.00000000 | 3.803099e-02 | 3.898558e-02 | $3.859172 \mathrm{e}-02$ |
|  | [12,] | 0.00 | 0.0 | 0.0 | 0.00 | 0.00000000 | $2.418491 \mathrm{e}-02$ | $2.479196 \mathrm{e}-02$ | $2.454149 \mathrm{e}-02$ |
|  | [13,] | 0.00 | 0.0 | 0.0 | 0.00 | 0.00000000 | $1.537982 \mathrm{e}-02$ | 1.576586e-02 | 1.560658e-02 |
|  | [14,] | 0.00 | 0.0 | 0.0 | 0.00 | 0.00000000 | 9.780435e-03 | 1.002593e-02 | 9.924637e-03 |
|  | [15,] | 0.00 | 0.0 | 0.0 | 0.00 | 0.00000000 | 6.219636e-03 | $6.375751 \mathrm{e}-03$ | 6.311339e-03 |
|  | [16,] | 0.00 | 0.0 | 0.0 | 0.00 | 0.00000000 | $3.955231 \mathrm{e}-03$ | $4.054508 \mathrm{e}-03$ | $4.013547 \mathrm{e}-03$ |
|  | [17,] | 0.00 | 0.0 | 0.0 | 0.00 | 0.00000000 | 2.515236e-03 | $2.578369 \mathrm{e}-03$ | $2.552320 \mathrm{e}-03$ |
|  | [18,] | 0.00 | 0.0 | 0.0 | 0.00 | 0.00000000 | 1.599505e-03 | $1.639653 \mathrm{e}-03$ | 1.623088e-03 |
|  |  |  |  | 9] |  | [,10] | [,11] | [,12] | [,13] |

$>[1] 0.000000 \mathrm{e}+,000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>[2] 0.000000 \mathrm{e}+,000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>[3] 0.000000 \mathrm{e}+,000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [4,] $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>[5] 0.000000 \mathrm{e}+,000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [6,] 2.104751e-02 1.561422e-02 1.151723e-02 8.466545e-03 6.211278e-03
> [7,] 2.281165e-01 2.167851e-01 2.032901e-01 1.885091e-01 1.730644e-01
> [8,] 1.450653e-01 1.378594e-01 1.292776e-01 1.198779e-01 1.100562e-01
$>$ [9,] 9.225087e-02 8.766840e-02 8.221101e-02 7.623353e-02 6.998766e-02
$>[10] 5.866476 \mathrm{e}-,025.575065 \mathrm{e}-025.228015 \mathrm{e}-024.847891 \mathrm{e}-024.450700 \mathrm{e}-02$
$>[11] 3.730647 \mathrm{e}-,023.545331 \mathrm{e}-023.324633 \mathrm{e}-023.082902 \mathrm{e}-022.830318 \mathrm{e}-02$
> $[12] 2.372417 \mathrm{e}-,022.254569 \mathrm{e}-022.114222 \mathrm{e}-021.960499 \mathrm{e}-021.799874 \mathrm{e}-02$
$>[13] 1.508682 \mathrm{e}-,021.433740 \mathrm{e}-021.344489 \mathrm{e}-021.246733 \mathrm{e}-021.144587 \mathrm{e}-02$
$>[14] 9.594109 \mathrm{e}-,039.117531 \mathrm{e}-038.549962 \mathrm{e}-037.928303 \mathrm{e}-037.278731 \mathrm{e}-03$
$>[15] 6.101147 \mathrm{e}-,035.798079 \mathrm{e}-035.437146 \mathrm{e}-035.041817 \mathrm{e}-034.628737 \mathrm{e}-03$
$>[16] 3.879880 \mathrm{e}-,033.687151 \mathrm{e}-033.457625 \mathrm{e}-033.206224 \mathrm{e}-032.943536 \mathrm{e}-03$
> [17,] 2.467318e-03 2.344757e-03 2.198795e-03 2.038923e-03 1.871872e-03
$>[18] 1.569033 \mathrm{e}-,031.491093 \mathrm{e}-031.398272 \mathrm{e}-031.296605 \mathrm{e}-031.190373 \mathrm{e}-03$
$>\quad[, 14] \quad[, 15] \quad[, 16] \quad[, 17] \quad[, 18]$
$>[1] 0.000000 \mathrm{e}+,000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>[2] 0.000000 \mathrm{e}+,000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>[3] 0.000000 \mathrm{e}+,000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>[4] 0.000000 \mathrm{e}+,000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [5,] $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [6,] 4.551060e-03 3.331976e-03 2.438204e-03 1.783578e-03 1.304415e-03
$>$ [7,] 1.574277e-01 1.419732e-01 1.270012e-01 1.127484e-01 9.939083e-02
$>$ [8,] 1.001124e-01 9.028447e-02 8.076344e-02 7.169967e-02 6.320525e-02
$>$ [9,] 6.366415e-02 5.741428e-02 5.135960e-02 4.559571e-02 4.019389e-02
$>[10] 4.048571 \mathrm{e}-,023.651125 \mathrm{e}-023.266092 \mathrm{e}-022.899552 \mathrm{e}-022.556035 \mathrm{e}-02$
$>$ [11,] 2.574593e-02 2.321847e-02 $2.076994 \mathrm{e}-021.843901 \mathrm{e}-021.625450 \mathrm{e}-02$
$>[12] 1.637252 \mathrm{e}-,021.476524 \mathrm{e}-021.320816 \mathrm{e}-02 \quad 1.172586 \mathrm{e}-02 \quad 1.033667 \mathrm{e}-02$
$>[13] 1.041172 \mathrm{e}-,029.389604 \mathrm{e}-038.399414 \mathrm{e}-03 \quad 7.456781 \mathrm{e}-036.573359 \mathrm{e}-03$
$>$ [14,] 6.621084e-03 5.971097e-03 5.341409e-03 4.741963e-03 4.180173e-03
$>[15] 4.210522 \mathrm{e}-,033.797178 \mathrm{e}-033.396743 \mathrm{e}-03 \quad 3.015540 \mathrm{e}-032.658282 \mathrm{e}-03$
$>[16] 2.677582 \mathrm{e}-,032.414726 \mathrm{e}-032.160078 \mathrm{e}-031.917661 \mathrm{e}-031.690472 \mathrm{e}-03$
$>[17] 1.702745 \mathrm{e}-,031.535588 \mathrm{e}-031.373651 \mathrm{e}-031.219491 \mathrm{e}-031.075016 \mathrm{e}-03$
$>[18] 1.082821 \mathrm{e}-,039.765207 \mathrm{e}-048.735408 \mathrm{e}-047.755067 \mathrm{e}-046.836307 \mathrm{e}-04$
$>\quad[, 19] \quad[, 20] \quad[, 21] \quad[, 22] \quad$ [,23]
$>$ [1,] $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [2,] $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [3,] $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [4,] $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>[5] 0.000000 \mathrm{e}+,000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [6,] 9.538315e-04 6.973975e-04 5.098657e-04 3.727414e-04 2.724849e-04
$>$ [7,] 8.704810e-02 7.578777e-02 6.563219e-02 5.656642e-02 4.854681e-02
$>$ [8,] 5.535618e-02 4.819544e-02 4.173724e-02 3.597208e-02 3.087219e-02
$>$ [9,] 3.520246e-02 3.064875e-02 2.654181e-02 2.287559e-02 1.963244e-02
$>[10] 2.238617 \mathrm{e}-,021.949035 \mathrm{e}-021.687864 \mathrm{e}-02 \quad 1.454719 \mathrm{e}-02 \quad 1.248479 \mathrm{e}-02$
$>[11] 1.423596 e-,021.239443 \mathrm{e}-021.073357 \mathrm{e}-02$ 9.250943e-03 7.939406e-03
$>$ [12,] 9.053020e-03 7.881944e-03 6.825761e-03 5.882919e-03 5.048878e-03
$>[13] 5.757054 \mathrm{e}-,035.012336 \mathrm{e}-034.340681 \mathrm{e}-033.741103 \mathrm{e}-033.210715 \mathrm{e}-03$
$>$ [14,] 3.661063e-03 3.187477e-03 2.760354e-03 2.379066e-03 2.041778e-03
$>[15] 2.328166 \mathrm{e}-,032.027001 \mathrm{e}-031.755382 \mathrm{e}-031.512911 \mathrm{e}-031.298421 \mathrm{e}-03$
$>[16] 1.480542 \mathrm{e}-,031.289023 \mathrm{e}-031.116294 \mathrm{e}-0319.621000 \mathrm{e}-048.256999 \mathrm{e}-04$
$>[17] 9.415160 \mathrm{e}-,048.197238 \mathrm{e}-047.098805 \mathrm{e}-04 \quad 6.118248 \mathrm{e}-04 \quad 5.250843 \mathrm{e}-04$
$>[18] 5.987348 \mathrm{e}-,045.212840 \mathrm{e}-044.514318 \mathrm{e}-043.890755 \mathrm{e}-043.339150 \mathrm{e}-04$
$>\quad[, 24] \quad[, 25] \quad[, 26] \quad[, 27] \quad[, 28]$
$>$ [1,] $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$ $>$ [2,] $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$ $>$ [3,] $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$ $>$ [4,] $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$ $>[5] 0.000000 \mathrm{e}+,000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$ $>$ [6,] 1.991889e-04 1.456059e-04 1.064355e-04 7.780170e-05 5.687069e-05 $>$ [7,] 4.150926e-02 3.537666e-02 3.006513e-02 2.548893e-02 2.156412e-02 $>$ [8,] 2.639683e-02 2.249695e-02 1.911921e-02 1.620909e-02 1.371319e-02
$>\quad[9] 1.678644 \mathrm{e}-,02 \quad 1.430640 \mathrm{e}-02 \quad 1.215841 \mathrm{e}-02 \quad 1.030778 \mathrm{e}-02 \quad 8.720581 \mathrm{e}-03$
$>[10] 1.067494 \mathrm{e}-,029.097820 \mathrm{e}-037.731854 \mathrm{e}-03 \quad 6.554992 \mathrm{e}-03 \quad 5.545647 \mathrm{e}-03$
$>[11] 6.788477 \mathrm{e}-,035.785544 \mathrm{e}-034.916890 \mathrm{e}-034.168493 \mathrm{e}-033.526623 \mathrm{e}-03$
[12,] $4.316971 \mathrm{e}-033.679180 \mathrm{e}-033.126780 \mathrm{e}-032.650854 \mathrm{e}-032.242673 \mathrm{e}-03$
[13,] 2.745276e-03 2.339688e-03 1.988402e-03 $1.685748 \mathrm{e}-03 \quad 1.426175 \mathrm{e}-03$
[14,] 1.745793e-03 $1.487869 \mathrm{e}-031.264477 \mathrm{e}-031.072012 \mathrm{e}-03 \quad 9.069422 \mathrm{e}-04$
[15,] 1.110196e-03 9.461752e-04 8.041144e-04 6.817206e-04 5.767485e-04
[16,] 7.060030e-04 6.016978e-04 5.113575e-04 4.335241e-04 3.667696e-04
[17,] 4.489659e-04 3.826355e-04 3.251857e-04 2.756894e-04 2.332384e-04
[18,] 2.855093e-04 2.433280e-04 2.067942e-04 1.753182e-04 1.483225e-04
$[, 29] \quad[, 30] \quad[, 31] \quad[, 33]$
$[1] 0.000000 \mathrm{e}+,000.000000 \mathrm{e}+002.262112 \mathrm{e}+031.650509 \mathrm{e}+037.104349 \mathrm{e}+02$
$[2] 0.000000 \mathrm{e}+,000.000000 \mathrm{e}+002.262112 \mathrm{e}+031.650509 \mathrm{e}+037.104349 \mathrm{e}+02$
$[3] \quad 0.000000 \mathrm{e}+,000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
[4,] $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$[5] \quad 0.000000 \mathrm{e}+,000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$[6] 4.157051 \mathrm{e}-,053.038649 \mathrm{e}-050.000000 \mathrm{e}+00 \quad 0.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
[7,] $1.821103 \mathrm{e}-021.535589 \mathrm{e}-020.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
[8,] $1.158088 \mathrm{e}-029.765213 \mathrm{e}-030.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
[9,] $7.364585 \mathrm{e}-036.209957 \mathrm{e}-030.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
[10,] 4.683334e-03 3.949075e-03 0.000000e+00 0.000000e+00 0.000000e+00
[11,] 2.978255e-03 2.511321e-03 $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$[12] 1.893951 \mathrm{e}-,031.597015 \mathrm{e}-030.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
[13,] 1.204414e-03 1.015584e-03 0.000000e+00 0.000000e+00 0.000000e+00
[14,] 7.659183e-04 6.458368e-04 0.000000e+00 0.000000e+00 0.000000e+00
$[15] 4.870677 \mathrm{e}-,044.107046 \mathrm{e}-04 \quad 0.000000 \mathrm{e}+00 \quad 0.000000 \mathrm{e}+00 \quad 0.000000 \mathrm{e}+00$
$[16] 3.097392 \mathrm{e}-,042.611779 \mathrm{e}-040.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
[17,] $1.969713 \mathrm{e}-041.660899 \mathrm{e}-04 \quad 0.000000 \mathrm{e}+00 \quad 0.000000 \mathrm{e}+00 \quad 0.000000 \mathrm{e}+00$
$[18] 1.252593 \mathrm{e}-,041.056210 \mathrm{e}-04 \quad 0.000000 \mathrm{e}+00 \quad 0.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$[, 34] \quad[, 35] \quad[, 36] \quad[, 38]$
$[1] 1.875698 \mathrm{e}+,024.075803 \mathrm{e}+018.437258 \mathrm{e}+00 \quad 1.728569 \mathrm{e}+00 \quad 3.533810 \mathrm{e}-01$
$[2] 1.875698 \mathrm{e}+,024.075803 \mathrm{e}+01 \quad 8.437258 \mathrm{e}+00 \quad 1.728569 \mathrm{e}+00 \quad 3.533810 \mathrm{e}-01$
[3,] $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [4,] $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>[5] 0.000000 \mathrm{e}+,000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>[6] 0.000000 \mathrm{e}+,000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [7,] $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>[8] 0.000000 \mathrm{e}+,000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>[9] 0.000000 \mathrm{e}+,000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>[10] 0.000000 \mathrm{e}+,000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>[11] 0.000000 \mathrm{e}+,000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>[12] 0.000000 \mathrm{e}+,000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
[13,] $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
[14,] $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
[15,] $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
[16,] $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [17,] $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>[18] 0.000000 \mathrm{e}+,000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
> [,39] [,40] [,41] [,42]
$>$ [1,] 7.221204e-02 1.475493e-02 3.014788e-03 6.159915e-04 1.258613e-04
$>$ [2,] $7.221204 \mathrm{e}-021.475493 \mathrm{e}-023.014788 \mathrm{e}-036.159915 \mathrm{e}-041.258613 \mathrm{e}-04$
$>[3] 0.000000 \mathrm{e}+,000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>[4] 0.000000 \mathrm{e}+,000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>[5] 0.000000 \mathrm{e}+,000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>[6] 0.000000 \mathrm{e}+,000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$ [7,] $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$ $[8] 0.000000 \mathrm{e}+,000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$ $[9] 0.000000 \mathrm{e}+,000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$ $[10] 0.000000 \mathrm{e}+,000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$ [11,] $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$ $[12] 0.000000 \mathrm{e}+,000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$ $[13] 0.000000 \mathrm{e}+,000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$ [14,] $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$ $[15] 0.000000 \mathrm{e}+,000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$ [16,] $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$ [17,] $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$ [18,] $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$ $[, 44] \quad[, 45] \quad[, 46] \quad[, 48]$ [1,] $2.571639 \mathrm{e}-05 \quad 5.254453 \mathrm{e}-06 \quad 1.073606 \mathrm{e}-06 \quad 2.193626 \mathrm{e}-07 \quad 4.482086 \mathrm{e}-08$ [2,] $2.571639 \mathrm{e}-055.254453 \mathrm{e}-061.073606 \mathrm{e}-06 \quad 2.193626 \mathrm{e}-07 \quad 4.482086 \mathrm{e}-08$
$>[3] 0.000000 \mathrm{e}+,000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>[4] 0.000000 \mathrm{e}+,000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [5,] $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>[6] 0.000000 \mathrm{e}+,000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [7,] $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [8,] $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>[9] 0.000000 \mathrm{e}+,000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>[10] 0.000000 \mathrm{e}+,000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>[11] 0.000000 \mathrm{e}+,000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>[12] 0.000000 \mathrm{e}+,000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>[13] 0.000000 \mathrm{e}+,000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>[14] 0.000000 \mathrm{e}+,000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>[15] 0.000000 \mathrm{e}+,000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>[16] 0.000000 \mathrm{e}+,000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
> [17,] $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>[18] 0.000000 \mathrm{e}+,000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
> [,49] [,50] [,51] [,52] [,53]
$>$ [1,] 9.157939e-09 1.871179e-09 3.823251e-10 7.811787e-11 1.596129e-11
$>$ [2,] 9.157939e-09 1.871179e-09 3.823251e-10 7.811787e-11 1.596129e-11
$>[3] 0.000000 \mathrm{e}+,000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>[4] 0.000000 \mathrm{e}+,000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>[5] 0.000000 \mathrm{e}+,000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>[6] 0.000000 \mathrm{e}+,000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [7,] $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [8,] $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [9,] $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [10,] $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>[11] 0.000000 \mathrm{e}+,000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>[12] 0.000000 \mathrm{e}+,000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>[13] 0.000000 \mathrm{e}+,000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>[14] 0.000000 \mathrm{e}+,000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>[15] 0.000000 \mathrm{e}+,000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>[16] 0.000000 \mathrm{e}+,000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>[17] 0.000000 \mathrm{e}+,000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>[18] 0.000000 \mathrm{e}+,000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$

```
> [,54]
> [1,] 3.261261e-12
> [2,] 3.261261e-12
> [3,] 0.000000e+00
> [4,] 0.000000e+00
> [5,] 0.000000e+00
> [6,] 0.000000e+00
> [7,] 0.000000e+00
> [8,] 0.000000e+00
> [9,] 0.000000e+00
> [10,] 0.000000e+00
> [11,] 0.000000e+00
> [12,] 0.000000e+00
> [13,] 0.000000e+00
> [14,] 0.000000e+00
> [15,] 0.000000e+00
> [16,] 0.000000e+00
> [17,] 0.000000e+00
> [18,] 0.000000e+00
> [ reached getOption("max.print") -- omitted 36 rows ]
```

Notice that a large portion of the entries are now zeros. We can also explore a summary of this MPM to see the reduction in estimated elements (the summary () function only counts elements as estimated if they are non-zero), particularly in comparison to the original ahistorical MPM.

```
summary (cypmatrix2fp_group)
>
> This ahistorical lefkoMat object contains 15 matrices.
>
> Each matrix is square with }54\mathrm{ rows and columns, and a total of 2916 elements.
> A total of 27165 survival transitions were estimated, with }1811\mathrm{ per matrix.
> A total of }720\mathrm{ fecundity transitions were estimated, with 48 per matrix.
> This lefkoMat object covers 1 population, 3 patches, and 5 time steps.
>
> The dataset contains a total of 74 unique individuals and 320 unique transitions.
>
> Vital rate modeling quality control:
>
> Survival estimated with 74 individuals and 320 individual transitions.
> Observation estimated with 70 individuals and 303 individual transitions.
> Primary size estimated with 70 individuals and 288 individual transitions.
Secondary size transition not estimated.
Tertiary size transition not estimated.
Reproductive status estimated with 70 individuals and 288 individual transitions.
Fecundity estimated with 51 individuals and 118 individual transitions.
Juvenile survival not estimated.
Juvenile observation probability not estimated.
Juvenile primary size transition not estimated.
Juvenile secondary size transition not estimated.
Juvenile tertiary size transition not estimated.
Juvenile reproduction probability not estimated.
Juvenile maturity transition probability not estimated.
```

```
>
> Survival probability sum check (each matrix represented by column in order):
> [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11] [,12]
>Min. 0.050 0.050 0.050 0.050 0.050 0.050 0.050 0.050 0.050 0.050 0.050 0.050
> 1st Qu. 0.879 0.743 0.802 0.867 0.927 0.840 0.670 0.741 0.824 0.900 0.888 0.786
> Median 0.979 0.956 0.965 0.978 0.988 0.971 0.940 0.956 0.969 0.983 0.981 0.964
> Mean 0.865 0.819 0.838 0.858 0.887 0.850 0.796 0.818 0.843 0.877 0.873 0.832
> 3rd Qu. 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000
> Max. 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000
> [,13] [,14] [,15]
> Min. 0.050 0.050 0.050
> 1st Qu. 0.838 0.885 0.936
> Median 0.971 0.981 0.990
> Mean 0.849 0.865 0.891
> 3rd Qu. 1.000 1.000 1.000
> Max. 1.000 1.000 1.000
summary(cypmatrix2fp)
>
> This ahistorical lefkoMat object contains 15 matrices.
>
> Each matrix is square with }54\mathrm{ rows and columns, and a total of 2916 elements.
> A total of 36165 survival transitions were estimated, with 2411 per matrix.
> A total of }720\mathrm{ fecundity transitions were estimated, with 48 per matrix.
> This lefkoMat object covers 1 population, 3 patches, and 5 time steps.
>
> The dataset contains a total of 74 unique individuals and 320 unique transitions.
>
> Vital rate modeling quality control:
>
> Survival estimated with 74 individuals and 320 individual transitions.
> Observation estimated with 70 individuals and 303 individual transitions.
> Primary size estimated with 70 individuals and 288 individual transitions.
> Secondary size transition not estimated.
> Tertiary size transition not estimated.
> Reproductive status estimated with 70 individuals and 288 individual transitions.
> Fecundity estimated with 51 individuals and }118\mathrm{ individual transitions.
> Juvenile survival not estimated.
> Juvenile observation probability not estimated.
> Juvenile primary size transition not estimated.
> Juvenile secondary size transition not estimated.
> Juvenile tertiary size transition not estimated.
> Juvenile reproduction probability not estimated.
> Juvenile maturity transition probability not estimated.
>
> Survival probability sum check (each matrix represented by column in order):
> [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11] [,12]
>Min. 0.050 0.050 0.050 0.050 0.050 0.050 0.050 0.050 0.050 0.050 0.050 0.050
> 1st Qu. 0.991 0.991 0.991 0.991 0.991 0.991 0.991 0.991 0.991 0.991 0.991 0.991
> Median 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000
> Mean 0.916 0.917 0.917 0.917 0.914 0.916 0.917 0.917 0.917 0.914 0.914 0.916
> 3rd Qu. 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000
```

```
> Max. 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000
> [,13] [,14] [,15]
> Min. 0.050 0.050 0.050
> 1st Qu. 0.991 0.991 0.991
> Median 1.000 1.000 1.000
> Mean 0.916 0.916 0.913
> 3rd Qu. 1.000 1.000 1.000
> Max. 1.000 1.000 1.000
```

We can see in the output that there are 600 fewer non-zero survival terms per matrix in our new ahistorical MPM. This is because we have eliminated the return transitions from the reproductive to the non-reproductive stages.

One important note here is that we used stage groups in this instance only to provide a convenient means of setting values later in the MPM itself. If we had wanted to test the impact of stage group on vital rates, then we would have needed to run function modelsearch() with option test.group = TRUE.

Using stage groups provides a great deal of power in structuring your MPMs. Now let's move on to the use of manually estimated vital rate models in function-based MPMs.

### 5.7 Environmental and individual covariates

Users may at times wish to develop MPMs conditioned on an extra variable, including environmental variables such as Spring precipitation, or a further individual state variable such as a frailty index. Package lefko3 can handle this easily with the modelsearch() function combined with the various function-based matrix estimators. Let's see an example of how this might work.

For our example, let's continue using the Cypripedium dataset with the life history model diagrammed in figure 5.1. We will see how total annual precipitation affects our vital rates, and use the resulting vital rates to create a new function-based MPM. First, we will need to add the proper variables to our original dataset.

```
cypdata_env <- cypdata
cypdata_env$prec.04 <- 92.2
cypdata_env$prec.05 <- 57.6
cypdata_env$prec.06 <- 96.0
cypdata_env$prec.07 <- 109.8
cypdata_env$prec.08 <- 111.9
cypdata_env$prec.09 <- 106.8
summary(cypdata_env)
> plantid patch X Y censor
> Min. : 164.0 A:23 Min. : 46.5 Min. : -28.00 Min. : 1
> 1st Qu.: 265.0 B:35 1st Qu.: 60.1 1st Qu.: 22.50 1st Qu.:1
> Median : 455.0 C:19 Median : 91.4 Median : 75.60 Median :1
> Mean : 669.1 Mean : 92.9 Mean : 55.54 Mean :1
> 3rd Qu.: 829.0 3rd Qu.:142.2 3rd Qu.: 80.10 3rd Qu.:1
> Max. :1560.0 Max. :173.0 Max. :142.40 Max. :1
>
> Inf2.04 Inf.04 Veg.04 Pod.04 Inf2.05
> Min. :0 Min. :0.0000 Min. : 0.000 Min. :0.0 Min. :0.00000
> 1st Qu.:0 1st Qu.:0.0000 1st Qu.: 1.000 1st Qu.:0.0 1st Qu.:0.00000
> Median :0 Median :0.0000 Median : 2.000 Median :0.0 Median :0.00000
```



The dataset is still formatted for the most part using blocks of variables for each year, except for the precipitation variables, which are at the very end of the dataset. We will standardize the dataset using verticalize3() using blocks, but inputting all precipitation variables by name (users wishing to use the block feature for precipitation must reorder the columns in the data frame).

```
cypfb_env <- verticalize3(data = cypdata_env, noyears = 6, firstyear = 2004,
    patchidcol = "patch", individcol = "plantid", blocksize = 4,
    sizeacol = "Inf2.04", sizebcol = "Inf.04", sizeccol = "Veg.04",
    repstracol = "Inf.04", repstrbcol = "Inf2.04", fecacol = "Pod.04",
    indcovacol = c("prec.04", "prec.05", "prec.06", "prec.07", "prec.08",
        "prec.09"), stageassign = cypframe_fb, stagesize = "sizeadded",
    NAasO = TRUE, age_offset = 4)
summary_hfv(cypfb_env)
>
> This hfv dataset contains }320\mathrm{ rows, }60\mathrm{ variables, 1 population,
> 3 patches, }74\mathrm{ individuals, and 5 time steps.
> rowid popid patchid individ year2
> Min. : 1.00 Length:320 A: 93 Min. : 164.0 Min. :2004
> 1st Qu.:21.00 Class :character B:154 1st Qu.: 391.0 1st Qu.:2005
> Median :37.50 Mode :character C: 73 Median : 453.0 Median :2006
> Mean :38.45
> 3rd Qu.:56.00
> Max. :77.00
            firstseen
    lastseen obsage obslifespan
> Min. :2004 Min. :2004 Min. :5.000 Min. :0.000
> 1st Qu.:2004 1st Qu.:2009 1st Qu.:6.000 1st Qu.:5.000
> Median :2004 Median :2009 Median :7.000 Median :5.000
> Mean :2004 Mean :2009 Mean :6.853 Mean :4.556
> 3rd Qu.:2004 3rd Qu.:2009 3rd Qu.:8.000 3rd Qu.:5.000
> Max. :2008 Max. :2009 Max. :9.000 Max. :5.000
> sizea1 sizeb1 sizec1 size1added
> Min. :0.000000 Min. : 0.0000 Min. : 0.0 Min. : 0.000
> 1st Qu.:0.000000 1st Qu.: 0.0000 1st Qu.: 0.0 1st Qu.: 0.000
> Median :0.000000 Median : 0.0000 Median : 1.0 Median : 2.000
> Mean :0.009375 Mean : 0.7469 Mean : 1.9 Mean : 2.656
> 3rd Qu.:0.000000 3rd Qu.: 1.0000 3rd Qu.: 3.0 3rd Qu.: 4.000
> Max. :1.000000 Max. :18.0000 Max. :13.0 Max. :21.000
> repstra1 repstrb1
Min. : 0.0000 Min. :0.000000
> 1st Qu.: 0.0000 1st Qu.:0.000000
> Median : 0.0000 Median :0.000000 Median : 0.0000 Median :0.0000
> Mean : 0.7469 Mean :0.009375 Mean : 0.7562 Mean :0.2656
> 3rd Qu.: 1.0000 3rd Qu.:0.000000 3rd Qu.: 1.0000 3rd Qu.:0.0000
> Max. :18.0000 Max. :1.000000 Max. :18.0000 Max. :7.0000
> fec1added
Min. :0.0000
1st Qu.:0.0000
Median :0.0000
Mean :0.2656
    3rd Qu.:0.0000 3rd Qu.: 96.00 3rd Qu.:1.0000 3rd Qu.:1.0000
    Min. : 0.00
    1st Qu.: 57.60
        Median : 92.20
        Mean : 70.64
```

| > | Max. :7.0000 <br> fecstatus1 | Max. :109.80 matstatus1 | Max. : 1.0000 alive1 | Max. : 1.0000 <br> stage1 |
| :---: | :---: | :---: | :---: | :---: |
| > | Min. 00.0000 | Min. 00.0000 | Min. 00.0000 | Length:320 |
| > | 1st Qu.:0.0000 | 1st Qu.:1.0000 | 1st Qu.:1.0000 | Class : character |
| > | Median :0.0000 | Median :1.0000 | Median :1.0000 | Mode :character |
| > | Mean :0.1344 | Mean :0.7688 | Mean :0.7688 |  |
| > | 3rd Qu.:0.0000 | 3rd Qu.:1.0000 | 3rd Qu.:1.0000 |  |
| > | ```Max. :1.0000 stage1index``` | $\begin{aligned} & \text { Max. }: 1.0000 \\ & \text { sizea2 } \end{aligned}$ | $\begin{gathered} \text { Max. } \quad: 1.0000 \\ \\ \text { sizeb2 } \end{gathered}$ | sizec2 |
| > | Min. : 0.00 | Min. $: 0.000000$ | Min. : 0.0000 | Min. : 0.000 |
| > | 1st Qu.: 6.00 | 1st Qu.:0.000000 | 1st Qu.: 0.0000 | 1st Qu.: 1.000 |
| > | Median : 8.00 | Median : 0.000000 | Median : 0.0000 | Median : 2.000 |
| > | Mean :14.17 | Mean :0.009375 | Mean : 0.8969 | Mean : 2.416 |
| > | 3rd Qu.:31.00 | 3rd Qu.:0.000000 | 3rd Qu.: 1.0000 | 3rd Qu.: 3.000 |
| > | $\begin{gathered} \text { Max. } \quad: 51.00 \\ \text { size2added } \end{gathered}$ |  | Max. :18.0000 repstrb2 | $\begin{aligned} & \text { Max. } \quad: 13.000 \\ & \text { repstr2added } \end{aligned}$ |
| > | Min. : 0.000 | Min. : 0.0000 | Min. $: 0.000000$ | 0 Min. : 0.0000 |
| > | 1st Qu.: 1.000 | 1st Qu.: 0.0000 | 1st Qu.:0.000000 | 1st Qu.: 0.0000 |
| > | Median : 2.000 | Median : 0.0000 | Median :0.000000 | 0 Median : 0.0000 |
| > | Mean : 3.322 | Mean : 0.8969 | Mean :0.009375 | 5 Mean : 0.9062 |
| > | 3rd Qu.: 4.000 | 3rd Qu.: 1.0000 | 3rd Qu.:0.000000 | 3rd Qu.: 1.0000 |
| > | Max. $: 24.000$ | Max. : 18.0000 | Max. 1.000000 | 0 Max. :18.0000 |
| > | feca2 | fec2added | indcova2 | obsstatus2 |
| > | Min. 0.0000 | Min. $\quad 0.0000$ | Min. : 57.60 | Min. 00.0000 |
| > | 1st Qu.:0.0000 | 1st Qu.:0.0000 | 1st Qu.: 92.20 | 1st Qu.:1.0000 |
| > | Median :0.0000 | Median :0.0000 | Median : 96.00 | Median :1.0000 |
| > | Mean :0.2906 | Mean :0.2906 | Mean : 92.77 | Mean :0.9531 |
| > | 3rd Qu.:0.0000 | 3rd Qu.:0.0000 | 3rd Qu.:109.80 | 3rd Qu.:1.0000 |
| > | Max. 7.0000 | Max. $\quad 7.0000$ | Max. : 111.90 | Max. 1.0000 |
| > | repstatus2 | fecstatus2 | matstatus2 | alive2 stage2 |
| > | Min. 00.0000 | Min. 00.0000 | Min. :1 Min. | :1 Length: 320 |
| > | 1st Qu.:0.0000 | 1st Qu.:0.0000 | 1st Qu.:1 1st | Qu.:1 Class :character |
| > | Median :0.0000 | Median :0.0000 | Median :1 Medi | ian :1 Mode :character |
| > | Mean :0.3688 | Mean :0.1562 | Mean :1 Mean | n :1 |
| > | 3rd Qu.:1.0000 | 3rd Qu.:0.0000 | 3rd Qu.:1 3rd | Qu. : 1 |
| > | Max. $: 1.0000$ | Max. : 1.0000 | Max. : 1 Max. | . 11 |
| > | stage2index | sizea3 | sizeb3 | sizec3 |
| > | Min. : 6.00 | Min. 00.000000 | Min. : 0.000 | Min. : 0.000 |
| > | 1st Qu.: 7.00 | 1st Qu.:0.000000 | 1st Qu.: 0.000 | 1st Qu.: 1.000 |
| > | Median :10.00 | Median :0.000000 | Median : 0.000 | Median : 1.000 |
| > | Mean :18.17 | Mean :0.009375 | Mean : 1.069 | Mean : 2.209 |
| > | 3rd Qu.:32.00 | 3rd Qu.:0.000000 | 3rd Qu.: 1.000 | 3rd Qu.: 3.000 |
| > | Max. : 54.00 | Max. $: 1.000000$ | Max. : 18.000 | Max. : 13.000 |
| > | size3added | repstra3 | repstrb3 | repstr3added |
| > | Min. : 0.000 | Min. : 0.000 | Min. 0.000000 | Min. : 0.000 |
| > | 1st Qu.: 1.000 | 1st Qu.: 0.000 | 1st Qu.:0.000000 | 1st Qu.: 0.000 |
| > | Median : 2.000 | Median : 0.000 | Median :0.000000 | Median : 0.000 |
| > | Mean : 3.288 | Mean : 1.069 | Mean :0.009375 | Mean : 1.078 |
| > | 3rd Qu.: 4.000 | 3rd Qu.: 1.000 | 3rd Qu.:0.000000 | 3rd Qu.: 1.000 |
| > | Max. 24.000 | Max. $: 18.000$ | Max. $: 1.000000$ | Max. : 18.000 |
| > | feca3 | fec3added | indcova3 | obsstatus3 repstatus3 |

$>$ Max. :7.0000 fecstatus1
$>$ Min. 0.0000
$>$ 1st Qu.:0.0000
$>$ Median :0.0000
$>$ Mean :0.1344
$>$ 3rd Qu.:0.0000
$>$ Max. :1.0000
stage1index
: 0.00
1st Qu.. 6.00
Median : 8.00
3rd Qu.: 31.00
Max. :51.00
size2added
Min. : 0.000
1st Qu.: 1.000
Median : 2.000
Mean : 3.322
3rd Qu.: 4.000
feca2
Min. :0.0000
1st Qu.:0.0000
Median :0.0000
Mean :0.2906
3rd Qu.:0.0000 Max. :7.0000 repstatus2
Min. : 0.0000
1st Qu.:0.0000
Median :0.0000
Mean :0.3688
3rd Qu.:1.0000
Max. :1.0000 stage2index
Min. : 6.00
1st Qu.: 7.00
Median :10.00
Mean :18.17
3rd Qu.:32.00
Max. : 54.00
size3added 1. 0.000 Median : 2.000 Mean : 3.288 3rd Qu.: 4.000 Max. :24.000 feca3

Max. : 109.80 matstatus1
Min. :0.0000
1st Qu.:1.0000
Median :1.0000
3rd Qu: 1.0000
Max. :1.0000 sizea2
Min. : 0.000000
1st Qu.:0.000000
Median :0.000000
Mean :0.009375
3rd Qu.:0.000000
Max. :1.000000 repstra2
Min. : 0.0000
1st Qu.: 0.0000
Median : 0.0000
Mean : 0.8969
3rd Qu.: 1.0000
fec2added
Min. :0.0000
1st Qu.:0.0000
Median :0.0000
Mean :0.2906
3rd Qu.:0.0000
Max. :7.0000
fecstatus2
Min. :0.0000
1st Qu.:0.0000
Median :0.0000
Mean :0.1562
3rd Qu.:0.0000 sizea3
Min. : 0.000000
1st Qu.:0.000000
Median :0.000000
Mean :0.009375
3rd Qu.:0.000000
. :1.000000 repstra3
Min. : 0.000
1st Qu.: 0.000
an : 0.000

3rd Qu.: 1.000
Max. :18.000
fec3added
indcova3 obsstatus3 repstatus3

```
> Min. :0.0000 Min. :0.0000 Min. : 57.6 Min. :0.0 Min. :0.0
> 1st Qu.:0.0000 1st Qu.:0.0000 1st Qu.: 96.0 1st Qu.:1.0 1st Qu.:0.0
Median :0.0000 Median :0.0000 Median :106.8 Median :1.0 Median :0.0
Mean :0.4562 Mean :0.4562 Mean : 96.1 Mean :0.9 Mean :0.4
3rd Qu.:0.0000 3rd Qu.:0.0000 3rd Qu.:109.8 3rd Qu.:1.0 3rd Qu.:1.0
Max. :8.0000 Max. :8.0000 Max. :111.9 Max. :1.0 Max. :1.0
    fecstatus3 matstatus3 alive3 stage3
Min. :0.0000 Min. :1 Min. :0.0000 Length:320
1st Qu.:0.0000 1st Qu.:1 1st Qu.:1.0000 Class :character
Median :0.0000 Median :1 Median :1.0000 Mode :character
Mean :0.2219 Mean :1 Mean :0.9469
3rd Qu.:0.0000 3rd Qu.:1 3rd Qu.:1.0000
Max. :1.0000 Max. :1 Max. :1.0000
> stage3index
> Min. : 0.00
> 1st Qu.: 7.00
> Median :10.00
> Mean :18.57
> 3rd Qu.:33.00
> Max. :54.00
```

Now we will proceed to run function modelsearch() and find the right parameterizations for our vital rates. We will include the options indcova = c("indcova3", "indcova2", "indcova1") and test. indcova $=$ TRUE so that modelsearch() includes our individual covariates. If these covariates were categorical and random, then we would also need to stipulate random.indcova $=$ TRUE (our individual covariate is actually quantitative and numerical, so we will use the default setting instead). To prevent this function from taking too long to run, we will only do the ahistorical set, we will drop the patch term, and we will set suite = "main" to prevent the inclusion of interaction terms.

```
cypmodels2_env <- modelsearch(cypfb_env, historical = FALSE, approach = "mixed",
    vitalrates = c("surv", "obs", "size", "repst", "fec"),
    sizedist = "negbin", size.trunc = TRUE, fecdist = "poisson", fec.zero = TRUE,
    suite = "main", size = c("size3added", "size2added", "size1added"),
    indcova = c("indcova3", "indcova2", "indcova1"), test.indcova = TRUE,
    quiet = "partial")
```

And now the summary.

```
summary(cypmodels2_env)
> This LefkoMod object includes 5 linear models.
    Best-fit model criterion used: aicc&k
>
>
>
    Survival model:
    Generalized linear mixed model fit by maximum likelihood (Laplace
            Approximation) [glmerMod]
    Family: binomial ( logit )
    Formula: alive3 ~ size2added + (1 | year2) + (1 | individ)
            Data: subdata
            AIC BIC logLik deviance df.resid
    128.1324 143.2057-60.0662 120.1324 316
```

```
> Random effects:
> Groups Name Std.Dev.
> individ (Intercept) 1.198381
> year2 (Intercept) 0.008826
> Number of obs: 320, groups: individ, 74; year2, 5
> Fixed Effects:
> (Intercept) size2added
> 2.0352 0.6344
optimizer (Nelder_Mead) convergence code: O (OK) ; O optimizer warnings; 1 lme4 warnings
>
>
>
> Observation model:
> Generalized linear mixed model fit by maximum likelihood (Laplace
Approximation) [glmerMod]
Family: binomial ( logit )
Formula: obsstatus3 ~ size2added + (1 | year2) + (1 | individ)
            Data: subdata
            AIC BIC logLik deviance df.resid
118.2567 133.1117 -55.1284 110.2567 299
Random effects:
Groups Name Std.Dev.
individ (Intercept) 1.078e-05
year2 (Intercept) 8.776e-01
Number of obs: 303, groups: individ, 70; year2, 5
Fixed Effects:
(Intercept) size2added
2.4904 0.3134
optimizer (Nelder_Mead) convergence code: O (OK) ; O optimizer warnings; 1 lme4 warnings
>
>
>
Size model:
Formula: size3added ~ (1 | year2) + (1 | individ)
Data: subdata
    AIC BIC logLik df.resid
1008.2763 1022.9282 -500.1382 284
Random-effects (co)variances:
>
Conditional model:
Groups Name Std.Dev.
year2 (Intercept) 0.1109
individ (Intercept) 1.0561
Number of obs: 288 / Conditional model: year2, 5; individ, 70
Dispersion parameter for truncated_nbinom2 family (): 2.59e+07
>
> Fixed Effects:
>
> Conditional model:
```

```
> (Intercept)
> 0.5761
>
>
>
> Secondary size model:
> [1] 1
>
>
>
> Tertiary size model:
> [1] 1
>
>
>
Reproductive status model:
Generalized linear mixed model fit by maximum likelihood (Laplace
    Approximation) [glmerMod]
    Family: binomial ( logit )
Formula: repstatus3 ~ indcova2 + repstatus2 + size2added + (1 | year2) +
            (1 | individ)
        Data: subdata
            AIC BIC logLik deviance df.resid
330.0418 352.0196 -159.0209 318.0418 282
Random effects:
Groups Name Std.Dev.
individ (Intercept) 0.0002695
year2 (Intercept) 0.2479120
Number of obs: 288, groups: individ, 70; year2, 5
Fixed Effects:
(Intercept) indcova2 repstatus2 size2added
> -4.23766 0.02954 1.71091 0.17187
>
>
>
Fecundity model:
Formula: feca2 ~ indcova2 + size2added + (1 | year2) + (1 | individ)
Zero inflation: ~size2added + (1 | year2) + (1 | individ)
Data: subdata
    AIC BIC logLik df.resid
    245.4867 270.4229 -113.7434 109
Random-effects (co)variances:
Conditional model:
Groups Name Std.Dev.
year2 (Intercept) 0.2517
individ (Intercept) 0.1927
Zero-inflation model:
Groups Name Std.Dev.
year2 (Intercept) 2.072e-04
```

```
individ (Intercept) 3.951e-08
Number of obs: 118 / Conditional model: year2, 5; individ, 51 / Zero-inflation model: year2, 5; ind
>
Fixed Effects:
Conditional model:
> (Intercept) indcova2 size2added
> 1.68997 -0.02397 0.06174
>
> Zero-inflation model:
> (Intercept) size2added
> 3.927 -1.618
>
>
> Juvenile survival model:
> [1] 1
>
>
>
> Juvenile observation model:
[1] 1
>
>
>
> Juvenile size model:
[1] 1
>
>
>
> Juvenile secondary size model:
[1] 1
>
>
>
> Juvenile tertiary size model:
[1] 1
>
>
>
> Juvenile reproduction model:
[1] 1
>
>
>
> Juvenile maturity model:
[1] 1
>
>
>
>
```

```
>
Number of models in survival table: 7
Number of models in observation table: 8
Number of models in size table: 8
Number of models in secondary size table: 1
Number of models in tertiary size table: 1
Number of models in reproduction status table: 8
Number of models in fecundity table: 60
Number of models in juvenile survival table: 1
Number of models in juvenile observation table: 1
Number of models in juvenile size table: 1
Number of models in juvenile secondary size table: 1
Number of models in juvenile tertiary size table: 1
Number of models in juvenile reproduction table: 1
Number of models in juvenile maturity table: 1
eneral model parameter names (column 1), and
specific names used in these models (column 2):
                    parameter_names mainparams
1
> 2
> 3
>4
observed in time t+1
6
7
8
> 9
    reproductive status in time t+1
        fecundity in time t+1 fec3
            fecundity in time t fec2
                sizea in time t size2
                sizea in time t-1 size1
                    sizeb in time t sizeb2
                    sizeb in time t-1 sizeb1
```

```
> 16 sizec in time t sizec2
> 17 sizec in time t-1 sizec1
> 18 reproductive status in time t repst2
> 19 reproductive status in time t-1 repst1
> 20 maturity status in time t+1 matst3
> 21 maturity status in time t matst2
22 age in time t age
    density in time t density
    individual covariate a in time t indcova2
    individual covariate a in time t-1 indcova1
    individual covariate b in time t indcovb2
    individual covariate b in time t-1 indcovb1
    individual covariate c in time t indcovc2
    individual covariate c in time t-1 indcovc1
            stage group in time t group2
            stage group in time t-1 group1
>
>
>
>
>
> Quality control:
>
> Survival model estimated with 74 individuals and 320 individual transitions.
> Survival model accuracy is 0.947.
> Observation status model estimated with 70 individuals and 303 individual transitions.
> Observation status model accuracy is 0.95.
> Primary size model estimated with 70 individuals and 288 individual transitions.
> Primary size model R-squared is 0.822.
> Secondary size model not estimated.
> Tertiary size model not estimated.
> Reproductive status model estimated with 70 individuals and 288 individual transitions.
> Reproductive status model accuracy is 0.719.
> Fecundity model estimated with 51 individuals and }118\mathrm{ individual transitions.
> Fecundity model R-squared is 0.524.
> Juvenile survival model not estimated.
> Juvenile observation status model not estimated.
> Juvenile primary size model not estimated.
> Juvenile secondary size model not estimated.
> Juvenile tertiary size model not estimated.
> Juvenile reproductive status model not estimated.
> Juvenile maturity status model not estimated.
```

Our individual covariate, total annual precipitation, is included in the best-fit vital rate models for reproductive status and the conditional portion of fecundity. Let's try building an MPM with a single covariate value, which will be equal to 100 cm of annual precipitation.

```
cypmatrix2f_env1 <- flefko2(stageframe = cypframe_fb, data = cypfb_env,
    supplement = cypsupp2_fb, modelsuite = cypmodels2_env, inda = 100)
summary(cypmatrix2f_env1)
>
> This ahistorical lefkoMat object contains 5 matrices.
```

```
>
> Each matrix is square with }54\mathrm{ rows and columns, and a total of 2916 elements.
> A total of 12055 survival transitions were estimated, with 2411 per matrix.
> A total of 240 fecundity transitions were estimated, with 48 per matrix.
> This lefkoMat object covers 1 population, 1 patch, and 5 time steps.
>
> The dataset contains a total of 74 unique individuals and 320 unique transitions.
>
> Vital rate modeling quality control:
>
> Survival estimated with 74 individuals and 320 individual transitions.
> Observation estimated with 70 individuals and 303 individual transitions.
> Primary size estimated with 70 individuals and 288 individual transitions.
Secondary size transition not estimated.
Tertiary size transition not estimated.
Reproductive status estimated with 70 individuals and 288 individual transitions.
Fecundity estimated with 51 individuals and }118\mathrm{ individual transitions.
Juvenile survival not estimated.
Juvenile observation probability not estimated.
Juvenile primary size transition not estimated.
Juvenile secondary size transition not estimated.
Juvenile tertiary size transition not estimated.
Juvenile reproduction probability not estimated.
Juvenile maturity transition probability not estimated.
Survival probability sum check (each matrix represented by column in order):
[,1] [,2] [,3] [,4] [,5]
Min. 0.050 0.050 0.050 0.050 0.050
1st Qu. 0.991 0.991 0.991 0.991 0.991
Median 1.000 1.000 1.000 1.000 1.000
Mean 0.915 0.916 0.916 0.917 0.915
3rd Qu. 1.000 1.000 1.000 1.000 1.000
Max. 1.000 1.000 1.000 1.000 1.000
```

Now let's build a version with a different, set precipitation value for each year (time $t$ only, meaning that there are five values in a six year dataset) based on the actual values.

```
precip_values <- c(92.2, 57.6, 96, 109.8, 111.9)
cypmatrix2f_env2 <- flefko2(stageframe = cypframe_fb, data = cypfb_env,
    supplement = cypsupp2_fb, modelsuite = cypmodels2_env, inda = precip_values)
summary(cypmatrix2f_env2)
>
    This ahistorical lefkoMat object contains 5 matrices.
>
Each matrix is square with 54 rows and columns, and a total of 2916 elements.
A total of }12055\mathrm{ survival transitions were estimated, with }2411\mathrm{ per matrix.
A total of 240 fecundity transitions were estimated, with 48 per matrix.
This lefkoMat object covers 1 population, 1 patch, and 5 time steps.
The dataset contains a total of }74\mathrm{ unique individuals and 320 unique transitions.
>
```

```
> Vital rate modeling quality control:
>
> Survival estimated with 74 individuals and 320 individual transitions.
> Observation estimated with 70 individuals and 303 individual transitions.
> Primary size estimated with 70 individuals and 288 individual transitions.
> Secondary size transition not estimated.
> Tertiary size transition not estimated.
> Reproductive status estimated with 70 individuals and 288 individual transitions.
> Fecundity estimated with 51 individuals and 118 individual transitions.
> Juvenile survival not estimated.
> Juvenile observation probability not estimated.
> Juvenile primary size transition not estimated.
> Juvenile secondary size transition not estimated.
> Juvenile tertiary size transition not estimated.
> Juvenile reproduction probability not estimated.
> Juvenile maturity transition probability not estimated.
>
> Survival probability sum check (each matrix represented by column in order):
> [,1] [,2] [,3] [,4] [,5]
> Min. 0.050 0.050 0.050 0.050 0.050
> 1st Qu. 0.991 0.991 0.991 0.991 0.991
> Median 1.000 1.000 1.000 1.000 1.000
> Mean 0.916 0.917 0.917 0.916 0.915
> 3rd Qu. 1.000 1.000 1.000 1.000 1.000
> Max. 1.000 1.000 1.000 1.000 1.000
```

Voilà! We're done! Now we can move on to analysis, including general projection, LTRE, etc.

### 5.8 Creating function-based MPMs without modelsearch()

Advanced users of lefko3 may wish to create their own models without the package's automated model selection function. In that case, lefko3's function-based MPM creation functions flefko2(), flefko3(), and aflefko2() can accommodate single models developed using a series of function covering generalized linear models and generalized linear mixed models in base R and a series of other packages. Currently, users can develop these models using the following functions.

1. $\operatorname{lm}()$ and $\operatorname{glm}()$ from the core $R$ package stats, with the latter limited to the following response distribution families: binomial, gaussian, Gamma, and poisson
2. glm.nb() from the core R package MASS
3. vglm() from package VGAM (Yee and Wild, 1996; Yee, 2015), with response distributions limited to pospoisson and posnegbinomial
4. zeroinfl() from package pscl (Zeileis et al., 2008), with response distributions limited to poisson and negbin
5. lmer and glmer from package lme4 (Bates et al., 2015), with the latter limited to the following response distribution families: binomial, gaussian, Gamma, and poisson
6. glmmTMB from package glmmTMB (Brooks et al., 2017), with response distribution families limited to nbinom2, truncated_poisson, and truncated_nbinom2

Generally, survival probability, observation probability, and reproduction probability must be modeled using either function $g l m()$ or $g l m e r()$, in either case using the binomial response. Size and fecundity may be modeled as Gaussian, gamma, Poisson, or negative binomial, with the Poisson and
negative binomial capable of being zero-inflated or zero-truncated. The binomial response is not allowed for size and fecundity.

Developing models manually requires a few key steps. First, the core dataset must be created and serve as the main dataset for modeling. Second, data subsets must be developed for conditional rates and probabilities. Third, the models may be developed. Fourth, a paramnames object must be made that identifies the names of variables from the dataset used in modeling. Finally, we can create our MPM using our manual models as inputs.

Let's try an example of a manual modeling exercise. Let's try to create a historical MPM for the Cypripedium candidum dataset, but forcing each model to include size and reproductive status in times $t$ and $t-1$, and occasion in time $t$ and individual identity. So, these will be mixed models. Let's start off by looking over our core $h f v$ format dataset.

```
summary_hfv(cypfb_v1)
>
> This hfv dataset contains }320\mathrm{ rows, }57\mathrm{ variables, 1 population,
3 patches, }74\mathrm{ individuals, and 5 time steps.
rowid popid patchid individ year2
Min. : 1.00 Length:320 A: 93 Min. : 164.0 Min. :2004
1st Qu.:21.00 Class :character B:154 1st Qu.: 391.0 1st Qu.:2005
Median :37.50 Mode :character C: 73 Median : 453.0 Median :2006
Mean :38.45
3rd Qu.:56.00
Max. :77.00
            firstseen
Min. :2004 Min. :2004
1st Qu.:2004 1st Qu.:2009
Median :2004 Median :2009
Mean :2004
3rd Qu.:2004 3rd Qu.:2009 3rd Qu.:8.000 3rd Qu.:5.000
Max. :2008 Max. :2009 Max. :9.000 Max. :5.000
> sizea1 sizeb1 sizec1 size1added
Min. :0.000000 Min. : 0.0000 Min. : 0.0 Min. : 0.000
1st Qu.:0.000000 1st Qu.: 0.0000 1st Qu.: 0.0 1st Qu.: 0.000
Median :0.000000 Median : 0.0000 Median : 1.0 Median : 2.000
Mean :0.009375 Mean : 0.7469 Mean : 1.9 Mean : 2.656
3rd Qu.:0.000000 3rd Qu.: 1.0000 3rd Qu.: 3.0 3rd Qu.: 4.000
Max. :1.000000 Max. :18.0000 Max. :13.0 Max. :21.000
            repstra1 repstrb1 repstr1added feca1
Min. : 0.0000
1st Qu.: 0.0000 1st Qu.:0.000000
Median : 0.0000 Median :0.000000 Median : 0.0000 Median :0.0000
Mean : 0.7469 Mean :0.009375 Mean : 0.7562 Mean :0.2656
    3rd Qu.: 1.0000 3rd Qu.:0.000000 3rd Qu.: 1.0000 3rd Qu.:0.0000
Max. :18.0000 Max. :1.000000 Max. :18.0000 Max. :7.0000
        fec1added obsstatus1 repstatus1 fecstatus1
    Min. :0.0000 Min. :0.0000 Min. :0.0000 Min. :0.0000
    1st Qu.:0.0000 1st Qu.:0.0000 1st Qu.:0.0000 1st Qu.:0.0000
    Median :0.0000 Median :1.0000 Median :0.0000 Median :0.0000
    Mean :0.2656 Mean :0.7469 Mean :0.2875 Mean :0.1344
    3rd Qu.:0.0000 3rd Qu.:1.0000 3rd Qu.:1.0000 3rd Qu.:0.0000
    Max. :7.0000 Max. :1.0000 Max. :1.0000 Max. :1.0000
        matstatus1 alive1 stage1 stage1index
```

```
> Min. :0.0000
> 1st Qu.:1.0000
> Median :1.0000
> Mean :0.7688
> 3rd Qu.:1.0000
> Max. :1.0000
> sizea2 sizeb2
> Min. :0.000000
> 1st Qu.:0.000000
> Median :0.000000
> Mean :0.009375
> 3rd Qu.:0.000000
> Max. :1.000000
        repstra2
> Min. : 0.0000
> 1st Qu.: 0.0000
> Median : 0.0000
> Mean : 0.8969
> 3rd Qu.: 1.0000
> Max. :18.0000
        fec2added
Min. :0.0000
1st Qu.:0.0000
Median :0.0000
Mean :0.2906
> 3rd Qu.:0.0000
> Max. :7.0000
> matstatus2 alive2 stage2 stage2index sizea3
>Min. :1 Min. :1 Length:320 Min. : 6.00 Min. :0.000000
> 1st Qu.:1 1st Qu.:1 Class :character 1st Qu.: 7.00 1st Qu.:0.000000
> Median :1 Median :1 Mode :character Median :10.00 Median :0.000000
> Mean :1 Mean :1 Mean :18.17 Mean :0.009375
> 3rd Qu.:1 3rd Qu.:1 3rd Qu.:32.00 3rd Qu.:0.000000
Max. :1 Max. :1 Max. :54.00 Max. :1.000000
Min. : 0.000
1st Qu.: 0.000
Median : 0.000
Mean : 1.069
3rd Qu.: 1.000
Max. :18.000
    repstrb3 repstr3added
Min. :0.000000 Min. : 0.000
1st Qu.:0.000000
Median :0.000000
Mean :0.009375
3rd Qu.:0.000000
Max. :1.000000 Max. :18.000 Max. :8.0000 Max. :8.0000
    obsstatus3 repstatus3 fecstatus3 matstatus3 alive3
Min. :0.0 Min. :0.0 Min. :0.0000 Min. :1 Min. :0.0000
1st Qu.:1.0 1st Qu.:0.0 1st Qu.:0.0000 1st Qu.:1 1st Qu.:1.0000
```

```
> Median :1.0 
Median :1.0 Median :0.0 
```










```
Median :1.0 
```

We will be estimating survival, and for that we need to assess whether individuals alive in time $t$ are still alive in time $t+1$. Here, variables with a 2 at the end of their names represent variables coding for status in time $t$, while 1 codes for time $t-1$ and 3 codes for time $t+1$. The summary states that alive2 equals 1 in all cases, meaning that all rows in this dataset are for individuals alive in time $t$ (this is the default output from the verticalize3() and historicalize3() functions). So, we can use this dataset to test for survival. Let's create our survival dataset by copying this dataset over to something named more appropriately.

```
surv_data <- cypfb_v1
```

Observation status is assessed in time $t+1$, and needs to be assessed using individuals that are alive in that time. So, let's create a new subset of these data to assess observation status.

```
obs_data <- subset(surv_data, alive3 == 1)
```

Primary size also needs to be assessed using individuals alive in time $t+1$. However, size can only be estimated for individuals that are observed. Hence, we need to subset obs_data to only those individuals observable in time $t+1$, as below.

```
size_data <- subset(obs_data, obsstatus3 == 1)
```

Reproduction status also requires that individuals are observed. Since this is the only requirement, we can simply copy size_data to a more appropriately named data frame for that model estimation.

```
repr_data <- size_data
```

Finally, there is the issue of fecundity. This rate takes some thought to handle properly, because it may be estimated differently in different situations, and hence its relationships to the variables in the dataset may change. In our case, we have utilized a pre-breeding model and use the number of fruits as a proxy for fecundity (the real fecundity rate needs to be multiplied by the number of seeds, the survival of seeds to the next time, and the germination rate, but this product can be handled through the supplement table). Fruits are produced in time $t$ for germination in time $t+1$, and reproductive status is determined by the presence of flowers, so we need a subset of individuals alive in time $t$ that also have repstatus2 $==1$, as below.

```
fec_data <- subset(surv_data, repstatus2 == 1)
```

We have now built all of our data frames, and can proceed to develop our models. Here, each model is listed in a single block of code. We will rely on the glmer() function from package lme4, and the $g l m m T M B()$ function from package $g l m m T M B$. Please install these packages if you have not yet
done so. The response distributions for survival, observation status, and reproduction status will be the binomial distribution. The response distributions for size and fecundity will be the zero-truncated negative binomial and the zero-inflated poisson, respectively.

```
surv_mod <- lme4::glmer(alive3 ~ size2added + size1added + repstatus2 +
    repstatus1 + (1 | year2) + (1 | individ), data = surv_data, family = binomial)
> boundary (singular) fit: see help('isSingular')
obs_mod <- lme4::glmer(obsstatus3 ~ size2added + size1added + repstatus2 +
    repstatus1 + (1 | year2) + (1 | individ), data = obs_data, family = binomial)
size_mod <- glmmTMB::glmmTMB(size3added ~ size2added + size1added + repstatus2 +
    repstatus1 + (1 | year2) + (1 | individ), data = size_data,
    family = glmmTMB::truncated_nbinom2)
> Warning in fitTMB(TMBStruc): Model convergence problem; non-positive-definite
> Hessian matrix. See vignette('troubleshooting')
> Warning in fitTMB(TMBStruc): Model convergence problem; false convergence (8).
> See vignette('troubleshooting')
repr_mod <- lme4::glmer(repstatus3 ~ size2added + size1added + repstatus2 +
    repstatus1 + (1 | year2) + (1 | individ), data = repr_data, family = binomial)
> boundary (singular) fit: see help('isSingular')
fec_mod <- glmmTMB::glmmTMB(feca3 ~ size2added + size1added + repstatus1 +
    (1 | year2) + (1 | individ), zi = ~., data = fec_data, family = poisson)
> Warning in (function (start, objective, gradient = NULL, hessian = NULL, :
> NA/NaN function evaluation
> Warning in (function (start, objective, gradient = NULL, hessian = NULL, :
> NA/NaN function evaluation
    Warning in (function (start, objective, gradient = NULL, hessian = NULL, :
    NA/NaN function evaluation
    Warning in fitTMB(TMBStruc): Model convergence problem; non-positive-definite
    Hessian matrix. See vignette('troubleshooting')
    Warning in fitTMB(TMBStruc): Model convergence problem; false convergence (8).
    See vignette('troubleshooting')
summary(surv_mod)
    Generalized linear mixed model fit by maximum likelihood (Laplace
        Approximation) [glmerMod]
    Family: binomial ( logit )
    Formula: alive3 ~ size2added + size1added + repstatus2 + repstatus1 +
                (1 | year2) + (1 | individ)
        Data: surv_data
            AIC BIC logLik deviance df.resid
        133.2 159.5 -59.6 119.2 313
    Scaled residuals:
    Min 1Q Median 3Q Max
    -8.6555 0.0883 0.1947 0.3066
    Random effects:
    Groups Name Variance Std.Dev.
    individ (Intercept) 1.044e-16 1.022e-08
    year2 (Intercept) 0.000e+00 0.000e+00
```

```
> Number of obs: 320, groups: individ, 74; year2, 5
>
Fixed effects:
Estimate Std. Error z value Pr(>|z|)
(Intercept) 1.50738 0.45233 3.332 0.000861 ***
s size2added 0.48153 0.24076 2.000 0.045493 *
> size1added 0.18764 0.20271 0.926 0.354645
r repstatus2 0.42624 0.68931 0.618 0.536338
repstatus1 -0.05451 0.71646 -0.076 0.939349
---
> Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
> Correlation of Fixed Effects:
> (Intr) sz2ddd sz1ddd rpstt2
size2added -0.592
size1added -0.358 -0.174
> repstatus2 -0.080-0.222 0.042
> repstatus1 -0.079 0.046 -0.318 -0.211
> optimizer (Nelder_Mead) convergence code: O (OK)
> boundary (singular) fit: see help('isSingular')
summary(obs_mod)
> Generalized linear mixed model fit by maximum likelihood (Laplace
> Approximation) [glmerMod]
Family: binomial ( logit )
Formula: obsstatus3 ~ size2added + size1added + repstatus2 + repstatus1 +
            (1 | year2) + (1 | individ)
        Data: obs_data
            AIC BIC logLik deviance df.resid
        122.3 148.3 -54.2 108.3 296
Scaled residuals:
Min 1Q Median 3Q Max
-6.2622 0.1121 0.1716 0.2435
Random effects:
    Groups Name Variance Std.Dev.
    individ (Intercept) 4.153e-07 0.0006444
    year2 (Intercept) 7.302e-01 0.8545190
    Number of obs: 303, groups: individ, 70; year2, 5
    Fixed effects:
            Estimate Std. Error z value Pr(>|z|)
    (Intercept) 2.4945 0.6046 4.126 3.69e-05 ***
    size2added 0.3837 0.1979 1.939 0.0525 .
    size1added -0.1712 0.1507 -1.136 0.2559
    repstatus2 0.4567 0.7075 0.645
    repstatus1 0.4619 0.7343 0.629
    ---
    Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
>
```

```
> Correlation of Fixed Effects:
> (Intr) sz2ddd sz1ddd rpstt2
> size2added -0.257
> size1added -0.149 -0.569
> repstatus2 -0.090 -0.111 -0.051
> repstatus1 -0.043 0.026 -0.199 -0.276
summary(size_mod)
> Family: truncated_nbinom2 ( log )
Formula:
size3added ~ size2added + size1added + repstatus2 + repstatus1 +
> (1 | year2) + (1 | individ)
Data: size_data
> AIC BIC logLik deviance df.resid
> NA NA NA NA 280
>
Random effects:
Conditional model:
    Groups Name Variance Std.Dev.
    year2 (Intercept) 0.01378 0.1174
    individ (Intercept) 0.93532 0.9671
    Number of obs: 288, groups: year2, 5; individ, 70
Dispersion parameter for truncated_nbinom2 family (): 1.86e+07
Conditional model:
Estimate Std. Error z value Pr(>|z|)
(Intercept) 0.51079 0.15819 3.229 0.00124 **
size2added 0.02195 0.01362 1.611
size1added -0.00599 0.01159 -0.517 0.60540
llllll
    repstatus1 0.04482 0.10744 0.417 0.67657
---
> Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
summary(repr_mod)
Generalized linear mixed model fit by maximum likelihood (Laplace
        Approximation) [glmerMod]
    Family: binomial ( logit )
    Formula: repstatus3 ~ size2added + size1added + repstatus2 + repstatus1 +
            (1 | year2) + (1 | individ)
        Data: repr_data
            AIC BIC logLik deviance df.resid
        334.4 360.1 -160.2 320.4 281
    Scaled residuals:
    Min 1Q Median 3Q Max
    -2.1990 -0.6565 -0.3669 0.6895 2.8967
>
Random effects:
```

```
Groups Name Variance Std.Dev.
    individ (Intercept) 0.0000 0.0000
    year2 (Intercept) 0.5097 0.7139
Number of obs: 288, groups: individ, 70; year2, 5
Fixed effects:
    Estimate Std. Error z value Pr(>|z|)
(Intercept) -1.60119 0.40839 -3.921 8.83e-05 ***
size2added 0.12168 0.06222 1.956 0.0505 .
size1added 0.07880 0.07391 1.066 0.2863
repstatus2 1.55291 0.31202 4.977 6.46e-07 ***
repstatus1 0.43554 0.36684 1.187 0.2351
---
Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
Correlation of Fixed Effects:
            (Intr) sz2ddd sz1ddd rpstt2
size2added -0.200
size1added -0.186 -0.520
repstatus2 -0.193-0.131 0.066
repstatus1 -0.110 -0.011 -0.172 -0.272
optimizer (Nelder_Mead) convergence code: O (OK)
boundary (singular) fit: see help('isSingular')
ummary(fec_mod)
Warning in sqrt(diag(vcovs)): NaNs produced
Warning in sqrt(diag(vcovs)): NaNs produced
    Family: poisson ( log )
    Formula: feca3 ~ size2added + size1added + repstatus1 + (1 | year2) +
        (1 | individ)
    Zero inflation:
    Data: fec_data
        AIC BIC logLik deviance df.resid
        NA NA NA NA 106
    Random effects:
    Conditional model:
    Groups Name Variance Std.Dev.
    year2 (Intercept) 0.6514 0.8071
    individ (Intercept) 0.7884 0.8879
    Number of obs: 118, groups: year2, 5; individ, 51
    Zero-inflation model:
    Groups Name Variance Std.Dev.
    year2 (Intercept) 0.6388 0.7993
    individ (Intercept) 7.3858 2.7177
    Number of obs: 118, groups: year2, 5; individ, 51
    Conditional model:
        Estimate Std. Error z value Pr(>|z|)
```

```
> (Intercept) -0.79519 0.26890 -2.957 0.0031 **
> size2added 0.05039
> size1added 0.01121 
> repstatus1 -0.40228 NaN NaN NaN
> ---
> Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
>
> Zero-inflation model:
> Estimate Std. Error z value Pr(>|z|)
> (Intercept) 3.48455 0.34168 10.20 <2e-16 ***
> size2added -2.10471 0.07889 -26.68 <2e-16 ***
> size1added 0.71915 NaN NaN NaN
> repstatus1 -5.15997 NaN NaN NaN
> ---
> Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The last thing that we need to worry about is to create a paramnames object, which is a data frame that contains all of the information necessary for lefko3 to translate these models into MPMs. This should be composed of three columns: a column describing each parameter, a column giving the name of each parameter from lefko3's parameter estimating protocols, and a column giving the name of each parameter within the modeling exercise. We can get a skeleton paramnames object this with the create_pm() function, as below.


```
> 27 individual covariate b in time t-1 indcovb1 none
> 28 individual covariate c in time t indcovc2 none
> 29 individual covariate c in time t-1 indcovc1 none
> 30 stage group in time t group2 none
> 31 stage group in time t-1 group1 none
```

Now we will fill in the modelparams column with the names of the variables that we are using. Bear in mind that only the terms that we actually used in modeling should be changed from none. Giving the names of other variables will cause errors in MPM creation. Here, we change the necessary terms and take a look at the final object.

```
our_pm$modelparams[1] <- "year2"
our_pm$modelparams[2] <- "individ"
our_pm$modelparams[4] <- "alive3"
our_pm$modelparams[5] <- "obsstatus3"
our_pm$modelparams[6] <- "size3added"
our_pm$modelparams[9] <- "repstatus3"
our_pm$modelparams[10] <- "feca3"
our_pm$modelparams[11] <- "feca2"
our_pm$modelparams[12] <- "size2added"
our_pm$modelparams[13] <- "size1added"
our_pm$modelparams[18] <- "repstatus2"
our_pm$modelparams[19] <- "repstatus1"
our_pm
> parameter_names mainparams modelparams
> 1
> 2
> 3
> 4
> 5
> 6
>7 sizeb in time t+1
> sizec in time t+1 sizec3 s
> 9 reproductive status in time t+1 repst3 repstatus3
> 10 fecundity in time t+1 fec3 feca3
> 11 fecundity in time t fec2 feca2
> 12 sizea in time t size2
> 13
> 14
> 15
> 16
> 17
> 18
> 19
> 20
> 21
> 22
> 23
> 24 individual covariate a in time t indcova2 none
> 25 individual covariate a in time t-1 indcova1 none
> 26 individual covariate b in time t indcovb2 none
```

```
> 27 individual covariate b in time t-1 indcovb1 none
> 28 individual covariate c in time t indcovc2 none
> 29 individual covariate c in time t-1 indcovc1 none
> 30 stage group in time t group2 none
> 31 stage group in time t-1 group1 none
```

Finally, let's create our new MPM. Note that we will not set modelsuite this time. Instead, we will use some other model setting options.

```
cypmatrix3f_manual <- flefko3(stageframe = cypframe_fb, data = cypfb_v1,
    supplement = cypsupp3_fb, surv_model = surv_mod, obs_model = obs_mod,
    size_model = size_mod, repst_model = repr_mod, fec_model = fec_mod,
    paramnames = our_pm, sparse_output = TRUE)
summary(cypmatrix3f_manual)
>
> This historical lefkoMat object contains 5 matrices.
>
> Each matrix is square with 2916 rows and columns, and a total of }8503056\mathrm{ elements.
> A total of 589540 survival transitions were estimated, with }117908\mathrm{ per matrix.
> A total of }11760\mathrm{ fecundity transitions were estimated, with }2352\mathrm{ per matrix.
> This lefkoMat object covers 1 population, 1 patch, and 5 time steps.
>
> The dataset contains a total of 74 unique individuals and 320 unique transitions.
>
> Survival probability sum check (each matrix represented by column in order):
> [,1] [,2] [,3] [,4] [,5]
> Min. 0.000 0.000 0.000 0.000 0.000
> 1st Qu. 0.992 0.992 0.992 0.992 0.992
> Median 0.999 1.000 1.000 0.999 0.999
> Mean 0.823 0.824 0.824 0.824 0.823
> 3rd Qu. 1.000 1.000 1.000 1.000 1.000
> Max. 1.000 1.000 1.000 1.000 1.000
```

Our summary is a little simpler than in previous cases, because we did not take advantage of the extra power provided by modelsearch(). However, we can still see the information of greatest importance. Please note that defining vital rate models manually is a highly sensitive exercise - it is likely that the first attempts will yield fatal errors, often involving odd error messages. If at first you don't succeed, try finessing the dataset, the vital rate models, and the paramnames object to align completely with each other.

### 5.9 Points to remember

1. Ahistorical MPMs should never be built using models that hold historical terms.
2. The choices of response distributions for size and fecundity strongly influence the relationships between independent terms and their responses. Response distributions strongly influence the structure of best-fit models, and can strongly alter MPMs and their analyses.
3. The more complex the global model, the longer it will take to run the modeling exercise. In extreme cases, in which highly complex models are run on slower computers with low memory levels, the function might run for several days. In these circumstances, the user is advised to carefully consider which terms are really needed for testing, and, if the resulting model is large, then setting the function to run on a powerful computer with lots of memory is advised.
4. All else being equal, mixed models are better than GLMs. This is because they account for the lack of independence among transitions originating from the same individual by treating individual as a random factor. However, the complexity of mixed models may at times make the modeling exercise run very long, or possibly fail. In these cases, switching to GLMs might help yield results on a reasonably time frame. 5 Always remember to assess the quality of the data and the resulting vital rate models. Function $h f v_{\mathrm{v}} \mathrm{qc}()$ can help with the former. Object qc in lefkoMod output can help with the latter.

## Chapter 6

## Matrix Models III: Age (Leslie), Hybrid Age, and Age-by-Stage MPMs

"The really frightening thing about middle age is that you know you'll grow out of it."

The first MPMs were age-classified, and are known as Leslie MPMs in honor of their creator (Leslie, 1945). However, in many organisms, age is considered less important in determining demography than other variables. Stage-classified, or Lefkovitch, MPMs were developed with this in mind, and they allow the life history of an organism to be stratified by life history stages instead of age (Lefkovitch, 1965). Life history stages can be defined as necessary by any status variable or combination of status variables (Caswell, 2001), making the stage-based MPM a very flexible and powerful means to analyze population dynamics.

For much of the history of the MPM, the dichotomy between age-classified and stage-classified MPMs was strict and driven by the lack of computing power to analyze complex relationships in large matrices. With the advent of powerful home computers, more and more ecologists have addressed questions in population dynamics using age-by-stage MPMs, which are MPMs that incorporate both age and stage (Caswell and Salguero-Gómez, 2013).

In this chapter, we will describe three kinds of age-classified MPMs, and how to build them using lefko3. They are as follows:

1. Leslie MPMs: These are purely age-classified, and so are the simplest MPM to consider (section 6.5).
2. Age-hybrid MPMs: These are typically Leslie MPMs with an extra stage added to reflect a single stage that does not fit the age-based convention (section 6.6).
3. Age-by-stage MPMs: These MPMs are built as block matrices with a stage-based design that is repeated across ages, and so are the most complicated of the age-incorporating MPMs (section $6.2)$.

### 6.1 What are age-based, age-hybrid, and age-by-stage MPM?

We will begin with a short introduction to the core theory on incorporating age into MPMs. We strongly advise readers interested in this style of MPM to read Caswell et al. (2018), which is the most comprehensive overview of the theory particularly behind age-by-stage MPMs that we are aware of.

Age-classified, or Leslie, MPMs treat the life history of an organism as a series of ages rather than stages. Each age is of equal duration, and the number of ages to include is decided through demographic studies that attempt to determine which groups of ages have unique demographic characteristics. The only transitions that are possible are one-way survival transitions from one age to the next, survival transitions between the final age and itself if the lifespan has no hard limit, and fecundity from any reproductive age to age 0 (if post-breeding) or 1 (if pre-breeding).

Let's look at an example. Let us suppose that we are studying an organism that can live for potentially many years, such as a long-lived seabird like an albatross. We conduct several years of monitoring, and find that the seabird at our monitored population seems to follow a basic pattern in which survival is relatively low in the first year, higher in the second, and from the third year on survival is typically quite high and stable. Although the maximum longevity is unknown, nonetheless birds seem to live not much longer than 20 years. Fecundity occurs from the third year on, with the first two years spent as a juvenile. In this circumstance, we might develop a Leslie matrix that looks like this.

$$
\left[\begin{array}{rrr}
0 & 0 & F_{1,3}  \tag{6.1}\\
S_{2,1} & 0 & 0 \\
0 & S_{3,2} & S_{3,3}
\end{array}\right]
$$

This matrix includes only three rows and columns. Survival-transition probabilities are typically just below the diagonal, and represent the probability of surviving from age $i$ (column $i$ ) to age $i+1$ (row $i+1$ ). There is a notable exception in element $S_{3,3}$, which is the survival-transition probability for an organism to stay alive within the final age. This allows us to model the tendency of organisms to live on without an explicit maximum longevity in a state in which the probability of survival is not expected to change. Fecundity is in the top row $\left(F_{1,3}\right)$, since all offspring begin in the first age.

Let's now consider a more complicated case. Perhaps we have a similar life history in a plant, in which we have three core ages. However, we might wish to add a dormant seed stage, which allows the plant to survive as a dormant propagule that does not fit the age dynamics noted above. In that case, we might build an age-hybrid MPM, in which we simply add a stage as a new row and column to the matrix, as below.

$$
\left[\begin{array}{rrrr}
S_{1,1} & 0 & 0 & F_{1,4}  \tag{6.2}\\
S_{2,1} & 0 & 0 & F_{2,4} \\
0 & S_{3,2} & 0 & 0 \\
0 & 0 & S_{4,3} & S_{4,4}
\end{array}\right]
$$

Note in the equation above that the first row and the first column are different, and there are now two fecundity paths reflecting the production of dormant seed and seed that germinate within a year of production. However, the rest of the MPM is the same as equation (6.1) except with the subscripts updated.

Finally, age-by-stage MPMs differ from the above because they incorporate both age and stage throughout, and so the matrix needs to be expanded to include both characteristics. The standard approach is to create block matrices in which each age has potentially several stages, and survivaltransition probabilities must describe the probability of transitioning from each stage at one age to each stage in the next.

As an example, let's take the three stage life history above, which has a single newborn stage and two adult stages, only one of which is reproductive. Let's further assume that age impacts survival and fecundity only for the first four years, beyond which survival and fecundity transitions remain essentially the same. These conditions lead to the following age-by-stage matrix.

$$
\left[\begin{array}{rrrrrrr}
0 & 0 & F_{1 A, 2 C} & 0 & F_{1 A, 3 C} & 0 & F_{1 A, 4+C}  \tag{6.3}\\
S_{2 B, 1 A} & 0 & 0 & 0 & 0 & 0 & 0 \\
S_{2 C, 1 A} & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & S_{3 B, 2 B} & S_{3 B, 2 C} & 0 & 0 & 0 & 0 \\
0 & S_{3 C, 2 B} & S_{3 C, 2 C} & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & S_{4 B, 3 B} & S_{4 B, 3 C} & S_{5+B, 4 B} & S_{5+B, 4+C} \\
0 & 0 & 0 & S_{4 C, 3 B} & S_{4 C, 3 C} & S_{5+C, 4 B} & S_{5+C, 4+C}
\end{array}\right]
$$

Here, we have only one possible initial stage, followed by two possible stages in all later ages. Looking only at the columns, column 1 refers to age 1 , which only includes the seedling stage; columns 2 and 3 refer to age 2 , which includes non-reproductive and reproductive adult stages; columns 4 and 5 refer to age 3 , which also includes non-reproductive and reproductive adult stages; and columns 6 and 7 refer to ages 4 and higher, once again also including non-reproductive and reproductive adult stages. While the columns refer to the From ages and stages, the rows refer to the To ages and stages and are in the same order as the columns. If we ignore stage and consider only the placement of age-related transitions, then we see that we are still generally following the Leslie MPM pattern of placing survival-transition probabilities below the diagonal for all but the final transitions. We also have more fecundity terms because although only one stage is reproductive, multiple reproductive ages occur, and all of these stay within the first row (more entry rows are possible only if the life history of the organism can start from multiple different stages).

The above matrix is ahistorical. However, unlike a pure Leslie MPM, the possibility of different stages within each age allows for the estimation of historical age-by-stage MPMs. Package lefko3 does not currently support historical age-by-stage matrices, but this capability will be added in the future.

### 6.2 Developing function-based age-by-stage MPMs

Package lefko3 can produce both raw (empirical) and function-based age-by-stage MPMs. Let's start with the latter. For our example, we will use the lathyrus dataset that comes with lefko3 to illustrate the estimation of age-by-stage function-based MPMs. First, let's load the dataset, and then look at its dimensions and a summary.

```
data(lathyrus)
dim(lathyrus)
> [1] 1119 38
summary(lathyrus)
        SUBPLOT
> Min. :1.000
> 1st Qu.:2.000
> Median :3.000
> Mean :3.223
> 3rd Qu.:4.000
> Max. :6.000 Max. :284.0 Max. :7032.0 Max. :8.900
>
> FCODE88
> Min. :0.0000
> 1st Qu.:0.0000
> Median :0.0000
> Mean :0.3399
> 3rd Qu.:1.0000
\begin{tabular}{|c|c|c|c|}
\hline GENET & \multicolumn{3}{|l|}{Volume88 lnVol88} \\
\hline Min. : 1.0 & Min. : 3.4 & \multicolumn{2}{|l|}{Min. :1.200} \\
\hline 1st Qu.: 48.0 & 1st Qu.: 63.0 & \multicolumn{2}{|l|}{1st Qu.:4.100} \\
\hline Median : 97.0 & Median : 732.5 & \multicolumn{2}{|l|}{Median :6.600} \\
\hline Mean : 110.2 & Mean : 749.4 & \multicolumn{2}{|l|}{Mean :5.538} \\
\hline 3rd Qu.:167.5 & 3rd Qu.:1025.5 & \multicolumn{2}{|l|}{3rd Qu.:6.900} \\
\hline Max. 284.0 & Max. :7032.0 & Max. :8.900 & \\
\hline & NA's : 404 & NA's : 404 & \\
\hline Flow88 & Intactseed88 & Dead1988 & Dormant1988 \\
\hline Min. : 1.00 & Min. : 0 & Mode:logical & Mode:logical \\
\hline 1st Qu.: 4.00 & 1st Qu.: 0 & NA's:1119 & NA's:1119 \\
\hline Median : 8.00 & Median : 0 & & \\
\hline Mean : 11.86 & Mean : 3 & & \\
\hline 3rd Qu.:15.00 & 3rd Qu.: 4 & & \\
\hline
\end{tabular}
```



| $>$ | Min. $: 1$ |  | Min. $: 1.000$ |
| :--- | :--- | :--- | ---: |
| $>$ | 1st Qu.:1 |  | 1st Qu. $: 2.000$ |
| $>$ | Median $: 1$ |  | Median $: 2.000$ |
| $>$ | Mean $: 1$ |  | Mean $: 1.973$ |
| $>$ | 3rd Qu.:1 | 3rd Qu. $: 2.000$ |  |
| $>$ | Max. $: 1$ | Max. $: 3.000$ |  |
| $>$ | NA's $: 1095$ | NA's $: 1082$ |  |

This dataset includes information on 1,119 individuals, so there are 1,119 rows with data (not counting the header). There are 38 columns. The first two columns are variables identifying each individual (SUBPLOT refers to the patch, and GENET refers to individual identity), with each individual's data entirely restricted to one row. This is followed by four sets of nine columns, each named VolumeXX, InVolXX, FCODEXX, FlowXX, IntactseedXX, Dead19XX, DormantXX, Missing19XX, and SeedlingXX, where XX corresponds to the year of observation and with years organized consecutively. Thus, columns 3-11 refer to year 1988, columns 12-20 refer to year 1989, etc.

To begin, we will create a stageframe that describes the organism's life cycle for this dataset. In this case, the life history model is a life cycle graph (figure 6.1).


Figure 6.1: Life history model of Lathyrus vernus using log leaf volume as the size classification metric

This model is based on the life history model provided in Ehrlén (2000), but we utilize a different
size classification based on the log leaf volume to make the size distribution more closely match a symmetric and somewhat normal distribution.

Our stageframe needs to include complete descriptions of all stages that occur in the dataset, with each stage defined uniquely, and also needs to describe the ages for each stage portrayed in our life history model. Since this object can be used for automated classification of individuals, all sizes, reproductive states, and other characteristics defining each proper stage in the dataset need to be accounted for explicitly. This can be difficult if a few data points exist outside the range of sizes specified in the stageframe. Such points can cause problems, because rare stages can cause an overestimation of survival for existing stages under some circumstances, and can also yield spurious values for survival-transition probabilities and fecundity rates. The final description of each stage occurring in the dataset must also avoid complete overlap with any other stage also found in the dataset, although partial overlap is allowed and expected.

Before creating the stage frame, let's explore the possible size variables. We will particularly look at summaries of the distribution of original and log sizes.

```
summary(c(lathyrus$Volume88, lathyrus$Volume89, lathyrus$Volume90,
    lathyrus$Volume91))
> Min. 1st Qu. Median Mean 3rd Qu. Max. NA's
> 1.8 14.7 123.0 484.2 
summary(c(lathyrus$lnVol88, lathyrus$lnVol89, lathyrus$lnVol90,
    lathyrus$lnVol91))
> Min. 1st Qu. Median Mean 3rd Qu. Max. NA's
> 0.600
```

The upper summary shows the original size, while the lower line shows the size given in logarithmic terms. We should note the size minima and maxima, because we have been using 0 as the size of vegetatively dormant individuals. The lowest uncorrected size is 1.8 , with a maximum of 7032.0 . The minimum corrected size is 0.600 , and the maximum corrected size is 8.900 . Since the minimum corrected size is above 0 (i.e. all $\log$ sizes should be positive), and since the number of NAs has not increased (increased NAs would suggest that some unusable log sizes occur in the dataset), we are still able to use the $\log$ size value 0 as an indicator of vegetative dormancy. Note, however, that vegetative dormancy is also currently included in the many NAs that occur in size variables in this dataset.

It can also help to take a look at plots of these distributions. We will plot raw and log volume (figure 6.2).

```
par(mfrow=c (1,2))
plot(density(c(lathyrus$Volume88, lathyrus$Volume89, lathyrus$Volume90,
    lathyrus$Volume91), na.rm = TRUE), main = "", xlab = "Volume", bty = "n")
title("a)", adj = 0)
plot(density(c(lathyrus$lnVol88, lathyrus$lnVol89, lathyrus$lnVol90,
    lathyrus$lnVol91), na.rm = TRUE), main = "", xlab = "Log volume", bty = "n")
title("b)", adj = 0)
```

Note how highly skewed the raw volume distribution is. This might cause some difficulty if we use raw size untransformed and with a Gaussian distribution. Certainly, a gamma distribution would be perfectly justified, and users are urged to try that approach. We will use the log volume here, which looks 'better' than the raw volume distribution in the sense that it is closer to some semblance of a Gaussian distribution, mostly through an increased level of symmetry. We can then assume that log volume is Gaussian-distributed and that the mean bears no relationship to the variance.

We will now develop a stageframe that incorporates the log volume of size. We will build this by creating vectors of the values describing each stage, always in the same order. Because we wish to


Figure 6.2: Density plot of aboveground plant volume (a) and log volume (b).
build an age-by-stage MPM, we will also incorporate age information for each stage. Here, we include minimum and maximum ages for each stage via the vectors minima and maxima ( NA in the maximum age vector is interpreted as meaning that there is no maximum).

```
sizevector <- c(0, 4.6, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 1, 2, 3, 4, 5, 6, 7, 8, 9)
stagevector <- c("Sd", "Sdl", "Dorm", "Sz1nr", "Sz2nr", "Sz3nr", "Sz4nr",
    "Sz5nr","Sz6nr", "Sz7nr", "Sz8nr", "Sz9nr", "Sz1r", "Sz2r", "Sz3r", "Sz4r",
    "Sz5r", "Sz6r", "Sz7r", "Sz8r", "Sz9r")
repvector <- c(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1)
obsvector <- c(0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1)
matvector <- c(0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1)
immvector <- c(1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
propvector <- c(1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
indataset <- c(0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1)
binvec <- c(0, 4.6, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5,
                0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5)
minima <- c(1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2)
maxima <- c(NA, 1, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,
    NA, NA, NA, NA, NA)
comments <- c("Dormant seed", "Seedling", "Dormant", "Size 1 Veg", "Size 2 Veg",
    "Size 3 Veg", "Size 4 Veg", "Size 5 Veg", "Size 6 Veg", "Size 7 Veg",
    "Size 8 Veg", "Size 9 Veg", "Size 1 Flo", "Size 2 Flo", "Size 3 Flo",
    "Size 4 Flo", "Size 5 Flo", "Size 6 Flo", "Size 7 Flo", "Size 8 Flo",
    "Size 9 Flo")
lathframeln <- sf_create(sizes = sizevector, stagenames = stagevector,
    repstatus = repvector, obsstatus = obsvector, propstatus = propvector,
    immstatus = immvector, matstatus = matvector, indataset = indataset,
    binhalfwidth = binvec, minage = minima, maxage = maxima, comments = comments)
```

| lathframeln |  |  |  |  |  |  |  |  |  |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| > | stage | size | size_b | $z e . c$ | min_age | max_age | repstatus | obsstatus | propstatus |
| $>1$ | Sd | 0.0 | NA | NA | 1 | NA | 0 | 0 | 1 |
| $>2$ | Sdl | 4.6 | NA | NA | 1 | 1 | 0 | 1 | 0 |
| $>3$ | Dorm | 0.0 | NA | NA | 2 | NA | 0 | 0 | 0 |
| > 4 | Sz1nr | 1.0 | NA | NA | 2 | NA | 0 | 1 | 0 |
| $>5$ | Sz2nr | 2.0 | NA | NA | 2 | NA | 0 | 1 | 0 |
| $>6$ | Sz3nr | 3.0 | NA | NA | 2 | NA | 0 | 1 | 0 |
| $>7$ | Sz4nr | 4.0 | NA | NA | 2 | NA | 0 | 1 | 0 |
| $>8$ | Sz5nr | 5.0 | NA | NA | 2 | NA | 0 | 1 | 0 |
| > 9 | Sz6nr | 6.0 | NA | NA | 2 | NA | 0 | 1 | 0 |
| > 10 | Sz7nr | 7.0 | NA | NA | 2 | NA | 0 | 1 | 0 |
| > 11 | Sz8nr | 8.0 | NA | NA | 2 | NA | 0 | 1 | 0 |
| > 12 | Sz9nr | 9.0 | NA | NA | 2 | NA | 0 | 1 | 0 |
| > 13 | Sz1r | 1.0 | NA | NA | 2 | NA | 1 | 1 | 0 |
| > 14 | Sz2r | 2.0 | NA | NA | 2 | NA | 1 | 1 | 0 |
| > 15 | Sz3r | 3.0 | NA | NA | 2 | NA | 1 | 1 | 0 |
| > 16 | Sz4r | 4.0 | NA | NA | 2 | NA | 1 | 1 | 0 |
| $>17$ | Sz5r | 5.0 | NA | NA | 2 | NA | 1 | 1 | 0 |
| > 18 | Sz6r | 6.0 | NA | NA | 2 | NA | 1 | 1 | 0 |
| > 19 | Sz7r | 7.0 | NA | NA | 2 | NA | 1 | 1 | 0 |
| > 20 | $\mathrm{Sz8r}$ | 8.0 | NA | NA | 2 | NA | 1 | 1 | 0 |
| > 21 | Sz9r | 9.0 | NA | NA | 2 | NA | 1 | 1 | 0 |
| $>$ | immstatus matstatus indataset binhalfwidth_raw sizebin_min sizebin_max |  |  |  |  |  |  |  |  |
| > 1 |  | 1 | 0 |  | 0 |  | 0.0 | 0.0 | 0.0 |
| $>2$ |  | 1 | 0 |  | 1 |  | 4.6 | 0.0 | 9.2 |
| $>3$ |  | 0 | 1 |  | 1 |  | 0.5 | -0.5 | 0.5 |
| $>4$ |  | 0 | 1 |  | 1 |  | 0.5 | 0.5 | 1.5 |
| $>5$ |  | 0 | 1 |  | 1 |  | 0.5 | 1.5 | 2.5 |
| $>6$ |  | 0 | 1 |  | 1 |  | 0.5 | 2.5 | 3.5 |
| $>7$ |  | 0 | 1 |  | 1 |  | 0.5 | 3.5 | 4.5 |
| $>8$ |  | 0 | 1 |  | 1 |  | 0.5 | 4.5 | 5.5 |
| $>9$ |  | 0 | 1 |  | 1 |  | 0.5 | 5.5 | 6.5 |
| > 10 |  | 0 | 1 |  | 1 |  | 0.5 | 6.5 | 7.5 |
| $>11$ |  | 0 | 1 |  | 1 |  | 0.5 | 7.5 | 8.5 |
| > 12 |  | 0 | 1 |  | 1 |  | 0.5 | 8.5 | 9.5 |
| $>13$ |  | 0 | 1 |  | 1 |  | 0.5 | 0.5 | 1.5 |
| > 14 |  | 0 | 1 |  | 1 |  | 0.5 | 1.5 | 2.5 |
| > 15 |  | 0 | 1 |  | 1 |  | 0.5 | 2.5 | 3.5 |
| > 16 |  | 0 | 1 |  | 1 |  | 0.5 | 3.5 | 4.5 |
| $>17$ |  | 0 | 1 |  | 1 |  | 0.5 | 4.5 | 5.5 |
| > 18 |  | 0 | 1 |  | 1 |  | 0.5 | 5.5 | 6.5 |
| > 19 |  | 0 | 1 |  | 1 |  | 0.5 | 6.5 | 7.5 |
| > 20 |  | 0 | 1 |  | 1 |  | 0.5 | 7.5 | 8.5 |
| > 21 |  | 0 | 1 |  | 1 |  | 0.5 | 8.5 | 9.5 |
|  | sizebin_center sizebin_width binhalfwidthb_raw sizebinb_min sizebinb_max |  |  |  |  |  |  |  |  |
| $>1$ |  |  | 0.0 |  | 0.0 |  | NA | NA | NA |
| $>2$ |  |  | 4.6 |  | 9.2 |  | NA | NA | NA |
| $>3$ |  |  | 0.0 |  | 1.0 |  | NA | NA | NA |
| $>4$ |  |  | 1.0 |  | 1.0 |  | NA | NA | NA |
| > 5 |  |  | 2.0 |  | 1.0 |  | NA | NA | NA |



| $>13$ | NA | NA | 0 | Size | 1 Flo |
| :---: | :---: | :---: | :---: | :---: | :---: |
| > 14 | NA | NA | 0 | Size | Flo |
| > 15 | NA | NA | 0 | Size | 3 Flo |
| > 16 | NA | NA | 0 | Size | 4 Flo |
| > 17 | NA | NA | 0 | Size | 5 Flo |
| > 18 | NA | NA | 0 | Size | 6 Flo |
| > 19 | NA | NA | 0 | Size | 7 Flo |
| > 20 | NA | NA | 0 | Size | 8 Flo |
| > 21 | NA | NA | 0 | Size | Flo |

Once the stageframe is created, we can reorganize the dataset into historically-formatted vertical (hfv) format. Before doing this, we need to alter the dataset slightly. Currently, the variable GENET lists the individual number, but only within each subpopulation. We wish to identify each individual within each subpopulation uniquely, and this requires us to develop a new ID variable. For this purpose, we will create a new variable that concatenates the subpopulation and genet number together into one string, as below.
lathyrus\$indiv_id <- paste(lathyrus\$SUBPLOT, lathyrus\$GENET)
Now let's use the verticalize3() function, which creates historically-formatted vertical datasets, as below. We will also replace NAs in size and fecundity variables with zeros for modelsearch to work properly when we build models of vital rates, so we will now set NAas0 $=$ TRUE. Some care needs to be taken with this last step, since some authors give missing values extra meaning not present in a value of zero. In our case, a missing value indicates that a plant was dead (both size and fecundity are missing), was alive but not sprouting (size was missing), or was alive but did not produce seed (fecundity was missing). In all cases, these NA values may be replaced by zero, because other variables indicate those conditions.

We also have two choices for use as our reproductive status and fecundity variables. The first choice, FCODE88 indicates whether a plant flowered. The second choice, Intactseed88, indicates the number of seed produced. The choice of which to use depends strongly on the aims of the study. In our case, we would like to treat all plants that flowered as reproductive, but treat fecundity in terms of real seed produced. The reason is that we believe that flowering plants have a different demography than non-flowering plants, either reflecting reproductive costs, or, conversely, because flowering plants might have more resources and hence higher survival than non-flowering plants, and so we wish to separate transitions among these two groups. So, let's use FCODE88 to indicate reproductive status, and Intactseed88 to indicate fecundity. Once complete, we will look at a summary.

Finally, note that in the input to the following function, we utilize a strictly repeating pattern of variable names arranged in the same order for each monitoring occasion. This arrangement allows us to enter only the first variable in each set, as long as noyears and blocksize are set properly and no gaps or shuffles appear in the dataset. The data management functions that we have created for lefko3 do not require such repeating patterns, but they do make the required input in the function much shorter and more succinct.

```
lathvertln <- verticalize3(lathyrus, noyears = 4, firstyear = 1988,
    patchidcol = "SUBPLOT", individcol = "indiv_id", blocksize = 9,
    juvcol = "Seedling1988", sizeacol = "lnVol88", repstracol = "FCODE88",
    fecacol = "Intactseed88", deadacol = "Dead1988", nonobsacol = "Dormant1988",
    stageassign = lathframeln, stagesize = "sizea", censorcol = "Missing1988",
    censorkeep = NA, censorRepeat = TRUE, NAas0 = TRUE, censor = TRUE)
summary_hfv(lathvertln)
>
```




This dataset has 2,527 rows representing our original dataset of over 1000 individuals. Ordinarily, we would now go on to produce either vital rate models to create function-based MPMs. However, the fact that we are incorporating age in our analysis leads to the problem that there are many individuals in our dataset that are of unknown age. Lathyrus is a long-lived plant, and this monitoring study lasted only four years, leading to a poor understanding of the ages of most plants. The hfv dataset that we have created includes an estimated age for each individual in each year, but this is estimated from the time in which the individual is first seen. Many individuals are first seen in the first year of the study, by which time many could have already been alive for years. So, we will subset our data to eliminate individuals first seen in the very first year of the study.

```
lathvertln_small <- subset(lathvertln, firstseen > 1988)
summary_hfv(lathvertln_small)
>
> This hfv dataset contains 531 rows, 54 variables, 1 population,
> 6 patches, }345\mathrm{ individuals, and 2 time steps.
\begin{tabular}{lllll}
\(>\) & \multicolumn{2}{c}{ rowid } & \multicolumn{1}{c}{ popid } & \multicolumn{1}{c}{ patchid }
\end{tabular} individ
> Mean : 585.3 Mean :3.077
> 3rd Qu.: 800.5 3rd Qu.:4.000
> Max. :1097.0 Max. :6.000
> year2 firstseen lastseen obsage obslifespan
> Min. :1989 Min. :1989 Min. :1989 Min. :1.00 Min. :0.000
> 1st Qu.:1989 1st Qu.:1989 1st Qu.:1990 1st Qu.:1.00 1st Qu.:1.000
> Median :1990 Median :1989 Median :1991 Median :1.00 Median :2.000
> Mean :1990 Mean :1989 Mean :1991 Mean :1.35 Mean :1.367
> 3rd Qu.:1990 3rd Qu.:1989 3rd Qu.:1991 3rd Qu.:2.00 3rd Qu.:2.000
> Max. :1990 Max. :1990 Max. :1991 Max. :2.00 Max. :2.000
> sizea1 size1added repstra1 repstr1added
> Min. :0.000 Min. :0.000 Min. :0.00000 Min. :0.00000
> 1st Qu.:0.000 1st Qu.:0.000 1st Qu.:0.00000 1st Qu.:0.00000
> Median :0.000 Median :0.000 Median :0.00000 Median :0.00000
> Mean :1.175 Mean :1.175 Mean :0.02072 Mean :0.02072
> 3rd Qu.:2.200 3rd Qu.:2.200 3rd Qu.:0.00000 3rd Qu.:0.00000
> Max. :8.400 Max. :8.400 Max. :1.00000 Max. :1.00000
> feca1 fec1added censor1 juvgiven1
> Min. : 0.0000 Min. : 0.0000 Min. :0 Min. :0.000
> 1st Qu.: 0.0000 1st Qu.: 0.0000 1st Qu.:0 1st Qu.:0.000
> Median : 0.0000 Median : 0.0000 Median :0 Median :0.000
> Mean : 0.1243 Mean : 0.1243 Mean :0 Mean :0.162
> 3rd Qu.: 0.0000 3rd Qu.: 0.0000 3rd Qu.:0 3rd Qu.:0.000
> Max. :34.0000 Max. :34.0000 Max. :0 Max. :1.000
            obsstatus1 repstatus1 fecstatus1 matstatus1
> Min. :0.0000 Min. :0.00000 Min. :0.00000 Min. :0.0000
> 1st Qu.:0.0000 1st Qu.:0.00000 1st Qu.:0.00000 1st Qu.:0.0000
> Median :0.0000 Median :0.00000 Median :0.00000 Median :0.0000
> Mean :0.3503 Mean :0.02072 Mean :0.01318 Mean :0.1883
> 3rd Qu.:1.0000 3rd Qu.:0.00000 3rd Qu.:0.00000 3rd Qu.:0.0000
> Max. :1.0000 Max. :1.00000 Max. :1.00000 Max. :1.0000
> alive1 stage1 stage1index sizea2
> Min. :0.0000 Length:531
> 1st Qu.:0.0000
> Median :0.0000
> Mean :0.3503
> 3rd Qu.:1.0000
> Max. :1.0000
        size2added
Min.0.000 Min :0.00000
> Min. :0.000 Min. :0.00000
> 1st Qu.:2.200 1st Qu.:0.00000
> Median :2.500 Median :0.00000 Median :0.00000 Median : 0.000
```



We are now down to only 531 rows and 345 individuals, so we have lost roughly $80 \%$ of the transition data here. That is a steep price to pay for accurate inference, but it is necessary in this case.

Let's look at fecundity. Fecundity is integer-based, suggesting that it can be treated as a count variable. This package currently allows eight choices of count distributions: Gaussian, gamma, Poisson, negative binomial, zero-inflated Poisson, zero-inflated negative binomial, zero-truncated Poisson, and zero-truncated negative binomial. To assess which to use, we should first assess whether the mean
and variance of the count are equal using a dispersion test. This test allows us to test whether the variance is greater than that expected under our mean value for fecundity using a chi-squared test. If it is not significantly different, then we may use some variant of the Poisson distribution. If the data are overdispersed, then we should use the negative binomial distribution. We should also test whether the number of zeros is more than expected under these distributions, and make the distribution zeroinflated if so. Note that, because we have not excluded 0s from fecundity using reproductive status, we should not use a zero-truncated distribution.

Let's start off by looking at a bar plot of the distribution of fecundity (figure 6.3).

```
hist(subset(lathvertln_small, repstatus2 == 1)$feca2, breaks = 35,
    main = "Fecundity", xlab = "Intact seeds produced in occasion t")
```


## Fecundity



Figure 6.3: Histogram of fecundity in occasion $t$

We see that the distribution conforms to a classic count variable with a very low mean value. The first bar suggests that there may be too many zeros, as well. Let's now formally test our assumptions, that the mean equals the variance and that the number of zeros meets expectation. Both tests use chi-squared distribution-based approaches, with the zero-inflation test in particular based on van den Broek (1995). We will do this through a quality control assessment of the entire dataset for modeling, using $h f v_{\mathrm{p}} \mathrm{qc}()$, into which we need to specify which vital rates we intend to model later, and the names of some key variables.

```
hfv_qc(lathvertln_small, indiv = "individ", year = "year2", age = "obsage",
    vitalrates = c("surv", "obs", "size", "repst", "fec"))
> Survival:
>
> Data subset has 54 variables and 531 transitions.
>
> Variable alive3 has O missing values.
> Variable alive3 is a binomial variable.
>
>
```

```
> Observation status:
>
> Data subset has 54 variables and 431 transitions.
>
> Variable obsstatus3 has 0 missing values.
> Variable obsstatus3 is a binomial variable.
>
>
> Primary size:
>
> Data subset has 54 variables and 396 transitions.
    Variable sizea3 has O missing values.
    Variable sizea3 appears to be a floating point variable.
    379 elements are not integers.
    The minimum value of sizea3 is 0.7 and the maximum is 8.8.
    The mean value of sizea3 is 3.153 and the variance is 1.818.
    The value of the Shapiro-Wilk test of normality is 0.8367 with P = 8.608e-20.
    Variable sizea3 differs significantly from a Gaussian distribution.
    Variable sizea3 is fully positive, lacking even Os.
Reproductive status:
    Data subset has 54 variables and 396 transitions.
    Variable repstatus3 has 0 missing values.
    Variable repstatus3 is a binomial variable.
Fecundity:
    Data subset has 54 variables and 19 transitions.
    Variable feca2 has 0 missing values.
    Variable feca2 appears to be an integer variable.
    Variable feca2 is fully non-negative.
    Overdispersion test:
        Mean feca2 is 4.053
        The variance in feca2 is 61.05
        The probability of this dispersion level by chance assuming that
        the true mean feca2 = variance in feca2,
        and an alternative hypothesis of overdispersion, is 8.249e-152
        Variable feca2 is significantly overdispersed.
    Zero-inflation and truncation tests:
        Mean lambda in feca2 is 0.01738
        The actual number of 0s in feca2 is 10
```

$>\quad$ The expected number of 0 s in feca2 under the null hypothesis is 0.3302
$>\quad$ The probability of this deviation in 0 s from expectation by chance is $1.719 \mathrm{e}-69$
> Variable feca2 is significantly zero-inflated.

We see that fecundity is significantly overdispersed, and has a significant excess of zeros. So, we should use a zero-inflated negative binomial distribution here. Note that our output also suggests problems with using the Gaussian distribution for size, but we will ignore that at this time.

Next, let's create an ahistorical supplement table organizing the extra data that we need to incorporate into our matrices. Each row refers to a specific transition. The first two of these transitions are set to specific probabilities, which are the probabilities of germination and seed dormancy, estimated from a separate study. The final two terms are fecundity multipliers, which mark which transitions correspond to fecundity and provide information on what multiple of fecundity estimated via linear modeling applies to each case. Note that we can also include proxy transitions, in which we define a specific transition as being equal to another in the matrix. The latter approach is useful when some transitions cannot be estimated given a particular dataset, and so need to be set to other, proxy values that are estimable.

```
lathsupp2 <- supplemental(stage3 = c("Sd", "Sdl", "Sd", "Sdl"),
    stage2 = c("Sd", "Sd", "rep", "rep"), givenrate = c(0.345, 0.054, NA, NA),
    multiplier \(=c(N A, N A, 0.345,0.054)\), type \(=c(1,1,3,3)\),
    stageframe \(=\) lathframeln, historical = FALSE, stagebased \(=\) TRUE,
    agebased = TRUE)
lathsupp2
\begin{tabular}{lrrrrrrrrr}
\(>\) & stage3 & stage2 & stage1 & age2 & eststage3 & eststage2 & eststage1 & estage2 & givenrate \\
\(>\) & 1 & Sd & Sd & <NA> & NA & <NA> & <NA> & <NA> & NA
\end{tabular} 0.345
```

Next we will run the modelsearch function with the new vertical dataset. This function will develop our best-fit vital rate models for us. This function looks simple, but it automates several crucial and complex tasks in MPM analysis. Specifically, it automates 1) the building of global models for each vital rate requested, 2) the exhaustive construction of all reduced models, and 3 ) the selection of the best-fit models. In relation to our previous uses of this function in chapter 5, the most noteworthy difference is the inclusion of an age term (age = "obsage", which we know from looking at the summary of the vertical dataset). We will develop the full effects models here, which include main effects and all two-way interactions between age and other terms, including size and reproductive status. Note that we include age = "obsage", which tells modelsearch() what the name of the age variable is in our dataset, and test.age = TRUE, which tells the function to include the term in the global model of each vital rate. Let's start off by generating a set of vital rate models that covers the entire population.

```
lathmodelsln2 <- modelsearch(lathvertln_small, historical = FALSE,
    approach = "mixed", suite = "full", bestfit = "AICc&k", juvestimate = "Sdl",
    vitalrates = c("surv", "obs", "size", "repst", "fec"), sizedist = "gaussian",
    fecdist = "negbin", indiv = "individ", year = "year2", age = "obsage",
```

```
year.as.random = TRUE, patch.as.random = TRUE, show.model.tables = TRUE,
fec.zero = TRUE, test.age = TRUE, quiet = "partial")
```

Let's see a summary of the lefkoMod object that we have created.

```
summary(lathmodelsln2)
> This LefkoMod object includes }7\mathrm{ linear models.
Best-fit model criterion used: aicc&k
>
>
>
Survival model:
Generalized linear mixed model fit by maximum likelihood (Laplace
        Approximation) [glmerMod]
    Family: binomial ( logit )
    Formula: alive3 ~ obsage + (1 | year2) + (1 | individ)
        Data: subdata
            AIC BIC logLik deviance df.resid
182.9812 198.3785 -87.4906 174.9812 343
Random effects:
    Groups Name Std.Dev.
    individ (Intercept) 62.23
    year2 (Intercept) 0.00
Number of obs: 347, groups: individ, 247; year2, 2
Fixed Effects:
(Intercept) obsage
        39.46 -13.65
optimizer (Nelder_Mead) convergence code: O (OK) ; 0 optimizer warnings; 1 lme4 warnings
Observation model:
Generalized linear mixed model fit by maximum likelihood (Laplace
    Approximation) [glmerMod]
    Family: binomial ( logit )
Formula: obsstatus3 ~ obsage + repstatus2 + sizea2 + (1 | year2) + (1 |
            individ) + obsage:repstatus2 + obsage:sizea2
        Data: subdata
            AIC BIC logLik deviance df.resid
128.8981 158.3667 -56.4490 112.8981 286
Random effects:
    Groups Name Std.Dev.
    individ (Intercept) 63.3445
    year2 (Intercept) 0.1541
    Number of obs: 294, groups: individ, 203; year2, 2
    Fixed Effects:
        (Intercept) obsage repstatus2 sizea2
            35.382 3.959 23.704 2.048
    obsage:repstatus2 obsage:sizea2
    -18.983 -4.057
    optimizer (Nelder_Mead) convergence code: O (OK) ; O optimizer warnings; 1 lme4 warnings
```

```
>
>
Size model:
Linear mixed model fit by REML ['lmerMod']
Formula: sizea3 ~ sizea2 + (1 | year2) + (1 | individ)
            Data: subdata
REML criterion at convergence: 700.2483
Random effects:
Groups Name Std.Dev.
individ (Intercept) 5.346e-05
year2 (Intercept) 4.826e-01
Residual 8.850e-01
Number of obs: 266, groups: individ, 191; year2, 2
Fixed Effects:
(Intercept) sizea2
> 0.7088 0.7777
optimizer (nloptwrap) convergence code: O (OK) ; 0 optimizer warnings; 1 lme4 warnings
>
>
>
Secondary size model:
[1] 1
>
>
>
Tertiary size model:
[1] 1
>
>
>
Reproductive status model:
Generalized linear mixed model fit by maximum likelihood (Laplace
        Approximation) [glmerMod]
    Family: binomial ( logit )
Formula: repstatus3 ~ obsage + (1 | year2) + (1 | individ)
            Data: subdata
            AIC BIC logLik deviance df.resid
71.3773 85.7112 -31.6886 63.3773 262
Random effects:
Groups Name Std.Dev.
individ (Intercept) 83.68658
year2 (Intercept) 0.06152
Number of obs: 266, groups: individ, 191; year2, 2
Fixed Effects:
(Intercept) obsage
-44.60 15.32
optimizer (Nelder_Mead) convergence code: O (OK) ; 0 optimizer warnings; 2 lme4 warnings
>
>
>
> Fecundity model:
```

```
Formula:
feca2 ~ obsage + sizea2 + (1 | year2) + (1 | individ) + obsage:sizea2
Zero inflation: ~obsage + repstatus2 + (1 | year2) + (1 | individ)
Data: subdata
AIC BIC logLik df.resid
-749.9108 -738.5776 386.9554 7
Random-effects (co)variances:
Conditional model:
Groups Name Std.Dev.
year2 (Intercept) 4.491e-12
individ (Intercept) 4.078e-06
Zero-inflation model:
    Groups Name Std.Dev.
    year2 (Intercept) 2.421e-07
    individ (Intercept) 5.205e+01
Number of obs: 19 / Conditional model: year2, 2; individ, 16 / Zero-inflation model: year2, 2; indi
Dispersion parameter for nbinom2 family (): 1.13e+16
Fixed Effects:
Conditional model:
(Intercept) obsage sizea2 obsage:sizea2
            1.47543 -7.53684 -0.07609 1.20358
Zero-inflation model:
(Intercept) obsage repstatus2
            -16.13 22.85 -16.13
Juvenile survival model:
Generalized linear mixed model fit by maximum likelihood (Laplace
        Approximation) [glmerMod]
    Family: binomial ( logit )
Formula: alive3 ~ (1 | year2) + (1 | individ)
        Data: subdata
            AIC BIC logLik deviance df.resid
    215.1077 224.7525-104.5539 209.1077 181
Random effects:
    Groups Name Std.Dev.
    individ (Intercept) 6.015e-05
year2 (Intercept) 0.000e+00
Number of obs: 184, groups: individ, 184; year2, 2
Fixed Effects:
(Intercept)
1.07
optimizer (Nelder_Mead) convergence code: O (OK) ; O optimizer warnings; 1 lme4 warnings
```

>

```
>
>
> Juvenile observation model:
> Generalized linear mixed model fit by maximum likelihood (Laplace
        Approximation) [glmerMod]
Family: binomial ( logit )
Formula: obsstatus3 ~ (1 | year2) + (1 | individ)
            Data: subdata
                AIC BIC logLik deviance df.resid
    28.1925 36.9524 -11.0962 22.1925 134
Random effects:
Groups Name Std.Dev.
individ (Intercept) 5.587e+01
year2 (Intercept) 1.151e-05
Number of obs: 137, groups: individ, 137; year2, 2
Fixed Effects:
(Intercept)
            13.77
optimizer (Nelder_Mead) convergence code: O (OK) ; O optimizer warnings; 1 lme4 warnings
>
>
>
> Juvenile size model:
[1] 1
>
>
>
> Juvenile secondary size model:
[1] 1
>
>
>
> Juvenile tertiary size model:
[1] 1
>
>
>
> Juvenile reproduction model:
[1] 0
>
>
>
> Juvenile maturity model:
[1] 1
>
>
>
>
>
> Number of models in survival table: 17
>
```

```
> Number of models in observation table: 16
>
Number of models in size table: 18
>
Number of models in secondary size table: 1
Number of models in tertiary size table: 1
Number of models in reproduction status table: 16
Number of models in fecundity table: 287
Number of models in juvenile survival table: 1
Number of models in juvenile observation table: 1
Number of models in juvenile size table: 1
Number of models in juvenile secondary size table: 1
Number of models in juvenile tertiary size table: 1
Number of models in juvenile reproduction table: 1
Number of models in juvenile maturity table: 1
General model parameter names (column 1), and
specific names used in these models (column 2):
            parameter_names mainparams
                                    time t year2
                            individual individ
                                    patch patch
                                    alive in time t+1 surv3
4
> 5
> 6
>
sizec in time t+1 sizec3
9 reproductive status in time t+1 repst3
10
11
12
13
14
>
16
17
> 18
            fecundity in time t+1
                    fec3
                    fecundity in time t fec2
                                    sizea in time t size2
                    sizea in time t-1 size1
                                    sizeb in time t sizeb2
                    sizeb in time t-1 sizeb1
                    sizec in time t sizec2
                    sizec in time t-1 sizec1
    reproductive status in time t repst2
```

```
> 19 reproductive status in time t-1 repst1
> 20 maturity status in time t+1 matst3
>21 maturity status in time t matst2
2 2 ~ a g e ~ i n ~ t i m e ~ t ~ a g e
2 3 ~ d e n s i t y ~ i n ~ t i m e ~ t ~ d e n s i t y ~
> 24 individual covariate a in time t indcova2
> 25 individual covariate a in time t-1 indcova1
> 26 individual covariate b in time t indcovb2
> 27 individual covariate b in time t-1 indcovb1
> 28 individual covariate c in time t indcovc2
> 29 individual covariate c in time t-1 indcovc1
> 30 stage group in time t group2
>31 stage group in time t-1 group1
>
>
>
>
>
> Quality control:
>
> Survival model estimated with 247 individuals and 347 individual transitions.
> Survival model accuracy is 1.
> Observation status model estimated with 203 individuals and 294 individual transitions.
> Observation status model accuracy is 1.
> Primary size model estimated with }191\mathrm{ individuals and 266 individual transitions.
> Primary size model R-squared is 0.632.
> Secondary size model not estimated.
> Tertiary size model not estimated.
> Reproductive status model estimated with }191\mathrm{ individuals and 266 individual transitions.
> Reproductive status model accuracy is 1.
> Fecundity model estimated with 16 individuals and 19 individual transitions.
> Fecundity model R-squared is 0.96.
> Juvenile survival model estimated with 184 individuals and 184 individual transitions.
> Juvenile survival model accuracy is 0.745.
> Juvenile observation status model estimated with }137\mathrm{ individuals and 137 individual transitions.
> Juvenile observation status model accuracy is 1.
> Juvenile primary size model not estimated.
> Juvenile secondary size model not estimated.
> Juvenile tertiary size model not estimated.
> Juvenile reproductive status model not estimated.
> Juvenile maturity status model not estimated.
```

We see that age is influential in survival, observation, reproductive status, and fecundity, though not in other vital rates. Accuracy in our adult binomial models is high, but primary size for adults and juveniles is explained by models with low $\mathrm{R}^{2}$, suggesting problems.

Next, we will create a second set that includes patch as a random factor. This model set will allow us to explore patch dynamics in addition to the population dynamics of the previous set.

```
lathmodelsln2p <- modelsearch(lathvertln_small, historical = FALSE,
    approach = "mixed", suite = "full", bestfit = "AICc&k", juvestimate = "Sdl",
    vitalrates = c("surv", "obs", "size", "repst", "fec"), sizedist = "gaussian",
    fecdist = "negbin", indiv = "individ", patch = "patchid", year = "year2",
```

```
age = "obsage", year.as.random = TRUE, patch.as.random = TRUE,
show.model.tables = TRUE, fec.zero = TRUE, test.age = TRUE, quiet = "partial")
```

And a summary.

```
summary(lathmodelsln2p)
This LefkoMod object includes 7 linear models.
Best-fit model criterion used: aicc&k
>
>
>
Survival model:
Generalized linear mixed model fit by maximum likelihood (Laplace
        Approximation) [glmerMod]
    Family: binomial ( logit )
Formula: alive3 ~ obsage + repstatus2 + sizea2 + (1 | year2) + (1 | patchid) +
            (1 | individ) + obsage:sizea2
        Data: subdata
            AIC BIC logLik deviance df.resid
    288.3454 319.1400 -136.1727 272.3454 339
Random effects:
    Groups Name Std.Dev.
    individ (Intercept) 7.285e-01
    patchid (Intercept) 5.452e-01
    year2 (Intercept) 3.494e-05
Number of obs: 347, groups: individ, 247; patchid, 6; year2, 2
Fixed Effects:
(Intercept) obsage repstatus2 sizea2 obsage:sizea2
6.0406 -2.8058 20.6037 0.6609
optimizer (Nelder_Mead) convergence code: O (OK) ; 0 optimizer warnings; 1 lme4 warnings
Observation model:
Generalized linear mixed model fit by maximum likelihood (Laplace
        Approximation) [glmerMod]
    Family: binomial ( logit )
Formula: obsstatus3 ~ obsage + repstatus2 + sizea2 + (1 | year2) + (1 |
            patchid) + (1 | individ) + obsage:sizea2 + repstatus2:sizea2
        Data: subdata
            AIC BIC logLik deviance df.resid
131.4644 164.6166 -56.7322 113.4644 285
Random effects:
Groups Name Std.Dev.
    individ (Intercept) 43.8384
    patchid (Intercept) 0.9045
    year2 (Intercept) 0.0000
Number of obs: 294, groups: individ, 203; patchid, 6; year2, 2
Fixed Effects:
        (Intercept)
            21.103
                        obsage repstatus2
                        41.223 4.327
    obsage:sizea2 repstatus2:sizea2
```

```
> -4.843 -6.055
optimizer (Nelder_Mead) convergence code: O (OK) ; 0 optimizer warnings; 1 lme4 warnings
>
>
>
Size model:
Linear mixed model fit by REML ['lmerMod']
Formula: sizea3 ~ sizea2 + (1 | year2) + (1 | patchid) + (1 | individ)
            Data: subdata
REML criterion at convergence: 700.2483
Random effects:
Groups Name Std.Dev.
individ (Intercept) 0.0000
patchid (Intercept) 0.0000
year2 (Intercept) 0.4826
Residual 0.8850
Number of obs: 266, groups: individ, 191; patchid, 6; year2, 2
Fixed Effects:
(Intercept) sizea2
    0.7088 0.7777
optimizer (nloptwrap) convergence code: O (OK) ; O optimizer warnings; 1 lme4 warnings
>
>
Secondary size model:
[1] 1
```



```
>
Tertiary size model:
[1] 1
Reproductive status model:
Generalized linear mixed model fit by maximum likelihood (Laplace
    Approximation) [glmerMod]
    Family: binomial ( logit )
Formula: repstatus3 ~ obsage + (1 | year2) + (1 | patchid) + (1 | individ)
    Data: subdata
        AIC BIC logLik deviance df.resid
    73.3774 91.2949 -31.6887 63.3774 261
Random effects:
    Groups Name Std.Dev.
individ (Intercept) 83.45090
patchid (Intercept) 0.06441
year2 (Intercept) 0.15574
Number of obs: 266, groups: individ, 191; patchid, 6; year2, 2
Fixed Effects:
(Intercept) obsage
-44.58 15.32
```

```
optimizer (Nelder_Mead) convergence code: O (OK) ; O optimizer warnings; 2 lme4 warnings
>
>
>
Fecundity model:
Formula: feca2 ~ sizea2 + (1 | year2) + (1 | individ)
Zero inflation: ~(1 | year2) + (1 | individ)
Data: subdata
    AIC BIC logLik df.resid
79.29325 86.84876 -31.64662 11
Random-effects (co)variances:
Conditional model:
    Groups Name Std.Dev.
    year2 (Intercept) 4.895e-06
    individ (Intercept) 1.146e-04
Zero-inflation model:
    Groups Name Std.Dev.
    year2 (Intercept) 0.003791
    individ (Intercept) 1.771162
Number of obs: 19 / Conditional model: year2, 2; individ, 16 / Zero-inflation model: year2, 2; indi
Dispersion parameter for nbinom2 family (): 3.12e+08
Fixed Effects:
```



```
Conditional model:
(Intercept) sizea2
        -6.265 1.157
Zero-inflation model:
(Intercept)
        -0.1905
    Juvenile survival model:
Generalized linear mixed model fit by maximum likelihood (Laplace
        Approximation) [glmerMod]
    Family: binomial ( logit )
    Formula: alive3 ~ (1 | year2) + (1 | patchid) + (1 | individ)
        Data: subdata
            AIC BIC logLik deviance df.resid
    217.1077 229.9675 -104.5539 209.1077 180
Random effects:
    Groups Name Std.Dev.
    individ (Intercept) 2.663e-07
    patchid (Intercept) 0.000e+00
    year2 (Intercept) 0.000e+00
Number of obs: 184, groups: individ, 184; patchid, 6; year2, 2
```

```
> Fixed Effects:
> (Intercept)
> 1.07
optimizer (Nelder_Mead) convergence code: O (OK) ; 0 optimizer warnings; 1 lme4 warnings
>
>
>
Juvenile observation model:
Generalized linear mixed model fit by maximum likelihood (Laplace
        Approximation) [glmerMod]
    Family: binomial ( logit )
Formula: obsstatus3 ~ (1 | year2) + (1 | patchid) + (1 | individ)
            Data: subdata
            AIC BIC logLik deviance df.resid
    30.1925 41.8724 -11.0962 22.1925 133
Random effects:
    Groups Name Std.Dev.
    individ (Intercept) 5.587e+01
    patchid (Intercept) 3.555e-05
    year2 (Intercept) 0.000e+00
Number of obs: 137, groups: individ, 137; patchid, 6; year2, 2
Fixed Effects:
(Intercept)
    13.77
optimizer (Nelder_Mead) convergence code: O (OK) ; 0 optimizer warnings; 1 lme4 warnings
>
>
>
> Juvenile size model:
[1] 1
Juvenile secondary size model:
[1] 1
>
>
>
> Juvenile tertiary size model:
[1] 1
>
>
>
> Juvenile reproduction model:
[1] 0
>
>
>
> Juvenile maturity model:
[1] 1
>
```

```
>
>
>
>
Number of models in survival table: 17
Number of models in observation table: 16
Number of models in size table: 18
Number of models in secondary size table: 1
Number of models in tertiary size table: 1
Number of models in reproduction status table: 18
Number of models in fecundity table: 58
Number of models in juvenile survival table: 1
Number of models in juvenile observation table: 1
Number of models in juvenile size table: 1
Number of models in juvenile secondary size table: 1
Number of models in juvenile tertiary size table: 1
Number of models in juvenile reproduction table: 1
Number of models in juvenile maturity table: 1
General model parameter names (column 1), and
specific names used in these models (column 2):
            parameter_names mainparams
                    time t year2
                    individual individ
                                    patch patch
4 ralive in time t+1 surv3
6 sizea in time t+1 size3
7 sizeb in time t+1 sizeb3
8 sizec in time t+1 sizec3
> 9 reproductive status in time t+1 repst3
10 fecundity in time t+1 fec3
11 fecundity in time t fec2
12 sizea in time t size2
```

```
> 13
> 14
> 15
> 16
> 17
> 18
> 19
> 20
> 21
> 22
> 23
> 24
24 individual covariate a in time t
> 25 individual covariate a in time t-1
> 26 individual covariate b in time t
> 27 individual covariate b in time t-1
> 28 individual covariate c in time t ind
> 29 individual covariate c in time t-1 indcovc1
> 30 stage group in time t group2
> 31 stage group in time t-1 group1
>
>
>
>
>
> Quality control:
>
> Survival model estimated with 247 individuals and 347 individual transitions.
> Survival model accuracy is 0.847.
> Observation status model estimated with 203 individuals and 294 individual transitions.
> Observation status model accuracy is 1.
> Primary size model estimated with }191\mathrm{ individuals and 266 individual transitions.
> Primary size model R-squared is 0.632.
> Secondary size model not estimated.
> Tertiary size model not estimated.
> Reproductive status model estimated with }191\mathrm{ individuals and 266 individual transitions.
> Reproductive status model accuracy is 1.
> Fecundity model estimated with 16 individuals and 19 individual transitions.
> Fecundity model R-squared is 0.834.
> Juvenile survival model estimated with 184 individuals and 184 individual transitions.
> Juvenile survival model accuracy is 0.745.
> Juvenile observation status model estimated with }137\mathrm{ individuals and 137 individual transitions.
> Juvenile observation status model accuracy is 1.
> Juvenile primary size model not estimated.
> Juvenile secondary size model not estimated.
> Juvenile tertiary size model not estimated.
> Juvenile reproductive status model not estimated.
> Juvenile maturity status model not estimated.
```

Note that including patch as a random factor changed the best-fit model for fecundity, which no longer includes observed age.

Next, we will create the ahistorical sets of matrices. We will match the ahistorical age-by-stage matrix estimation function, aflefko2(), with the appropriate ahistorical input, including the ahistor-
ical lefkoMod objects lathmodelsln2 and lathmodelsln2p. Model sets that include historical terms should not be used to create ahistorical matrices, since the coefficients in the best-fit models are estimated assuming a specific model structure that either relies on historical terms or does not. Historical vital rate models may yield biased results if used to construct ahistorical matrices. Also note that lefko3 does not currently allow the construction of historical age-by-stage MPMs. Let's start off by developing the population-only MPM.

```
lathmat2age <- aflefko2(year = "all", stageframe = lathframeln,
    modelsuite = lathmodelsln2, data = lathvertln_small, supplement = lathsupp2,
    continue = TRUE, reduce = FALSE)
summary(lathmat2age)
>
> This ahistorical lefkoMat object contains 2 matrices.
>
> Each matrix is square with 63 rows and columns, and a total of 3969 elements.
> A total of }1428\mathrm{ survival transitions were estimated, with }714\mathrm{ per matrix.
> A total of 72 fecundity transitions were estimated, with 36 per matrix.
> This lefkoMat object covers 1 population, 1 patch, and 2 time steps.
>
The dataset contains a total of 345 unique individuals and 531 unique transitions.
>
Vital rate modeling quality control:
>
Survival estimated with }247\mathrm{ individuals and 347 individual transitions.
Observation estimated with }203\mathrm{ individuals and 294 individual transitions.
Primary size estimated with }191\mathrm{ individuals and 266 individual transitions.
Secondary size transition not estimated.
Tertiary size transition not estimated.
Reproductive status estimated with }191\mathrm{ individuals and 266 individual transitions.
Fecundity estimated with 16 individuals and 19 individual transitions.
Juvenile survival estimated with }184\mathrm{ individuals and }184\mathrm{ individual transitions.
Juvenile observation estimated with }137\mathrm{ individuals and }137\mathrm{ individual transitions.
Juvenile primary size transition not estimated.
Juvenile secondary size transition not estimated.
Juvenile tertiary size transition not estimated.
Juvenile reproduction probability not estimated.
Juvenile maturity transition probability not estimated.
Survival probability sum check (each matrix represented by column in order):
> [,1] [,2]
Min. 0.000 0.000
1st Qu. 0.000 0.000
Median 0.185 0.185
Mean 0.352 0.366
3rd Qu. 0.858 0.962
Max. 1.000 1.000
```

This first model set led to the development of two matrices, because although there are four years of data, we have limited our use of the data to individuals first seen from the second year on. Function aflefko2() has assessed patterns in the data set and has found that the maximum age is three years (although the summary of the input dataset shows a maximum obsage of two years, this value refers
to age in time $t$, and so the maximum age observed in the dataset is actually three years). We have informed the function that this age is not terminal and that the demography of that age should continue onward (continue $=$ TRUE). This has resulted in a block matrix with three ages and 21 stages, and so 63 age-stage combinations and 63 rows and columns. Of course, this particular plant species is actually long-lived, and so there may very well be further vital rate variability in across the lifespan of the plant. However, as our dataset only includes four years of study and we do not have absolute ages for any plant, we can only include four years of relative age at best. Thus, our settings are actually the most parsimonious under the circumstances.

The quality control section gives us a sense of the amount of data used to model each vital rate, and also shows us that the survival-transition ( U ) matrices are composed entirely of proper probabilities yielding stage survival probability falling between 0.0 and 1.0. Matrix estimation can sometimes create spurious values, such as stage survival greater than 1.0. Such values can occur for a variety of reasons, but the most common is the inclusion through a supplement table or overwrite table of externallydetermined survival probabilities that are too high. Make sure to check your matrix column sums each time you estimate MPMs to prevent this problem. Survival greater than 1.0 can lead to strange effects on metrics of population dynamics.

Let's now develop the patch-level MPMs.

```
lathmat2agep <- aflefko2(year = "all", patch = "all", stageframe = lathframeln,
    modelsuite = lathmodelsln2p, data = lathvertln_small, supplement = lathsupp2,
    continue = TRUE, reduce = FALSE)
summary(lathmat2agep)
>
> This ahistorical lefkoMat object contains 12 matrices.
>
> Each matrix is square with 63 rows and columns, and a total of 3969 elements.
> A total of }8508\mathrm{ survival transitions were estimated, with }709\mathrm{ per matrix.
> A total of 432 fecundity transitions were estimated, with 36 per matrix.
> This lefkoMat object covers 1 population, 6 patches, and 2 time steps.
>
> The dataset contains a total of 345 unique individuals and 531 unique transitions.
>
> Vital rate modeling quality control:
>
> Survival estimated with 247 individuals and 347 individual transitions.
> Observation estimated with }203\mathrm{ individuals and 294 individual transitions.
> Primary size estimated with }191\mathrm{ individuals and 266 individual transitions.
Secondary size transition not estimated.
Tertiary size transition not estimated.
Reproductive status estimated with }191\mathrm{ individuals and 266 individual transitions.
Fecundity estimated with 16 individuals and 19 individual transitions.
Juvenile survival estimated with }184\mathrm{ individuals and 184 individual transitions.
Juvenile observation estimated with }137\mathrm{ individuals and }137\mathrm{ individual transitions.
Juvenile primary size transition not estimated.
Juvenile secondary size transition not estimated.
Juvenile tertiary size transition not estimated.
Juvenile reproduction probability not estimated.
Juvenile maturity transition probability not estimated.
Survival probability sum check (each matrix represented by column in order):
> [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11] [,12]
```

```
> Min. 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000
> 1st Qu. 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000
> Median 0.769 0.833 0.684 0.716 0.769 0.822 0.593 0.621 0.686 0.718 0.769 0.820
> Mean 0.544 0.558 0.526 0.539 0.542 0.556 0.513 0.524 0.527 0.539 0.542 0.556
> 3rd Qu. 0.994 0.998 0.993 0.996 0.994 0.998 0.992 0.994 0.994 0.996 0.994 0.998
> Max. 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000
```

This second model set led to the development of 12 matrices, reflecting our subset number of three years and six patches. The rest of the output looks quite similar and even the survival-transition matrix column sum summaries look extremely similar, suggesting little impact of patch.

We can get a sense of what these matrices look like by visualizing them. Let's use the image3() function to look at just one (figure 6.4).

```
image3(lathmat2age, used = 1)
> [[1]]
```



Figure 6.4: Visualization of 1st A Matrix. Red area corresponds to non-zero elements
The clear squares refer to zero elements, and the red elements refer to non-zero values corresponding to survival transitions and fecundity. The vast number of zeros may be surprising, but this matrix is a supermatrix organized by age first, with stage organizing within-age blocks. The first age is age 1, which cannot be adult, and so we find zeros in adult stages at age 1. The adult block occurs from age 2 , and this block can perpetuate indefinitely. The number of elements estimated is greater now than in the typical ahistorical MPM, because now we have added age as a major factor for analysis. This matrix is overwhelmingly composed of elements that must be zero, and so it is a rather sparse matrix $((714+36) / 3969=18.9 \%$ of elements $)$.

Given the amount of white space, we might prefer to remove impossible age-stage combinations and reduce the size of the matrices. We can do this by recreating our matrices with reduce = TRUE. Let's try that with the main population set.

```
lathmat2age_red <- aflefko2(year = "all", stageframe = lathframeln,
    modelsuite = lathmodelsln2, data = lathvertln_small, supplement = lathsupp2,
    continue = TRUE, reduce = TRUE)
```

```
summary(lathmat2age_red)
>
> This ahistorical lefkoMat object contains 2 matrices.
>
> Each matrix is square with 42 rows and columns, and a total of 1764 elements.
> A total of }1428\mathrm{ survival transitions were estimated, with }714\mathrm{ per matrix.
> A total of 72 fecundity transitions were estimated, with 36 per matrix.
> This lefkoMat object covers 1 population, 1 patch, and 2 time steps.
>
> The dataset contains a total of 345 unique individuals and 531 unique transitions.
>
> Vital rate modeling quality control:
>
> Survival estimated with 247 individuals and 347 individual transitions.
> Observation estimated with }203\mathrm{ individuals and 294 individual transitions.
> Primary size estimated with }191\mathrm{ individuals and 266 individual transitions.
> Secondary size transition not estimated.
> Tertiary size transition not estimated.
> Reproductive status estimated with }191\mathrm{ individuals and 266 individual transitions.
> Fecundity estimated with 16 individuals and 19 individual transitions.
> Juvenile survival estimated with }184\mathrm{ individuals and 184 individual transitions.
> Juvenile observation estimated with }137\mathrm{ individuals and 137 individual transitions.
> Juvenile primary size transition not estimated.
> Juvenile secondary size transition not estimated.
> Juvenile tertiary size transition not estimated.
> Juvenile reproduction probability not estimated.
> Juvenile maturity transition probability not estimated.
>
> Survival probability sum check (each matrix represented by column in order):
> [,1] [,2]
> Min. 0.000 0.000
> 1st Qu. 0.185 0.185
> Median 0.345 0.345
> Mean 0.529 0.549
> 3rd Qu. 1.000 1.000
> Max. 1.000 1.000
```

This exercise has eliminated 21 rows and columns, yielding a matrix with 42 rows and 42 columns. The total number of estimated elements has not changed, meaning that our matrices are now much denser $((714+36) / 1764=42.5 \%)$. Let's take a look at an image of the first matrix (figure 6.5).

```
image3(lathmat2age_red, used = 1)
> [[1]]
```

Some of the white space has been reduced, and we see greater coverage of the matrix by non-zero elements.

We can see the order of ages and stages using the agestages element of the lefkoMat object we produced, as below. Note that our matrix is 42 rows by 42 columns, and this object gives us the exact order used.


Figure 6.5: Visualization of 1st A Matrix (reduced). Red area corresponds to non-zero elements


| $>$ | 30 | 9 | $\mathrm{Sz6nr}$ |
| ---: | ---: | ---: | ---: |
| $>$ | 3 |  |  |
| $>31$ | 10 | $\mathrm{Sz7nr}$ | 3 |
| $>32$ | 11 | Sz 8 nr | 3 |
| $>33$ | 12 | Sz 9 nr | 3 |
| $>34$ | 13 | $\mathrm{Sz1r}$ | 3 |
| $>35$ | 14 | Sz 2 r | 3 |
| $>36$ | 15 | $\mathrm{Sz3r}$ | 3 |
| $>37$ | 16 | $\mathrm{Sz4r}$ | 3 |
| $>38$ | 17 | $\mathrm{Sz5r}$ | 3 |
| $>39$ | 18 | $\mathrm{Sz6r}$ | 3 |
| $>40$ | 19 | $\mathrm{Sz7r}$ | 3 |
| $>41$ | 20 | Sz 8 r | 3 |
| $>42$ | 21 | $\mathrm{Sz9r}$ | 3 |

Now let's estimate the element-wise arithmetic mean matrices. The first lefkoMat object created will include a single mean matrix, while the second will include six mean matrices for the patches followed by a grand mean matrix, yielding a total of seven matrices.

```
lathmat2mean <- lmean(lathmat2age)
lathmat2pmean <- lmean(lathmat2agep)
summary(lathmat2mean)
>
> This ahistorical lefkoMat object contains 1 matrix.
>
> Each matrix is square with 63 rows and columns, and a total of 3969 elements.
> A total of }718\mathrm{ survival transitions were estimated, with }718\mathrm{ per matrix.
> A total of 36 fecundity transitions were estimated, with 36 per matrix.
> This lefkoMat object covers 1 population, 1 patch, and O time steps.
>
> The dataset contains a total of 345 unique individuals and 531 unique transitions.
>
> Vital rate modeling quality control:
>
> Survival estimated with 247 individuals and 347 individual transitions.
> Observation estimated with 203 individuals and 294 individual transitions.
Primary size estimated with }191\mathrm{ individuals and 266 individual transitions.
Secondary size transition not estimated.
Tertiary size transition not estimated.
Reproductive status estimated with }191\mathrm{ individuals and 266 individual transitions.
Fecundity estimated with 16 individuals and 19 individual transitions.
Juvenile survival estimated with }184\mathrm{ individuals and 184 individual transitions.
Juvenile observation estimated with }137\mathrm{ individuals and 137 individual transitions.
Juvenile primary size transition not estimated.
Juvenile secondary size transition not estimated.
Juvenile tertiary size transition not estimated.
Juvenile reproduction probability not estimated.
Juvenile maturity transition probability not estimated.
Survival probability sum check (each matrix represented by column in order):
[,1]
Min. 0.000
```

```
> 1st Qu. 0.000
> Median 0.185
> Mean 0.359
> 3rd Qu. 0.910
> Max. 1.000
summary(lathmat2pmean)
>
> This ahistorical lefkoMat object contains 7 matrices.
>
> Each matrix is square with 63 rows and columns, and a total of 3969 elements.
> A total of 4991 survival transitions were estimated, with 713 per matrix.
> A total of 252 fecundity transitions were estimated, with 36 per matrix.
> This lefkoMat object covers 1 population, }7\mathrm{ patches, and 0 time steps.
>
> The dataset contains a total of 345 unique individuals and 531 unique transitions.
>
> Vital rate modeling quality control:
>
> Survival estimated with }247\mathrm{ individuals and 347 individual transitions.
> Observation estimated with 203 individuals and 294 individual transitions.
> Primary size estimated with }191\mathrm{ individuals and 266 individual transitions.
Secondary size transition not estimated.
Tertiary size transition not estimated.
Reproductive status estimated with }191\mathrm{ individuals and 266 individual transitions.
Fecundity estimated with 16 individuals and 19 individual transitions.
Juvenile survival estimated with }184\mathrm{ individuals and }184\mathrm{ individual transitions.
Juvenile observation estimated with }137\mathrm{ individuals and 137 individual transitions.
> Juvenile primary size transition not estimated.
> Juvenile secondary size transition not estimated.
> Juvenile tertiary size transition not estimated.
Juvenile reproduction probability not estimated.
Juvenile maturity transition probability not estimated.
Survival probability sum check (each matrix represented by column in order):
> [,1] [,2] [,3] [,4] [,5] [,6] [,7]
>Min. 0.000 0.000 0.000 0.000 0.000 0.000 0.000
> 1st Qu. 0.000 0.000 0.000 0.000 0.000 0.000 0.000
Median 0.815 0.700 0.804 0.607 0.702 0.802 0.738
Mean 0.551 0.533 0.549 0.519 0.533 0.549 0.539
3rd Qu. 0.997 0.995 0.996 0.993 0.995 0.996 0.995
> Max. 1.000 1.000 1.000 1.000 1.000 1.000 1.000
```

We see one overall population mean in the first case, and a set of six patch-level means and one population mean in the second case. Note that the population mean in each case should be a little different, because the population mean in the second set weights each patch equally, while the overall population mean in the first case weights individual transitions equally regardless of patch of origin.

### 6.3 Developing raw age-by-stage MPMs

Now let's focus our attention on raw (empirical) age-by-stage MPMs. For this, we will use the function arlefko2(). Raw MPMs generally require a great deal more data to parameterize properly. We will
also need a stageframe with fewer stages, in order to limit the number of artificial zeros entering our matrices. Let's start with the following life history model, which assumes a grand total of seven stages.


Figure 6.6: Life history model of Lathyrus vernus using log leaf volume as the size classification metric

Let's now look at our stageframe for this model. The actual stage definitions in terms of size bins are no longer based on the log leaf volume. Instead, they are based on the raw leaf volume. In order to deal with the fact that small plants are far more common than large plants, the bins have increasing widths with increasing leaf size. We will also lump all reproduction into a single flowering class that can include individuals of any size, provided that they have sprouted.

```
sizevector <- c(0, 100, 13, 127, 3730, 3800, 0)
stagevector <- c("Sd", "Sdl", "VSm", "Sm", "VLa", "Flo", "Dorm")
repvector <- c(0, 0, 0, 0, 0, 1, 0)
obsvector <- c(0, 1, 1, 1, 1, 1, 0)
matvector <- c(0, 0, 1, 1, 1, 1, 1)
immvector <- c(1, 1, 0, 0, 0, 0, 0)
propvector <- c(1, 0, 0, 0, 0, 0, 0)
indataset <- c(0, 1, 1, 1, 1, 1, 1)
binvec <- c(0, 100, 11, 103, 3500, 3800, 0.5)
```



Let's now restandardize our core dataset, using the new stageframe for stage assignment and the original size variable for size classification. We will also cut sightings of individuals in the first year, as before, to result in a dataset in which individuals are ages that we can be reasonably sure about.

```
lathvert_raw <- verticalize3(lathyrus, noyears = 4, firstyear = 1988,
    patchidcol = "SUBPLOT", individcol = "indiv_id", blocksize = 9,
    juvcol = "Seedling1988", sizeacol = "Volume88", repstracol = "FCODE88",
    fecacol = "Intactseed88", deadacol = "Dead1988", nonobsacol = "Dormant1988",
    stageassign = lathframe_raw, stagesize = "sizea", censorcol = "Missing1988",
    censorkeep = NA, censorRepeat = TRUE, censor = TRUE)
lathvert_raw_small <- subset(lathvert_raw, firstseen > 1988)
dim(lathvert_raw_small)
> [1] 531 54
summary_hfv(lathvert_raw_small)
>
> This hfv dataset contains 531 rows, 54 variables, 1 population,
> 6 patches, 345 individuals, and 2 time steps.
> rowid popid patchid individ
>Min. : 30.0 Length:531 Min. :1.000 Length:531
> 1st Qu.: 301.0 Class :character 1st Qu.:2.000 Class :character
> Median : 582.0 Mode :character Median :3.000 Mode :character
> Mean : 585.3 Mean :3.077
> 3rd Qu.: 800.5 3rd Qu.:4.000
> Max. :1097.0 Max. :6.000
>
> year2 firstseen lastseen obsage obslifespan
> Min. :1989 Min. :1989 Min. :1989 Min. :1.00 Min. :0.000
> 1st Qu.:1989 1st Qu.:1989 1st Qu.:1990 1st Qu.:1.00 1st Qu.:1.000
> Median :1990 Median :1989 Median :1991 Median :1.00 Median :2.000
Mean :1990 Mean :1989 Mean :1991 Mean :1.35 Mean :1.367
3rd Qu.:1990 3rd Qu.:1989 3rd Qu.:1991 3rd Qu.:2.00 3rd Qu.:2.000
Max. :1990 Max. :1990 Max. :1991 Max. :2.00 Max. :2.000
sizea1 size1added repstra1 repstr1added
Min. : 3.60 Min. : 0.00 Min. :0.0000 Min. :0.00000
1st Qu.: 9.00 1st Qu.: 0.00 1st Qu.:0.0000 1st Qu.:0.00000
Median : 12.60 Median : 0.00 Median :0.0000 Median :0.00000
Mean : 168.22 Mean : 58.92 Mean :0.0591 Mean :0.02072
3rd Qu.: 65.22 3rd Qu.: 9.00 3rd Qu.:0.0000 3rd Qu.:0.00000
Max. :4394.20 Max. :4394.20 Max. :1.0000 Max. :1.00000
NA's :345 NA's :345
    feca1 fec1added censor1 juvgiven1 obsstatus1
Min. : 0 Min. : 0.0000 Min. :0 Min. :0.000 Min. :0.0000
1st Qu.: 0 1st Qu.: 0.0000 1st Qu.:0 1st Qu.:0.000 1st Qu.:0.0000
Median : 4 Median : 0.0000 Median :0 Median :0.000 Median :0.0000
Mean : 6 Mean : 0.1243 Mean :0 Mean :0.162 Mean :0.3503
3rd Qu.: 6 3rd Qu.: 0.0000 3rd Qu.:0 3rd Qu.:0.000 3rd Qu.:1.0000
Max. :34 Max. :34.0000 Max. :0 Max. :1.000 Max. :1.0000
NA's :520
        repstatus1
Min. :0.00000
1st Qu.:0.00000
Median :0.00000
Mean :0.02072 Mean :0.01318
\begin{tabular}{ll}
\multicolumn{2}{c}{ matstatus1 }
\end{tabular}\(\frac{2}{c}\) alive1 \(\quad: 0.0000\)
```



```
```

3rd Qu.:0.00000 3rd Qu.:0.00000 3rd Qu.:0.0000 3rd Qu.:1.0000

```
```

3rd Qu.:0.00000 3rd Qu.:0.00000 3rd Qu.:0.0000 3rd Qu.:1.0000
Max. :1.00000 Max. :1.00000 Max. :1.0000 Max. :1.0000
Max. :1.00000 Max. :1.00000 Max. :1.0000 Max. :1.0000
stage1 stage1index sizea2
stage1 stage1index sizea2
Length:531
Length:531
Class :character
Class :character
> Mode :character
> Mode :character
Min. :0.00 Min. : 1.80
Min. :0.00 Min. : 1.80
1st Qu.:0.00 1st Qu.: 9.00 1st Qu.: 9.00
1st Qu.:0.00 1st Qu.: 9.00 1st Qu.: 9.00
Median :0.00 Median : 12.60 Median : 12.60
Median :0.00 Median : 12.60 Median : 12.60
Mean :1.09 Mean : 99.38 Mean : 97.89
Mean :1.09 Mean : 99.38 Mean : 97.89
3rd Qu.:2.00 3rd Qu.: 40.80 3rd Qu.: 37.40
3rd Qu.:2.00 3rd Qu.: 40.80 3rd Qu.: 37.40
Max. :6.00 Max. :4394.20 Max. :4394.20
Max. :6.00 Max. :4394.20 Max. :4394.20
Max. :6.00 Max. :4394.20 Max. :4394.20
Max. :6.00 Max. :4394.20 Max. :4394.20
repstra2 repstr2added
repstra2 repstr2added
repstra2
repstra2
1st Qu.:0.00000

```
```

1st Qu.:0.00000

```
```




```
```

Median :0.00000

```
```

Median :0.00000
3rd Qu.:0.00000 3rd Qu.:0.00000 3rd Qu.: 6.000 3rd Qu.: 0.000
3rd Qu.:0.00000 3rd Qu.:0.00000 3rd Qu.: 6.000 3rd Qu.: 0.000
Max. :1.00000 Max. :1.00000 Max. :34.000 Max. :34.000
Max. :1.00000 Max. :1.00000 Max. :34.000 Max. :34.000
NA's :8 NA's :512
NA's :8 NA's :512
censor2 juvgiven2 obsstatus2 repstatus2
censor2 juvgiven2 obsstatus2 repstatus2
Min. :0 Min. :0.0000 Min. :0.0000 Min. :0.00000
Min. :0 Min. :0.0000 Min. :0.0000 Min. :0.00000
1st Qu.:0 1st Qu.:0.0000 1st Qu.:1.0000 1st Qu.:0.00000
1st Qu.:0 1st Qu.:0.0000 1st Qu.:1.0000 1st Qu.:0.00000
Median :0 Median :0.0000 Median :1.0000 Median :0.00000
Median :0 Median :0.0000 Median :1.0000 Median :0.00000
Mean :0 Mean :0.3465 Mean :0.9849 Mean :0.03578
Mean :0 Mean :0.3465 Mean :0.9849 Mean :0.03578
3rd Qu.:0 3rd Qu.:1.0000 3rd Qu.:1.0000 3rd Qu.:0.00000
3rd Qu.:0 3rd Qu.:1.0000 3rd Qu.:1.0000 3rd Qu.:0.00000
Max. :0 Max. :1.0000 Max. :1.0000 Max. :1.00000

```
```

Max. :0 Max. :1.0000 Max. :1.0000 Max. :1.00000

```
```
































| Min. :0 | Min. $\quad 0.0000$ | Min. 00.0000 | Min. 00.00000 |
| :---: | :---: | :---: | :---: |
| 1st Qu.:0 | 1st Qu.:0.0000 | 1st Qu.:1.0000 | 1st Qu.:0.00000 |
| Median :0 | Median :0.0000 | Median :1.0000 | Median :0.00000 |
| Mean : 0 | Mean :0.3465 | Mean :0.9849 | Mean :0.03578 |
| 3rd Qu.:0 | 3rd Qu.:1.0000 | 3rd Qu.:1.0000 | 3rd Qu.:0.00000 |
| Max. : 0 | Max. $: 1.0000$ | Max. $: 1.0000$ | Max. 1.00000 |

```
    Min. : 1.80 Min. : 0.00
```

    Min. : 1.80 Min. : 0.00
    repstr2added feca2
repstr2added feca2
fec2added
fec2added
Min. :0.00000 Min. : 0.000 Min. : 0.000
Min. :0.00000 Min. : 0.000 Min. : 0.000
1st Qu.:0.00000 1st Qu.: 0.000 1st Qu.: 0.000
1st Qu.:0.00000 1st Qu.: 0.000 1st Qu.: 0.000
Median :0.00000
Median :0.00000
fecstatus2 matstatus2 alive2 stage2
fecstatus2 matstatus2 alive2 stage2
Min. :0.00000 Min. :0.0000 Min. :1 Length:531
Min. :0.00000 Min. :0.0000 Min. :1 Length:531
1st Qu.:0.00000 1st Qu.:0.0000
1st Qu.:0.00000 1st Qu.:0.0000
1st Qu.:1 Class :character
1st Qu.:1 Class :character
Median :1 Mode :character
Median :1 Mode :character
Mean :1
Mean :1
3rd Qu.:0.00000 3rd Qu.:1.0000 3rd Qu.:1
3rd Qu.:0.00000 3rd Qu.:1.0000 3rd Qu.:1
Mrd Qu.:0.00000
Mrd Qu.:0.00000
Max. :1.00000 Max. :1.0000 Max. :1
Max. :1.00000 Max. :1.0000 Max. :1
*

```
    *
```
























```
Median :0.00000 
```

Median :0.00000
Min. : 0.000 Min. : 0.000
Min. : 0.000 Min. : 0.000
size2added
size2added
repstra2 (anmon
repstra2 (anmon
Mean :0.01695 Mean :0.6535 Mean :

```
Mean :0.01695 Mean :0.6535 Mean :
```

$>$
$>$
$>$
$>$
$>$
$>$
$>$
$>$
$>$
$>$
$>$
$>$
$>$
$>$
$>$
$>$
$>$
$>$
$>$
$>$
$>$
$>$
$>$
$>$
$>$
$>$
$>$
$>$
$>$
$>$
$>$
$>$
$>$
$>$
$>$
$>$
$>$
$>$
$>$
$>$
$>$
$>$
$>$
$>$
$>$
$>$
$>$
$>$
$>$

| > | obsstatus3 | repstatus3 | fecstatus3 | matstatus3 |
| :---: | :---: | :---: | :---: | :---: |
| > | Min. $: 0.0000$ | Min. :0.00000 | Min. :0.00000 | Min. :1 |
| $>$ | 1st Qu.:0.0000 | 1st Qu.:0.00000 | 1st Qu.:0.00000 | 1st Qu.:1 |
| > | Median :1.0000 | Median :0.00000 | Median :0.00000 | Median :1 |
| > | Mean :0.7458 | Mean :0.04143 | Mean :0.01318 | Mean :1 |
| > | 3rd Qu.:1.0000 | 3rd Qu.:0.00000 | 3rd Qu.:0.00000 | 3rd Qu.:1 |
| > | Max. $: 1.0000$ | Max. $: 1.00000$ | Max. 1.00000 | Max. $: 1$ |
| > |  |  |  |  |
| $>$ | alive3 | stage3 | stage3index |  |
| > | Min. 00.0000 | Length:531 | Min. $\quad 0.000$ |  |
| > | 1st Qu.:1.0000 | Class :character | 1st Qu.:3.000 |  |
| > | Median :1.0000 | Mode :character | Median :3.000 |  |
| > | Mean :0.8117 |  | Mean :3.073 |  |
| > | 3rd Qu.:1.0000 |  | 3rd Qu.:4.000 |  |
| > | Max. $: 1.0000$ |  | Max. 77.000 |  |

Once again we have a reduced dataset with 531 rows and 345 individuals.
Now let's build our matrices. We will use the same supplement table as before in ahistorical analysis, but we will redefine it for our new stageframe.

```
lathsupp2 <- supplemental(stage3 = c("Sd", "Sdl", "Sd", "Sdl"),
    stage2 = c("Sd", "Sd", "rep", "rep"), givenrate = c(0.345, 0.054, NA, NA),
    multiplier = c(NA, NA, 0.345, 0.054), type = c(1, 1, 3, 3),
    stageframe = lathframe_raw, historical = FALSE, stagebased = TRUE,
    agebased = TRUE)
lathmat2p_raw <- arlefko2(data = lathvert_raw_small, stageframe = lathframe_raw,
    supplement = lathsupp2, stages = c("stage3", "stage2", "stage1"),
    patch = "all", patchcol = "patchid", yearcol = "year2", agecol = "obsage",
    indivcol = "individ")
summary(lathmat2p_raw)
>
> This ahistorical lefkoMat object contains 12 matrices.
>
> Each matrix is square with }21\mathrm{ rows and columns, and a total of 441 elements.
> A total of 209 survival transitions were estimated, with 17.417 per matrix.
> A total of 12 fecundity transitions were estimated, with 1 per matrix.
> This lefkoMat object covers 1 population, 6 patches, and 2 time steps.
> The dataset contains a total of 345 unique individuals and 531 unique transitions.
>
> Survival probability sum check (each matrix represented by column in order):
> [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11] [,12]
> Min. 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000
> 1st Qu. 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000
> Median 0.000 0.000 0.000 0.399 0.000 0.399 0.000 0.399 0.000 0.399 0.000 0.000
> Mean 0.281 0.349 0.261 0.385 0.274 0.463 0.268 0.423 0.223 0.388 0.234 0.279
> 3rd Qu. 0.399 0.778 0.399 0.750 0.399 1.000 0.399 0.778 0.399 0.818 0.399 0.399
> Max. 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000
```

Here we have 12 matrices, each with 21 rows and columns reflecting seven stages and three ages.

There are a total of 441 elements per matrix, but only an average of 18.42 are estimated per matrix due to the sparsity of data relative to the size of the matrix. Let's take a look at the first matrix.


One of the more interesting problems that we can see here is the fact that since this matrix reflects a very small dataset, both in terms of numbers of observed transitions and in terms of years of observation, we cannot really estimate transitions staying within the final age. To yield such transitions, we would
typically need a large and long enough dataset that we could deliberately reduce the number of ages to a number less than the longest seen, perhaps reflecting stable demographic patterns beyond some age. Clearly, here is one area in which function-based MPMs have a clear advantage.

To attempt to increase the number of estimated elements, let's develop an MPM that does not discriminate patches.

```
lathmat2_raw <- arlefko2(data = lathvert_raw_small, stageframe = lathframe_raw,
    supplement = lathsupp2, stages = c("stage3", "stage2", "stage1"),
    yearcol = "year2", agecol = "obsage", indivcol = "individ")
summary(lathmat2_raw)
>
> This ahistorical lefkoMat object contains 2 matrices.
>
> Each matrix is square with 21 rows and columns, and a total of 441 elements.
> A total of 59 survival transitions were estimated, with 29.5 per matrix.
> A total of 6 fecundity transitions were estimated, with 3 per matrix.
> This lefkoMat object covers 1 population, 1 patch, and 2 time steps.
>
> The dataset contains a total of 345 unique individuals and 531 unique transitions.
>
> Survival probability sum check (each matrix represented by column in order):
> [,1] [,2]
> Min. 0.000 0.000
> 1st Qu. 0.000 0.000
> Median 0.000 0.399
> Mean 0.271 0.481
> 3rd Qu. 0.399 0.913
> Max. 1.000 1.000
```

Note the increase in estimated elements - now we have 32.5 elements per matrix, but only two matrices. Let's take a look at the first matrix here.

```
lathmat2_raw$A[[1]]
> [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8]
> [1,] 0.000 0.00000000 0.00000000 0.00000000 0.0000000 0 2.07000000 0.000
> [2,] 0.000 0.00000000 0.00000000 0.00000000 0.0000000 0 0.32400000 0.000
> [3,] 0.000 0.00000000 0.00000000 0.00000000 0.0000000 0 0.00000000 0.000
> [4,] 0.000 0.00000000 0.00000000 0.00000000 0.0000000 0 0.00000000 0.000
> [5,] 0.000 0.00000000 0.00000000 0.00000000 0.0000000 0 0.00000000 0.000
> [6,] 0.000 0.00000000 0.00000000 0.00000000 0.0000000 0 0.00000000 0.000
> [7,] 0.000 0.00000000 0.00000000 0.00000000 0.0000000 0 0.00000000 0.000
> [8,] 0.345 0.00000000 0.00000000 0.00000000 0.0000000 0 0.00000000 0.000
> [9,] 0.054 0.00000000 0.00000000 0.00000000 0.0000000 0 0.00000000 0.000
> [10,] 0.000 0.70085470 0.78787879 0.25490196 0.1111111 0 0.00000000 0.000
> [11,] 0.000 0.02564103 0.12121212 0.52941176 0.3333333 0 0.36363636 0.000
> [12,] 0.000 0.00000000 0.00000000 0.00000000 0.2222222 0 0.27272727 0.000
> [13,] 0.000 0.02564103 0.03030303 0.05882353 0.1111111 0 0.09090909 0.000
> [14,] 0.000 0.00000000 0.00000000 0.01960784 0.1666667 0 0.27272727 0.000
> [15,] 0.000 0.00000000 0.00000000 0.00000000 0.0000000 0 0.00000000 0.345
> [16,] 0.000 0.00000000 0.00000000 0.00000000 0.0000000 0 0.00000000 0.054
> [17,] 0.000 0.00000000 0.00000000 0.00000000 0.0000000 0 0.00000000 0.000
```

| > [18,] | 0.000 | 0.00000000 | 0.00000000 | 0.000000000 | 0.0000000 |  | 00.000 | 00000 | 0.000 |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| > [19,] | 0.000 | 0.00000000 | 0.00000000 | 0.000000000 | 0.0000000 |  | 00.000 | 00000 | 0.000 |
| > [20,] | 0.000 | 0.00000000 | 0.00000000 | 0.000000000 | 0.0000000 |  | 00.000 | 00000 | 0.000 |
| $>$ [21,] | 0.000 | 0.00000000 | 0.00000000 | 0.00000000 | 0.0000000 |  | 00.000 | 00000 | 0.000 |
| > | [,9] | [,10] [,11] | [,12] [,13] | [,14] [,15] | ] [,16] | [,17] | [,18] | [,19] | [,20] |
| $>\quad[1$, | 0 | 00 | 00 | 00.000 | 0 | 0 | 0 | 0 | 0 |
| $>\quad[2$, | 0 | 00 | 00 | 00.000 | 0 | 0 | 0 | 0 | 0 |
| $>$ [3,] | 0 | 00 | 00 | 00.000 | 0 | 0 | 0 | 0 | 0 |
| $>$ [4, ] | 0 | 00 | 00 | 00.000 | 0 | 0 | 0 | 0 | 0 |
| $>$ [5,] | 0 | 00 | 00 | 00.000 | 0 | 0 | 0 | 0 | 0 |
| $>\quad[6$, | 0 | 00 | 00 | 00.000 | 0 | 0 | 0 | 0 | 0 |
| $>\quad[7$, | 0 | 00 | 00 | 00.000 | 0 | 0 | 0 | 0 | 0 |
| $>$ [8,] | 0 | 00 | 00 | 00.000 | 0 | 0 | 0 | 0 | 0 |
| $>$ [9,] | 0 | 00 | 00 | 00.000 | 0 | 0 | 0 | 0 | 0 |
| $>$ [10,] | 0 | 00 | 00 | 00.000 | 0 | 0 | 0 | 0 | 0 |
| $>$ [11,] | 0 | 00 | 00 | 00.000 | 0 | 0 | 0 | 0 | 0 |
| $>$ [12,] | 0 | 00 | 00 | 00.000 | 0 | 0 | 0 | 0 | 0 |
| $>$ [13,] | 0 | 00 | 00 | 00.000 | 0 | 0 | 0 | 0 | 0 |
| $>$ [14,] | 0 | 00 | 00 | 00.000 | 0 | 0 | 0 | 0 | 0 |
| $>$ [15,] | 0 | 00 | 00 | 00.345 | 50 | 0 | 0 | 0 | 0 |
| $>$ [16,] | 0 | 00 | 00 | 00.054 | 40 | 0 | 0 | 0 | 0 |
| > [17,] | 0 | 00 | 00 | 00.000 | 0 | 0 | 0 | 0 | 0 |
| $>$ [18,] | 0 | 00 | 00 | 00.000 | 0 | 0 | 0 | 0 | 0 |
| > [19,] | 0 | 00 | 00 | 000.000 | 0 | 0 | 0 | 0 | 0 |
| $>$ [20,] | 0 | 00 | 00 | 000.000 | 0 | 0 | 0 | 0 | 0 |
| $>$ [21, ] | 0 | 00 | 00 | 000.000 | 0 | 0 | 0 | 0 | 0 |
| > | [,21] |  |  |  |  |  |  |  |  |
| $>$ [1,] | 0 |  |  |  |  |  |  |  |  |
| $>$ [2,] | 0 |  |  |  |  |  |  |  |  |
| $>$ [3,] | 0 |  |  |  |  |  |  |  |  |
| $>$ [4,] | 0 |  |  |  |  |  |  |  |  |
| $>$ [5,] | 0 |  |  |  |  |  |  |  |  |
| $>$ [6,] | 0 |  |  |  |  |  |  |  |  |
| $>$ [7,] | 0 |  |  |  |  |  |  |  |  |
| $>$ [8,] | 0 |  |  |  |  |  |  |  |  |
| $>$ [9,] | 0 |  |  |  |  |  |  |  |  |
| $>$ [10, ] | 0 |  |  |  |  |  |  |  |  |
| $>$ [11,] | 0 |  |  |  |  |  |  |  |  |
| $>$ [12,] | 0 |  |  |  |  |  |  |  |  |
| $>$ [13,] | 0 |  |  |  |  |  |  |  |  |
| $>$ [14,] | 0 |  |  |  |  |  |  |  |  |
| $>$ [15,] | 0 |  |  |  |  |  |  |  |  |
| $>[16$, | 0 |  |  |  |  |  |  |  |  |
| > [17, ] | 0 |  |  |  |  |  |  |  |  |
| $>$ [18,] | 0 |  |  |  |  |  |  |  |  |
| $>$ [19,] | 0 |  |  |  |  |  |  |  |  |
| $>$ [20,] | 0 |  |  |  |  |  |  |  |  |
| > [21,] | 0 |  |  |  |  |  |  |  |  |

### 6.4 Using supplements with age-by-stage MPMs

Supplement tables can be used to provide extra data for matrix construction according to ages as well as stages. Let's take an example. In the previous function-based and raw MPM construction scenarios, we assumed that dormant seeds stayed dormant seeds at a rate of 0.345 , an germinated at a rate of 0.054. However, suppose that we had data suggesting that two- and three-year old seed survived and germinated at only half those rates. In those cases, we would include an age vector holding the ages in time $t$ associated with the transitions to modify, as below.

```
lathsupp2_age <- supplemental(stage3 =
    c("Sd", "Sdl", "Sd", "Sdl", "Sd", "Sdl", "Sd", "Sdl"),
    stage2 = c("Sd", "Sd", "Sd", "Sd", "Sd", "Sd", "rep", "rep"),
    age2 = c(1, 1, 2, 2, 3, 3, NA, NA),
    givenrate = c(0.345, 0.054, 0.173, 0.027, 0.173, 0.027, 0.173, 0.027, NA, NA),
    multiplier = c(NA, NA, NA, NA, NA, NA, 0.345, 0.054),
    type = c(1, 1, 1, 1, 1, 1, 3, 3),
    stageframe = lathframeln, historical = FALSE, stagebased = TRUE,
    agebased = TRUE)
lathsupp2_age
> Warning in format.data.frame(if (omit) x[seq_len(n0), , drop = FALSE] else x, :
> corrupt data frame: columns will be truncated or padded with NAs
> stage3 stage2 stage1 age2 eststage3 eststage2 eststage1 estage2 givenrate
```




```
\(>3\) Sd Sd <NA> 2 <NA> <NA> 3 <NA> NA 0.173
\(>4\) Sdl Sd <NA> 2 <NA> <NA> 4 <NA> NA 0.027
```



```
\begin{tabular}{lrrrrlllll}
\(>6\) & Sdl & Sd & <NA> & 3 & <NA> & <NA> & <NA> & NA & 0.027 \\
\(>7\) & Sd & rep & <NA> & NA & <NA> & <NA> & <NA> & NA & 0.173
\end{tabular}
\begin{tabular}{llllllll}
\(>\) & Sdl rep <NA> NA <NA> & <NA> & <NA> & NA & 0.027
\end{tabular}
multiplier convtype convtype_t12
1 NA 1
2 NA 1
N NA 1
N NA 1
5 NA 1
6 NA 1
>7 0.345 3
> 8 0.054 3
```

Let's use this new supplement with the raw MPM and take a look at the summary.

```
lathmat2p_raw_age <- arlefko2(data = lathvert_raw_small, stageframe = lathframe_raw,
    supplement = lathsupp2_age, stages = c("stage3", "stage2", "stage1"),
    patch = "all", patchcol = "patchid", yearcol = "year2", agecol = "obsage",
    indivcol = "individ")
summary(lathmat2p_raw_age)
>
> This ahistorical lefkoMat object contains 12 matrices.
>
```

```
> Each matrix is square with 21 rows and columns, and a total of 441 elements.
> A total of 209 survival transitions were estimated, with 17.417 per matrix.
> A total of }12\mathrm{ fecundity transitions were estimated, with 1 per matrix.
> This lefkoMat object covers 1 population, }6\mathrm{ patches, and 2 time steps.
>
> The dataset contains a total of 345 unique individuals and 531 unique transitions.
>
> Survival probability sum check (each matrix represented by column in order):
> [,1] [, 2] [,3] [,4] [,5] [,6] [,7] [, 8] [,9] [,10] [, 11] [,12]
>Min. 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000
> 1st Qu. 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000
>Median 0.000 0.000 0.000 0.200 0.000 0.200 0.000 0.200 0.000 0.200 0.000 0.000
l Mean 0.262 0.330 0.242 0.366 0.256 0.444 0.249 0.404 0.204 0.369 0.216 0.260
> 3rd Qu. 0.399 0.778 0.399 0.750 0.399 1.000 0.399 0.778 0.200 0.818 0.200 0.399
>Max. 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000
```

The numbers of estiamted elements are the same as before, because we are modifying the same elements as before. However, let's take a look at one of the associated A matrices to get an idea of what happened.

| > |  | [,1] | [,2] | [,3] | [,4] | [,5] | ] [,6] | [,7] | [,8] | [,9] | [,10] | [,11] |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| > | [1,] | 0.000 | 0.0000000 | 0.0000000 | 0.0000 | 0.000 | 0 | 1.035 | 0.000 | 0 | 0 | 0 |
| > | [2,] | 0.000 | 0.0000000 | 0.0000000 | 0.0000 | 0.000 | 0 | 0.162 | 2.000 | 0 | 0 | 0 |
| > | [3,] | 0.000 | 0.0000000 | 0.0000000 | 0.0000 | 0.000 | 0 | 0.000 | 0.000 | 0 | 0 | 0 |
| > | [4, ] | 0.000 | 0.0000000 | 0.0000000 | 0.0000 | 0.000 | 0 | 0.000 | 0.000 | 0 | 0 | 0 |
| > | [5,] | 0.000 | 0.0000000 | 0.0000000 | 0.0000 | 0.000 | 0 | 0.000 | 0.000 | 0 | 0 | 0 |
| > | [6,] | 0.000 | 0.0000000 | 0.0000000 | 0.0000 | 0.000 | 0 | 0.000 | 0.000 | 0 | 0 | 0 |
| > | [7, ] | 0.000 | 0.0000000 | 0.0000000 | 0.0000 | 0.000 | 0 | 0.000 | 0.000 | 0 | 0 | 0 |
| > | [8,] | 0.345 | 0.0000000 | 0.0000000 | 0.0000 | 0.000 | 0 | 0.000 | 0.000 | 0 | 0 | 0 |
| > | [9,] | 0.054 | 0.0000000 | 0.0000000 | 0.0000 | 0.000 | 0 | 0.000 | 0.000 | 0 | 0 | 0 |
| > | [10,] | 0.000 | 0.9047619 | 0.7142857 | 0.1875 | 0.250 | 0 | 0.000 | 0.000 | 0 | 0 | 0 |
| > | [11,] | 0.000 | 0.0000000 | 0.1428571 | 0.6875 | 0.500 | 0 | 1.000 | 0.000 | 0 | 0 | 0 |
| > | [12,] | 0.000 | 0.0000000 | 0.0000000 | 0.0000 | 0.000 | 0 | 0.000 | 0.000 | 0 | 0 | 0 |
| > | [13,] | 0.000 | 0.0000000 | 0.0000000 | 0.0625 | 0.125 | 50 | 0.000 | 0.000 | 0 | 0 | 0 |
| > | [14,] | 0.000 | 0.0000000 | 0.0000000 | 0.0000 | 0.125 | 50 | 0.000 | 0.000 | 0 | 0 | 0 |
| > | [15,] | 0.000 | 0.0000000 | 0.0000000 | 0.0000 | 0.000 | 0 | 0.000 | 0.173 | 0 | 0 | 0 |
| > | [16,] | 0.000 | 0.0000000 | 0.0000000 | 0.0000 | 0.000 | 0 | 0.000 | 0.027 | 0 | 0 | 0 |
| > | [17,] | 0.000 | 0.0000000 | 0.0000000 | 0.0000 | 0.000 | 0 | 0.000 | 0.000 | 0 | 0 | 0 |
| > | [18,] | 0.000 | 0.0000000 | 0.0000000 | 0.0000 | 0.000 | 0 | 0.000 | 0.000 | 0 | 0 | 0 |
| > | [19,] | 0.000 | 0.0000000 | 0.0000000 | 0.0000 | 0.000 | 0 | 0.000 | 0.000 | 0 | 0 | 0 |
| > | [20,] | 0.000 | 0.0000000 | 0.0000000 | 0.0000 | 0.000 | 0 | 0.000 | 0.000 | 0 | 0 | 0 |
| > | [21,] | 0.000 | 0.0000000 | 0.0000000 | 0.0000 | 0.000 | 0 | 0.000 | 0.000 | 0 | 0 | 0 |
| > |  | [,12] | [,13] [,14] | ] [,15] [ | [,16] [,17 | 17] [, | [18] [ | [,19] [ | [,20] [, | [,21] |  |  |
| $>$ | [1, ] | 0 | 0 | 00.000 | 0 | 0 | 0 | 0 | 0 | 0 |  |  |
|  | [2,] | 0 | 0 | 00.000 | 0 | 0 | 0 | 0 | 0 | 0 |  |  |
|  | [3, ] | 0 | 0 | 00.000 | 0 | 0 | 0 | 0 | 0 | 0 |  |  |
|  | [4, ] | 0 | 0 | 00.000 | 0 | 0 | 0 | 0 | 0 | 0 |  |  |
|  | [5,] | 0 | 0 | 00.000 | 0 | 0 | 0 | 0 | 0 | 0 |  |  |
|  | [6,] | 0 | 0 | 00.000 | 0 | 0 | 0 | 0 | 0 | 0 |  |  |
|  | [7,] | 0 | 0 | 00.000 | 0 | 0 | 0 | 0 | 0 | 0 |  |  |
|  | [8,] | 0 | 0 | 00.000 | 0 | 0 | 0 | 0 | 0 | 0 |  |  |
|  | [9,] | 0 | 0 | 00.000 | 0 | 0 | 0 | 0 | 0 | 0 |  |  |


| $>[10]$, | 0 | 0 | 0 | 0.000 | 0 | 0 | 0 | 0 | 0 | 0 |
| :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- |
| $>[11]$, | 0 | 0 | 0 | 0.000 | 0 | 0 | 0 | 0 | 0 | 0 |
| $>[12]$, | 0 | 0 | 0 | 0.000 | 0 | 0 | 0 | 0 | 0 | 0 |
| $>[13]$, | 0 | 0 | 0 | 0.000 | 0 | 0 | 0 | 0 | 0 | 0 |
| $>[14]$, | 0 | 0 | 0 | 0.000 | 0 | 0 | 0 | 0 | 0 | 0 |
| $>[15]$, | 0 | 0 | 0 | 0.173 | 0 | 0 | 0 | 0 | 0 | 0 |
| $>[16]$, | 0 | 0 | 0 | 0.027 | 0 | 0 | 0 | 0 | 0 | 0 |
| $>[17]$, | 0 | 0 | 0 | 0.000 | 0 | 0 | 0 | 0 | 0 | 0 |
| $>[18]$, | 0 | 0 | 0 | 0.000 | 0 | 0 | 0 | 0 | 0 | 0 |
| $>[19]$, | 0 | 0 | 0 | 0.000 | 0 | 0 | 0 | 0 | 0 | 0 |
| $>[20]$, | 0 | 0 | 0 | 0.000 | 0 | 0 | 0 | 0 | 0 | 0 |
| $>[21]$, | 0 | 0 | 0 | 0.000 | 0 | 0 | 0 | 0 | 0 | 0 |

We can see the modified elements in columns 1,8 , and 15 , and these are repeated in the other A and U matrices. To single out the differences, let's see a matrix difference between the MPM produced with the original supplement table, and the new MPM.


| $>$ | $[12]$, | 0 | 0.000 | 0 | 0 | 0 | 0 | 0 |
| :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- |
| $>[13]$, | 0 | 0.000 | 0 | 0 | 0 | 0 | 0 | 0 |
| $>[14]$, | 0 | 0.000 | 0 | 0 | 0 | 0 | 0 | 0 |
| $>[15]$, | 0 | 0.172 | 0 | 0 | 0 | 0 | 0 | 0 |
| $>[16]$, | 0 | 0.027 | 0 | 0 | 0 | 0 | 0 | 0 |
| $>[17]$, | 0 | 0.000 | 0 | 0 | 0 | 0 | 0 | 0 |
| $>[18]$, | 0 | 0.000 | 0 | 0 | 0 | 0 | 0 | 0 |
| $>[19]$, | 0 | 0.000 | 0 | 0 | 0 | 0 | 0 | 0 |
| $>[20]$, | 0 | 0.000 | 0 | 0 | 0 | 0 | 0 | 0 |
| $>[21]$, | 0 | 0.000 | 0 | 0 | 0 | 0 | 0 | 0 |

And we see differences only in columns 8 and 15, as expected.
Let's now move on to Leslie MPMs.

### 6.5 Age-classified (Leslie) MPMs

Package lefko3 can also be used to estimate Leslie MPMs, which are purely age-based and so include no individual history. Here, we will illustrate how to create both raw and function-based Leslie MPMs using the rleslie() and fleslie() functions, respectively. We will ignore the dormant seed stage, as including dormant seeds would require using an age-by-stage approach. Our final matrices will take the following form.

$$
\left[\begin{array}{rrrr}
F_{0,0} & F_{0,1} & F_{0,2} & F_{0,3}  \tag{6.4}\\
S_{1,0} & 0 & 0 & 0 \\
0 & S_{2,1} & 0 & 0 \\
0 & 0 & S_{3,2} & S_{3,3}
\end{array}\right]
$$

Normally, we would need to develop a stageframe. However, in the purely age-based case, a stageframe is unnecessary and instead all we need is the standardized hfv dataset. So, let's start by creating that. Most of the settings will be as before. However, we need to include both NRasRep $=$ TRUE and NOasObs = TRUE to make sure that stage classification ignores whether the individual was actually reproductive and actually observed at each time (the latter has to do with the fact that individuals can be alive with a size of zero if they are vegetatively dormant). We will also subset our data to only those individuals whose age we are reasonably sure of, by eliminating those individuals first observed in the first monitoring session.

```
lathvert_base <- verticalize3(lathyrus, noyears = 4, firstyear = 1988,
    patchidcol = "SUBPLOT", individcol = "indiv_id", blocksize = 9,
    sizeacol = "Volume88", repstracol = "FCODE88", fecacol = "Intactseed88",
    deadacol = "Dead1988", censorcol = "Missing1988", censorkeep = NA,
    censorRepeat= TRUE, censor = TRUE, NAas0 = TRUE, NRasRep = TRUE,
    NOasObs = TRUE)
lathvert_age <- subset(lathvert_base, firstseen > 1988)
summary_hfv(lathvert_age)
>
> This hfv dataset contains 531 rows, 51 variables, 1 population,
> 6 patches, 345 individuals, and 2 time steps.
> rowid popid patchid individ
> Min. : 30.0 Length:531 Min. :1.000 Length:531
> 1st Qu.: 301.0 Class :character 1st Qu.:2.000 Class :character
```




So far so good! Now let's create the vital rate models. This will go much more quickly than last time, because we no longer care about size and reproductive status as factors determining vital rates. So, we will set suite $=$ "cons" to prevent these factors from being tested, and set age = "obsage" to incorporate our age at time $t$ variable into all models. We will also use global.only $=$ TRUE to prevent the age term from being dropped (normally we would not set this, but in our case the dataset is so small that age will drop out in the best-fit models, so we will force the models to include age in this case).

```
lathmodels2_age <- modelsearch(lathvert_age, historical = FALSE,
    approach = "mixed", suite = "cons", bestfit = "AICc&k", age = "obsage",
    vitalrates = c("surv", "fec"), fecdist = "negbin", indiv = "individ",
    year = "year2", year.as.random = TRUE, patch.as.random = TRUE,
    show.model.tables = TRUE, fec.zero = TRUE, global.only = TRUE,
    test.age = TRUE, quiet = "partial")
```

Let's see a summary of the models.

```
summary(lathmodels2_age)
> This LefkoMod object includes 2 linear models.
> Best-fit model criterion used: global model only
>
>
>
> Survival model:
> Generalized linear mixed model fit by maximum likelihood (Laplace
        Approximation) [glmerMod]
Family: binomial ( logit )
Formula: alive3 ~ obsage + (1 | year2) + (1 | individ)
            Data: subdata
                AIC BIC logLik deviance df.resid
    588.5991 605.6982 -290.2996 580.5991 527
Random effects:
Groups Name Std.Dev.
    individ (Intercept) 0.0000
    year2 (Intercept) 0.2988
Number of obs: 531, groups: individ, 345; year2, 2
Fixed Effects:
> (Intercept) obsage
> 1.08983 0.09083
optimizer (Nelder_Mead) convergence code: O (OK) ; 0 optimizer warnings; 1 lme4 warnings
>
>
>
> Observation model:
[1] 1
>
>
>
> Size model:
[1] 1
>
>
>
> Secondary size model:
[1] 1
>
>
>
> Tertiary size model:
[1] 1
>
>
>
> Reproductive status model:
[1] 1
>
>
```

```
>
Fecundity model:
Formula: feca2 ~ obsage + (1 | year2) + (1 | individ)
Zero inflation: ~.
Data: subdata
    AIC BIC logLik df.resid
NA NA NA 10
Random-effects (co)variances:
Conditional model:
Groups Name Std.Dev.
year2 (Intercept) 0.02271
    individ (Intercept) 1.19287
>
Zero-inflation model:
Groups Name Std.Dev.
year2 (Intercept) 0.09092
    individ (Intercept) 0.17023
Number of obs: 19 / Conditional model: year2, 2; individ, 16 / Zero-inflation model: year2, 2; indi
Dispersion parameter for nbinom2 family (): 88.6
Fixed Effects:
>
Conditional model:
(Intercept) obsage
            1.3220 0.3938
Zero-inflation model:
(Intercept) obsage
            -3.842 2.635
>
>
> Juvenile survival model:
[1] 1
>
>
>
> Juvenile observation model:
[1] 1
>
>
>
> Juvenile size model:
[1] 1
>
>
>
> Juvenile secondary size model:
> [1] 1
```

```
>
>
>
Juvenile tertiary size model:
[1] 1
>
>
>
Juvenile reproduction model:
[1] 1
>
>
>
> Juvenile maturity model:
[1] 1
Number of models in survival table: 1
Number of models in observation table: 1
Number of models in size table: 1
Number of models in secondary size table: 1
Number of models in tertiary size table: 1
Number of models in reproduction status table: 1
Number of models in fecundity table: 1
Number of models in juvenile survival table: 1
Number of models in juvenile observation table: 1
Number of models in juvenile size table: 1
Number of models in juvenile secondary size table: 1
Number of models in juvenile tertiary size table: 1
Number of models in juvenile reproduction table: 1
Number of models in juvenile maturity table: 1
```

```
>
> General model parameter names (column 1), and
> specific names used in these models (column 2):
> parameter_names mainparams
> 1 time t year2
> 2 individual individ
3 patch patch
4 alive in time t+1 surv3
> observed in time t+1 obs3
>6 sizea in time t+1 size3
>7 sizeb in time t+1 sizeb3
> 8 sizec in time t+1 sizec3
> 9 reproductive status in time t+1 repst3
> 10 fecundity in time t+1 fec3
> 11 fecundity in time t fec2
> 12 sizea in time t size2
> 13 sizea in time t-1 size1
14 sizeb in time t sizeb2
15 sizeb in time t-1 sizeb1
sizec in time t sizec2
            sizec in time t-1 sizec1
        reproductive status in time t repst2
        reproductive status in time t-1 repst1
            maturity status in time t+1 matst3
            maturity status in time t matst2
                    age in time t age
            density in time t density
        individual covariate a in time t indcova2
        individual covariate a in time t-1 indcova1
        individual covariate b in time t indcovb2
    individual covariate b in time t-1 indcovb1
        individual covariate c in time t indcovc2
        individual covariate c in time t-1 indcovc1
            stage group in time t group2
            stage group in time t-1 group1
>
>
>
>
>
> Quality control:
>
> Survival model estimated with 345 individuals and 531 individual transitions.
> Survival model accuracy is 0.761.
> Observation status model not estimated.
> Primary size model not estimated.
> Secondary size model not estimated.
> Tertiary size model not estimated.
> Reproductive status model not estimated.
> Fecundity model estimated with 16 individuals and 19 individual transitions.
> Fecundity model R-squared is 0.834.
```

```
> Juvenile survival model not estimated.
> Juvenile observation status model not estimated.
> Juvenile primary size model not estimated.
> Juvenile secondary size model not estimated.
> Juvenile tertiary size model not estimated.
> Juvenile reproductive status model not estimated.
> Juvenile maturity status model not estimated.
```

The survival and fecundity models both include observed age, as well as our random terms.
Let's now build a supplement table. This requires some different thinking than in the previous case. Here, we will ignore dormant seeds and focus only on individuals from the point of germination. Thus, we no longer care about the survival rate of dormant seeds. Instead, we wish to incorporate a fecundity multiplier accounting for the germination rate. Let's try that, as below. Note that we need to include the historical $=$ FALSE, stagebased $=$ FALSE, agebased $=$ TRUE set of options for this to. work properly.

```
lathsupp2_ageonly <- supplemental(age2 = c(2, 3), multiplier = c(0.345, 0.345),
    type = c(3, 3), historical = FALSE, stagebased = FALSE, agebased = TRUE)
lathsupp2_ageonly
> stage3 stage2 stage1 age2 eststage3 eststage2 eststage1 estage2 givenrate
>1 <NA> <NA> <NA> 2 <NA> <NA> <NA> <NA> NA NA
>2 <NA> <NA> <NA> 3 <NA> <NA> <NA> NA NA
> multiplier convtype convtype_t12
> 1 0.345 3
> 2 0.345 3
```

Now we will create the raw matrices. Note that this MPM does not require any set of vital rate models - only the dataset and, if appropriate, a supplement table.

```
lathmat2ageonly_raw <- rleslie(data = lathvert_age, year = "all",
    supplement = lathsupp2_ageonly, yearcol = "year2", indivcol = "individ")
summary(lathmat2ageonly_raw)
>
> This ahistorical lefkoMat object contains 2 matrices.
>
> Each matrix is square with 3 rows and columns, and a total of 9 elements.
> A total of 3 survival transitions were estimated, with 1.5 per matrix.
> A total of 3 fecundity transitions were estimated, with 1.5 per matrix.
> This lefkoMat object covers 1 population, 1 patch, and 2 time steps.
>
> The dataset contains a total of 345 unique individuals and 531 unique transitions.
>
> Survival probability sum check (each matrix represented by column in order):
> [,1] [,2]
> Min. 0.000 0.000
> 1st Qu. 0.000 0.343
> Median 0.000 0.687
> Mean 0.274 0.473
> 3rd Qu. 0.411 0.709
> Max. 0.822 0.731
```

We have just created the raw Leslie MPM. We have two matrices, because we needed to eliminate the first year of data to estimate age properly, leaving only three observation periods. The quality control looks reasonable in terms of the survival-transition probabilities falling within the range of 0.0 to 1.0. Let's also take a look at one of the matrices, to get a handle on its structure.

| lathmat2ageonly_raw\$A[[2]] |  |  |  |
| :---: | :---: | :---: | :---: |
| > | [,1] | [,2] | [,3] |
| > [1,] | 0.02608696 | 0.01483871 | 0 |
| > [2,] | 0.68695652 | 0.0000000 | 0 |
| > [3,] | 0.00000000 | 0.73118280 | 0 |

Our matrix certainly looks like a valid Leslie matrix, with survival transitions in the subdiagonal and fecundity at the top. The last column is entirely composed of zeros, and that is likely due to a combination of a small dataset and a short dataset in which we cannot find individuals whom we know are older than three years old. If we view the first matrix, we will see a further column full of zeroes, as well.

Let's now build the function-based versions and see what happens.

```
lathmat2ageonly_func <- fleslie(year = "all", data = lathvert_age,
    supplement = lathsupp2_ageonly, modelsuite = lathmodels2_age)
summary(lathmat2ageonly_func)
>
> This ahistorical lefkoMat object contains 2 matrices.
>
> Each matrix is square with 3 rows and columns, and a total of 9 elements.
> A total of 6 survival transitions were estimated, with 3 per matrix.
> A total of 6 fecundity transitions were estimated, with 3 per matrix.
> This lefkoMat object covers 1 population, 1 patch, and 2 time steps.
>
> The dataset contains a total of 345 unique individuals and 531 unique transitions.
>
> Vital rate modeling quality control:
>
> Survival estimated with 345 individuals and 531 individual transitions.
> Observation probability not estimated.
> Primary size transition not estimated.
> Secondary size transition not estimated.
> Tertiary size transition not estimated.
> Reproduction probability not estimated.
> Fecundity estimated with 16 individuals and 19 individual transitions.
> Juvenile survival not estimated.
> Juvenile observation probability not estimated.
> Juvenile primary size transition not estimated.
> Juvenile secondary size transition not estimated.
> Juvenile tertiary size transition not estimated.
> Juvenile reproduction probability not estimated.
> Juvenile maturity transition probability not estimated.
>
> Survival probability sum check (each matrix represented by column in order):
> [,1] [,2]
> Min. 0.809 0.713
```

```
> 1st Qu. 0.816 0.722
> Median 0.823 0.731
> Mean 0.822 0.731
> 3rd Qu. 0.829 0.740
> Max. 0.835 0.749
lathmat2ageonly_func$A[[2]]
> [,1] [,2] [,3]
> [1,] 0.2304600 0.2783520 0.3391766
> [2,] 0.7130813 0.0000000 0.0000000
> [3,] 0.0000000 0.7312994 0.7487683
```

Notice that we have two matrices here (reflecting the two estimable years), and we have estimates for survival transition probabilities staying within the last age. Clearly, function-based approaches have an advantage in this small, short dataset.

### 6.6 Age-hybrid MPMs

In some cases, we may wish to create and MPM that mostly conforms to Leslie format, but perhaps has a single stage thrown in. Such a situation may occur, for example, if the user wishes to build an MPM that has a dormant propagule stage, such as a plant with a dormant seed. In that circumstance, the life of the plant from germination onward might be purely age-based, but prior to that the plant might spend an unknown number of years as dormant seed with its own particular survival probability. Here is a sample matrix corresponding to that situation:

$$
\left[\begin{array}{rrrr}
S_{1,1} & 0 & 0 & F_{1,4}  \tag{6.5}\\
S_{2,1} & 0 & 0 & F_{2,4} \\
0 & S_{3,2} & 0 & 0 \\
0 & 0 & S_{4,3} & S_{4,4}
\end{array}\right]
$$

Note that this is the same matrix as equation (6.2).
Fortunately, this is an easy situation to do deal with. First, we create a Leslie MPM without the added stage. In this case, that means repeating either the function-based or raw Leslie MPM that we already created in section 6.5. For example, here is our function-based MPM:

```
lathmat2ageonly_func
> $A
> $A[[1]]
> [,1] [,2] [,3]
> [1,] 0.2318637 0.2787757 0.3392217
> [2,] 0.8089559 0.0000000 0.0000000
> [3,] 0.0000000 0.8226009 0.8354695
>
> $A[[2]]
> [,1] [,2] [,3]
> [1,] 0.2304600 0.2783520 0.3391766
> [2,] 0.7130813 0.0000000 0.0000000
> [3,] 0.0000000 0.7312994 0.7487683
>
>
> $U
> $U[[1]]
> [,1] [,2] [,3]
```

```
[1,] 0.0000000 0.0000000 0.0000000
[2,] 0.8089559 0.0000000 0.0000000
[3,] 0.0000000 0.8226009 0.8354695
>
$U[[2]]
[,1] [,2] [,3]
[1,] 0.0000000 0.0000000 0.0000000
[2,] 0.7130813 0.0000000 0.0000000
[3,] 0.0000000 0.7312994 0.7487683
>
>
$F
$F[[1]]
[,1] [,2] [,3]
[1,] 0.2318637 0.2787757 0.3392217
[2,] 0.0000000 0.0000000 0.0000000
[3,] 0.0000000 0.0000000 0.0000000
$F[[2]]
            [,1] [,2] [,3]
[1,] 0.23046 0.278352 0.3391766
[2,] 0.00000 0.000000 0.0000000
[3,] 0.00000 0.000000 0.0000000
>
>
> $ahstages
> stage_id stage original_size original_size_b original_size_c min_age max_age
> 1 1 Age1 NA
> 2 2 Age2 NA
3 Age3 NA NA NA NA
> repstatus obsstatus propstatus immstatus matstatus entrystage indataset
\begin{tabular}{llllllll}
1 & 1 & 1 & 0 & 0 & 1 & 1 & 1
\end{tabular}
2 1 1 1 1 % 0 % 0 % 0
    binhalfwidth_raw sizebin_min sizebin_max sizebin_center sizebin_width
1 NA NA NA NA NA
>2 NA NA NA NA N
3 NA NA NA NA NA
> binhalfwidthb_raw sizebinb_min sizebinb_max sizebinb_center sizebinb_width
1 NA NA NA NA
2 NA NA NA NA NA
3 NA NA NA NA NA
> binhalfwidthc_raw sizebinc_min sizebinc_max sizebinc_center sizebinc_width
\begin{tabular}{llllll}
\(>1\) & NA & NA & NA & NA & NA \\
\(>2\) & NA & NA & NA & NA & NA \\
\(>3\) & NA & NA & NA & NA
\end{tabular}
> NA NA NA NA
> group comments alive almostborn
> 1 0 No description 1 0
> 2 0 No description 1 0
> 3 0 No description 1 0
>
```

```
> $agestages
> X1
> 1 NA
>
> $hstages
> X1
> 1 NA
>
> $labels
> pop patch year2
> 1 1 1 1 1989
> 2 1 1 1990
>
> $matrixqc
> [1] 6 6 2
>
> $modelqc
> vital_rate individuals transitions distribution accuracy
1 survival 345 531 binomial 0.7608286
2 observation 0 NA
3 size 0 0 gaussian NA
4 sizeb 0 0 0 NA
5 sizec 0-NA
> 6 reproduction 
> 8 juvenile_survival 0 0 0 binomial N
>9 juvnile_observation 0 0 binomial NA
> 10 juvenile_size 0 0 gaussian NA
> 11 juvenile_sizeb 0 0 NA
> 12 %uvenile_sizec 
> 14 juvenile_maturity 0 0 binomial NA
>
> $dataqc
> [1] 345 531
>
> attr(,"class")
> [1] "lefkoMat"
```

Now let's copy this over to a new MPM with a more appropriate name for what we wish to do.

```
lathmat2_agehybrid <- lathmat2ageonly_func
```

One thing to note about this MPM is that, although we did not build a stageframe for this MPM, there is nonetheless a stageframe included as this MPM's ahstages object. Let's take a look at that.

```
lathmat2_agehybrid$ahstages
> stage_id stage original_size original_size_b original_size_c min_age max_age
> 1 1 Age1 NA NA NA N
>2 2 Age2 NA NA NA N
> 3 Age3 NA NA NA NA
> repstatus obsstatus propstatus immstatus matstatus entrystage indataset
```

302CHAPTER 6. MATRIX MODELS III: AGE (LESLIE), HYBRID AGE, AND AGE-BY-STAGE MPMS


Finally, let's use the add_stage() function to add a new stage before the first age, which is shown as Age1 in the stageframe above.

```
lathmat2_agehybrid <- add_stage(lathmat2_agehybrid, add_before = 1,
    stage_name = "DormSeed")
lathmat2_agehybrid
> $A
> $A[[1]]
> [,1] [,2] [,3] [,4]
> [1,] 0 0.0000000 0.0000000 0.0000000
> [2,] 0 0.2318637 0.2787757 0.3392217
> [3,] 0 0.8089559 0.0000000 0.0000000
> [4,] 0 0.0000000 0.8226009 0.8354695
>
> $A[[2]]
> [,1] [,2] [,3] [,4]
> [1,] 0 0.0000000 0.0000000 0.0000000
> [2,] 0 0.2304600 0.2783520 0.3391766
> [3,] 0 0.7130813 0.0000000 0.0000000
> [4,] 0 0.0000000 0.7312994 0.7487683
>
>
> $U
> $U[[1]]
> [,1] [,2] [,3] [,4]
> [1,] 0 0.0000000 0.0000000 0.0000000
> [2,] 0 0.0000000 0.0000000 0.0000000
> [3,] 0 0.8089559 0.0000000 0.0000000
> [4,] 0 0.0000000 0.8226009 0.8354695
>
> $U[[2]]
```

```
> [,1] [,2] [,3] [,4]
[1,] 0 0.0000000 0.0000000 0.0000000
[2,] 0 0.0000000 0.0000000 0.0000000
[3,] 0 0.7130813 0.0000000 0.0000000
[4,] 0 0.0000000 0.7312994 0.7487683
>
$F
$F[[1]]
[,1] [,2] [,3] [,4]
[1,] 0 0.0000000 0.0000000 0.0000000
```



```
[3,] 0 0.0000000 0.0000000 0.0000000
[4,] 0 0.0000000 0.0000000 0.0000000
>
> $F[[2]]
> [,1] [,2] [,3] [,4]
[1,] 0 0.00000 0.000000 0.0000000
[2,] 0 0.23046 0.278352 0.3391766
[3,] 0 0.00000 0.000000 0.0000000
> [4,] 0 0.00000 0.000000 0.0000000
>
>
> $ahstages
l
max_age repstatus obsstatus propstatus immstatus matstatus entrystage
```



```
> 2 
```



```
> indataset binhalfwidth_raw sizebin_min sizebin_max sizebin_center
> 1 0 1 1 < 0.5 l
>2 1 NA NA NA
> 3 1 NA NA NA NA
4 1 NA NA NA NA
> sizebin_width binhalfwidthb_raw sizebinb_min sizebinb_max sizebinb_center
\begin{tabular}{rrrrrr}
\(>\) & 1 & 0.5 & 1 & 0.5 & 1.5 \\
\(>\) & 2 & NA & NA & NA & NA \\
\(>3\) & NA & NA & NA & NA & NA \\
\(>4\) & NA & NA & NA & NA & NA \\
\hline
\end{tabular}
4NA NA NA NA NA
sizebinb_width binhalfwidthc_raw sizebinc_min sizebinc_max sizebinc_center
\begin{tabular}{llllll}
\(>1\) & 0.5 & 1 & 0.5 & 1.5 & 1
\end{tabular}
> 2 NA NA NA NA
> 3 NA NA NA NA NA
> 4 NA NA NA NA NA
> sizebinc_width group comments alive almostborn
> 1 0.5 0 new stage 1 0
```

```
\begin{tabular}{lllll}
\(>2\) & NA & 0 No description & 1 & 0 \\
\(>3\) & NA & 0 No description & 1 & 0 \\
\(>4\) & NA & 0 No description & 1 & 0
\end{tabular}
>
$agestages
        X1
    1 NA
>
$hstages
            X1
1 NA
$labels
pop patch year2
1 1 1 1989
2 1 1 1990
$matrixqc
    [1] 6 6 2
$modelqc
                    vital_rate individuals transitions distribution accuracy
1 survival 
2 observation 0 0 binomial NA
> 3 size 0 0 gaussian NA
>4 sizeb 0
> 5 sizec 0 0 0
>6 reproduction 0 0 binomial NA
7 fecundity 16 19 negbin 0.8343949
8 juvenile_survival 0 0 binomial N
9 juvnile_observation 0 0 bA
lrllll
llllll
> 13 juvenile_reproduction 0 0 binomial NA
> 14 juvenile_maturity 0 0 binomial NA
> $dataqc
> [1] 345 531
>
> attr(,"class")
> [1] "lefkoMat"
```

We now have a new MPM, but although the new stage that we have added is represented in the MPM, and the dimensions of the matrices have changed to 4 -by- 4 , we still need to edit the matrix to reflect the demography of this new stage. Here, we do that, adding new survival transition probabilities and splitting the fecundity values between the dormant stage and the first germinated age using the edit_1M() function.

```
lathmat2_agehybrid$ahstages$propstatus[1] <- 1 # Dormant seeds are propagules
lathmat2_agehybrid <- edit_lM(lathmat2_agehybrid,
```

```
    stage3 = c("DormSeed", "Age1", "Age1", "DormSeed"),
stage2 = c("DormSeed", "DormSeed", "Age3", "Age3"),
eststage3 = c(NA, NA, NA, "Age1"),
eststage2 = c(NA, NA, NA, "Age3"),
givenrate = c(0.5, 0.2, NA, NA),
multiplier = c(NA, NA, 0.5, 1),
type = c(1, 1, 3, 2))
lathmat2_agehybrid
$A
$A[[1]]
    [,1] [,2] [,3] [,4]
> [1,] 0.5 0.0000000 0.0000000 0.1696108
> [2,] 0.2 0.2318637 0.2787757 0.1696108
> [3,] 0.0 0.8089559 0.0000000 0.0000000
> [4,] 0.0 0.0000000 0.8226009 0.8354695
>
> $A[[2]]
    [,1] [,2] [,3] [,4]
> [1,] 0.5 0.0000000 0.0000000 0.1695883
> [2,] 0.2 0.2304600 0.2783520 0.1695883
> [3,] 0.0 0.7130813 0.0000000 0.0000000
> [4,] 0.0 0.0000000 0.7312994 0.7487683
>
>
> $U
$U[[1]]
    [,1] [,2] [,3] [,4]
    [1,] 0.5 0.0000000 0.0000000 0.0000000
> [2,] 0.2 0.0000000 0.0000000 0.0000000
> [3,] 0.0 0.8089559 0.0000000 0.0000000
> [4,] 0.0 0.0000000 0.8226009 0.8354695
>
> $U[[2]]
> [,1] [,2] [,3] [,4]
> [1,] 0.5 0.0000000 0.0000000 0.0000000
> [2,] 0.2 0.0000000 0.0000000 0.0000000
> [3,] 0.0 0.7130813 0.0000000 0.0000000
> [4,] 0.0 0.0000000 0.7312994 0.7487683
>
>
> $F
> $F[[1]]
> [,1] [,2] [,3] [,4]
> [1,] 0 0.0000000 0.0000000 0.1696108
> [2,] 0 0.2318637 0.2787757 0.1696108
> [3,] 0 0.0000000 0.0000000 0.0000000
> [4,] 0 0.0000000 0.0000000 0.0000000
>
> $F[[2]]
> [,1] [,2] [,3] [,4]
```

```
> [1,] 0 0.00000 0.000000 0.1695883
> [2,] 0}00.23046 0.278352 0.1695883
> [3,] 0 0.00000 0.000000 0.0000000
> [4,] 0 0.00000 0.000000 0.0000000
>
>
> $ahstages
stage_id stage original_size original_size_b original_size_c min_age
> 1 4 DormSeed
2 1 Age1 N
3 2 Age2 NA NA NA
```



```
1 [rlllllll
> 2 1 1 %rrlllll
> 3 <rrlllll
> indataset binhalfwidth_raw sizebin_min sizebin_max sizebin_center
1 0 1 1 0.5 0.5
2 1 NA NA NA
3 1 NA NA NA NA
4 1 NA NA NA NA
sizebin_width binhalfwidthb_raw sizebinb_min sizebinb_max sizebinb_center
> 1 0.5 1 1 0.5 1
>2 NA NA NA NA
> N NA NA NA NA
>4 NA NA NA NA
> sizebinb_width binhalfwidthc_raw sizebinc_min sizebinc_max sizebinc_center
> 1 0.5 1 1 0.5 1
> 2 NA NA NA NA
> NA NA NA NA NA
> 4 NA NA NA NA NA
> sizebinc_width group comments alive almostborn
> 1 0.5 0 new stage 1 0
> 2 NA 0 No description 1 0
> NA 0 No description 1 0
>4 NA 0 No description 1 0
$hstages
X1
1 NA
$agestages
    X1
    1 NA
>
$labels
    pop patch year2
1 1 1 1989
2 1 1 1990
>
```

```
> $dataqc
> [1] 345 531
>
> $matrixqc
> [1] 10 6 2
>
> $modelqc
> vital_rate individuals transitions distribution accuracy
> 1 survival 345 531 binomial 0.7608286
2 observation 0 0 bA
3 size 0 0 gaussian NA
4 sizeb 0 0 NA
5 sizec 0 0 NA
>6 reproduction 0 0 binomial NA
> 7 fecundity 16 negbin 0.8343949
> juvenile_survival 0 0 binomial NA
> 9 juvnile_observation 0 0 binomial NA
> juvenile_size 0 0 gaussian NA
> juvenile_sizeb 0 0 NA
> 12 rerrenile_sizec juvenil
> juvenile_maturity 0 0 binomial NA
>
> attr(,"class")
> [1] "lefkoMat"
```

Everything looks quite good! Congratulations on creating your first Leslie MPMs, age-hybrid MPMs, and age-by-stage MPMs!

### 6.7 Points to remember

1. Age-based (Leslie) MPMs can be be built in raw or function-based form using functions rleslie() and fleslie(), respectively. Age-hybrid MPMs can be built using the Leslie MPM creating functions in conjunction with functions add_stage() and edit_1M(). Age-by-stage MPMs can be built in raw or function-based forms using functions arlefko2() and aflefko2(), respectively.
2. Age-by-stage MPMs are larger than either age-based or stage-based MPMs, and so generally require more data to parameterize properly.
3. Not all possible ages need to be modeled in age-based and age-by-stage MPMs. Often, the final age in these MPMs represents a long stretch of the adult span of life during which vital rates are not expected to change dramatically, and so is capable of self-transition while other ages are not.
4. Age-based MPMs are not historical and cannot be turned into historical age-based MPMs.

308CHAPTER 6. MATRIX MODELS III: AGE (LESLIE), HYBRID AGE, AND AGE-BY-STAGE MPMS

## Chapter 7

## Matrix Models IV: Integral Projection Models

"If people do not believe that mathematics is simple, it is only because they do not realize how complicated life is."

\author{

- John von Neumann
}

As seen in Chapter 5, MPMs may be estimated using functions representing vital rates, where these vital rate functions may then be used to estimate each matrix element. We term this approach the function-based MPM, because of its use of functions to estimate matrix elements. One reason that this approach is very powerful is that it allows population ecologists to create simulations in which vital rates themselves are manipulated, for example via altered climate relationships or management regimes.

Easterling et al. (2000) proposed a special case of the function-based MPM called the integral projection model (IPM). In integral projection models, the familiar projection equation $\mathbf{n}_{\mathbf{t}+\mathbf{1}}=\mathbf{A} \mathbf{n}_{\mathbf{t}}$ changes to

$$
\begin{equation*}
n(k, t+1)=\int_{L}^{U} K(k, j) n(j, t) d j \tag{7.1}
\end{equation*}
$$

where an individual in state $j$ in time $t$ either transitions to state $k$ or produces offspring in state $k$ in time $t+1, n(j, t) d j$ refers to the number of individuals with their state in the range between $j$ and $j+d j, L$ and $U$ represent the lower and upper bounds of the state variable, and $K(k, j)$ is the projection kernel $K(k, j)=P(k, j)+F(k, j)$. In this projection kernel, $P(k, j)$ represents the survival-transition probability from state $j$ in time $t$ to state $k$ in time $t+1$, and $F(k, j)$ represents the production of offspring in state $k$ in time $t+1$ by an individual in state $j$ in time $t$. Because this equation is written as an integral as opposed to a discrete summation, it may appear to be quite different from the matrix approaches that we have seen so far. Indeed, there are those who have attempted to use these equations in their pure, integral forms, and when used analytically IPMs are not really MPMs. However, in practice, IPMs are typically discretized so that they may be parameterized as matrices. This discretization allows practitioners to use matrix approaches for analysis, because the projection kernel in equation 7.1 becomes perfectly analogous to the discrete projection equation $\mathbf{n}_{\mathbf{t}+\mathbf{1}}=\mathbf{A} \mathbf{n}_{\mathbf{t}}$.

Just as in other function-based MPMs, the IPM generally assumes that survival probabilities follow a binomial distribution and so may be estimated via generalized linear models, generalized linear mixed models, general additive models, or related approaches assuming a binomial response. Likewise, probabilities of reproduction, observation, or maturity, should also follow a binomial response. We see a difference in the estimation of the size probability, because IPMs require the use of a continuous size metric. In fact, the "traditional" IPM assumes a Gaussian distribution for size, and the underlying
assumptions of the Gaussian distribution make assumptions that may fail in many circumstances and hence lead to biased results. A more flexible IPM approach allows the use of other distributions, including continuous distributions such as the gamma distribution, and discrete distributions such as the Poisson or negative binomial distributions when necessary. Package lefko3 allows all of these distributions.

Discretized IPMs assume vital rate models that parameterize the main kernel populating the matrix elements. Let's review the fourteen vital rate models possible in lefko3:

1. Survival probability - This is the probability of surviving from occasion $t$ to occasion $t+1$, given that the individual is in stage $j$ in occasion $t$ (and, if historical, in stage $l$ in occasion $t-1$ ). In lefko3, this parameter may be modeled as a function of up to three size metrics, reproductive status, patch, year, age, and individual identity, and a number of individual or environmental covariates in occasions $t$ and $t$-1. This parameter is required in all function-based matrices.
2. Observation probability - This is the probability of observation in occasion $t+1$ of an individual in stage $k$ given survival from occasion $t$ to occasion $t+1$. This parameter is only used when at least one stage is technically not observable. For example, some plants are capable of vegetative dormancy, in which case they are alive but do not necessarily sprout in all years. In these cases, the probability of sprouting may be estimated as the observation probability. Note that this probability does not refer to observer effort, and so should only be used to differentiate completely unobservable stages where the observation status refers to an important biological phenomenon, such as when individuals may be alive but have a size of zero. In lefko3, this parameter may be modeled as a function of up to three size metrics, reproductive status, patch, year, age, and individual identity, and a number of individual or environmental covariates in occasions $t$ and $t-1$.
3. Primary size transition probability - This is the probability of becoming size $k$ in occasion $t+1$ assuming survival from occasion $t$ to occasion $t+1$ and observation in that time. If multiple size metrics are used, then this refers only to the first of these, which we may refer to as the primary size variable. In lefko3, this parameter may be modeled as a function of up to three size metrics, reproductive status, patch, year, age, and individual identity, and a number of individual or environmental covariates in occasions $t$ and $t$-1. This parameter is required in all function-based size-classified matrices.
4. Secondary size transition probability - This is the probability of becoming size $k$ in occasion $t+1$ assuming survival from occasion $t$ to occasion $t+1$ and observation in that time, within a second size metric used for classification in addition to the primary metric. In lefko3, this parameter may be modeled as a function of up to three size metrics, reproductive status, patch, year, age, and individual identity, and a number of individual or environmental covariates in occasions $t$ and $t-1$.
5. Tertiary size transition probability - This is the probability of becoming size $k$ in occasion $t+1$ assuming survival from occasion $t$ to occasion $t+1$ and observation in that time, within a third size metric used for classification in addition to the primary and secondary metrics. In lefko3, this parameter may be modeled as a function of up to three size metrics, reproductive status, patch, year, age, and individual identity, and a number of individual or environmental covariates in occasions $t$ and $t-1$.
6. Reproduction probability - This is the probability of becoming reproductive in occasion $t+1$ given survival from occasion $t$ to occasion $t+1$, and observation in that time. Note that this should be used only if the researcher wishes to separate breeding from non-breeding mature stages. If all adult stages are potentially reproductive and no separation of reproducing from non-reproducing adults is required by the life history model, then this parameter should not be estimated. In lefko3, this parameter may be modeled as a function of up to three size metrics, reproductive status, patch, year, age, and individual identity, and a number of individual or environmental covariates in occasions $t$ and $t-1$.
7. Fecundity rate - This refers to the rate of production of new individuals into stages in time
$t+1$ as offspring from reproduction events happening in time $t$. Under the default setting, this is the rate of successful production of offspring in occasion $t$ by individuals alive, observable, and reproductive in that time, and, if assuming a pre-breeding model and sufficient information is provided in the dataset, the survival of those offspring into occasion $t+1$ in whatever juvenile class is possible. Thus, the fecundity rate of seed-producing plants might be split into seedlings, which are plants that germinated within a year of seed production, and dormant seeds. Alternatively, it may be given only as produced fruits or seeds, with the survival and germination of seeds provided elsewhere in the MPM development process, such as within a supplement table. An additional setting allows fecundity rate to be estimated using data provided for occasion $t+1$ instead of occasion $t$. In lefko3, this parameter may be modeled as a function of up to three size metrics, reproductive status, patch, year, age, and individual identity, and a number of individual or environmental covariates in occasions $t$ and $t-1$.
8. Juvenile survival probability - This is the probability of surviving from juvenile stage $j$ in occasion $t$ to occasion $t+1$. It is used only when the user wishes to model vital rates for a single size-unclassified juvenile period separately from adults. In lefko3, this parameter may be modeled as a function of up to three size metrics, patch, year, and individual identity, and a number of individual or environmental covariates in occasions $t$ and $t-1$.
9. Juvenile observation probability - This is the probability of observation in occasion $t+1$ of an individual in juvenile stage $j$ in occasion $t$ given survival from occasion $t$ to occasion $t+1$. It is used only when the user wishes to model vital rates for a single size-unclassified juvenile period separately from adults. In lefko3, this parameter may be modeled as a function of up to three size metrics, patch, year, and individual identity, and a number of individual or environmental covariates in occasions $t$ and $t-1$.
10. Juvenile primary size transition probability - This is the probability of becoming stage $k$ in occasion $t+1$ assuming survival from juvenile stage $j$ in occasion $t$ to occasion $t+1$ and observation in that time. It is in terms of a single size metric. It is used only when the user wishes to model vital rates for a single size-unclassified juvenile period separately from adults. In lefko3, this parameter may be modeled as a function of up to three size metrics, patch, year, and a number of individual or environmental covariates in occasions $t$ and $t-1$, and individual identity.
11. Juvenile secondary size transition probability - This is the probability of becoming stage $k$ in occasion $t+1$ assuming survival from juvenile stage $j$ in occasion $t$ to occasion $t+1$ and observation in that time, in a secondary size metric in addition to the primary size metric. It is used only when the user wishes to model vital rates for a single size-unclassified juvenile period separately from adults. In lefko3, this parameter may be modeled as a function of up to three size metrics, patch, year, and individual identity, and a number of individual or environmental covariates in occasions $t$ and $t-1$.
12. Juvenile tertiary size transition probability - This is the probability of becoming stage $k$ in occasion $t+1$ assuming survival from juvenile stage $j$ in occasion $t$ to occasion $t+1$ and observation in that time, in a tertiary size metric in addition to the primary and secondary size metrics. It is used only when the user wishes to model vital rates for a single size-unclassified juvenile period separately from adults. In lefko3, this parameter may be modeled as a function of up to three size metrics, patch, year, and individual identity, and a number of individual or environmental covariates in occasions $t$ and $t-1$.
13. Juvenile reproduction probability - This is the probability of reproducing in mature stage $k$ in occasion $t+1$ given survival from juvenile stage $j$ in occasion $t$ to occasion $t+1$, and observation in that time. It is used only when the user wishes to model vital rates for a single size-unclassified juvenile period separately from adults. In lefko3, this parameter may be modeled as a function of up to three size metrics, patch, year, and individual identity, and a number of individual or environmental covariates in occasions $t$ and $t-1$.
14. Juvenile maturity probability - This is the probability of becoming mature in occasion $t+1$ given survival from juvenile stage $j$ in occasion $t$ to occasion $t+1$. It is used only when the user
wishes to model vital rates for a single size-unclassified juvenile period separately from adults. In lefko3, this parameter may be modeled as a function of up to three size metrics, patch, year, and individual identity, and a number of individual or environmental covariates in occasions $t$ and $t-1$. Note that this parameter denotes transition to maturity.

Of these fourteen vital rates, most users will estimate at least parameters (1) survival probability, (3) primary size transition probability, and (7) fecundity rate. These three are the default set for function modelsearch(). Parameters (2) observation probability and (6) reproduction probability may be used when some stages are included that are completely unobservable (and so do not have any size), or that are mature but non-reproductive, respectively. Parameters (4) secondary size transition and (5) tertiary size transition should only be used when size classification involves more than one size variable. Parameters (8) through (14) should only be added if the dataset contains juvenile individuals transitioning to maturity, and these juveniles live essentially as a single juvenile stage for some amount of time before transitioning to maturity, or before transitioning to a stage that is size-classified in the same manner as adult stages are. If juveniles can be classified by size similarly to adults (or at least on the same scale), then only vital rates (1) through (7) should be used and stage groups can be used with supplement tables to disallow transitions back to juvenile stages. If multiple juvenile stages exist on a different size classification system than adults, then stage groups may also be included as categorical variables in linear vital rate modeling in rates (1) through (7) to stratify vital rate models properly.

Let us assume that the state of the individual is represented by a continuous variable, such as a continuous size metric. This continuous size metric will be used to estimate parameter (3), the primary size transition probability. It may be Gaussian distributed, as is often assumed in IPMs (Doak et al., 2021), or may follow a different continuous distribution such as the gamma distribution. The survival-transition kernel $P(k, j)$ and the fecundity kernel $F(k, j)$ will estimated as in the functionbased MPM, as products of conditional rates or probabilities that are themselves estimated via linear models, additive models, or some other function-based approach. How, then, is the primary size transition probability estimated?

In practice, the continuous state variable is broken down into a series of continuous domains, each with its own midpoint and upper and lower bounds. Individual domains under the Gaussian and gamma distributions are shown in figures 7.1 a and d, respectively. The domain midpoints are sometimes referred to as mesh points, and together with their upper and lower bounds they compose a series of size bins of generally equal size. To approximate a continuous size, we choose a rather high number of size bins, $m$, perhaps on the order of 100 or even more.

### 7.1 Midpoint method vs. cumulative density function (CDF)

There are several methods to estimate the size transition probabilities associated with each size bin. The original method developed is referred to as the midpoint method (Doak et al., 2021). It is the default in packages such as IPMpack and in some published guides to IPM creation (Metcalf et al., 2013; Merow et al., 2014). The mesh points are then defined as $j_{i}=L+(i-0.5) h$, where $i$ is the set of integers from 1 to $m(i=1,2, \ldots, m), L$ is the lower state bound as before, and $h$ is the width of the state bin or size bin, given as $h=(U-L) / m$. If each kernel is composed of a vital rate function assuming some sort of probability distribution, and size is distributed on a Gaussian, gamma, or other continuous distribution, then $h$ accounts for the area under the distribution density curve for the corresponding vital rate contributing to the kernel at each midpoint size value used in the model (figures 7.1 b and e). If we think of the integral as being approximated by a series of rectangles under the function being integrated, then $h$ accounts for the width of the rectangle in the area approximation. Thus, we have the following.


Figure 7.1: Shaded regions indicate the true size transition probability involved in transition (a,d), the estimated size transition probability via the midpoint method (b,e), and the estimated size transition probability via $\operatorname{CDF}$ (c,f) for the Gaussian (a-c) and gamma (d-f) distributions.

$$
\begin{equation*}
n\left(x_{j}, t+1\right)=h \sum_{i=1}^{m} K\left(x_{j}, x_{i}\right) n\left(x_{i}, t\right) \tag{7.2}
\end{equation*}
$$

Doak et al. (2021) pointed out that the midpoint method yields biased results, often overestimating size transition probabilities. They proposed a second method based on the cumulative density function associated with the continuous distribution being used. We will call this the CDF method here. In this method, the cumulative probability associated with the lower and higher boundaries of the size bin are first calculated. Then, the cumulative probability associated with the lower boundary is subtracted from the cumulative probability associated with the higher boundary, yielding the exact probability associated with the size bin itself (figures 7.1 c and f ). This method does not yield biased results, and so is the default method used in lefko3 (although the midpoint method is available as an option).

The practical impact is that this approach has us creating size bins, just as in the function-based approach. If we have created size bins, then we have essentially created size-classified life history stages. Thus, equation 7.2 can be treated as a matrix projection, perfectly analogous to $\mathbf{n}_{\mathbf{t}+\mathbf{1}}=\mathbf{K} \mathbf{n}_{\mathbf{t}}$. So from a practical standpoint, an integral projection model is simply a function-based matrix projection model in which a continuous size metric determines demography. Indeed, Ellner and Rees (2006) further proposed a generalization allowing IPMs to be developed with discrete stages in some portions of the life history, referring to this as a complex integral projection model. So in practice, there is now virtually no practical difference between IPMs and function-based MPMs, although there are theoretical differences due to the assumption of integrals over continuous size in the former.

### 7.2 Creating IPMs

How do we create IPMs in package lefko3? This turns out to be quite easy. To illustrate the process, we will use the Lathyrus vernus dataset (see section 1.8.2). In this exercise, we will create both ahistorical and historical IPMs.

First, let's take a look at the dataset.

```
library(lefko3)
data(lathyrus)
summary(lathyrus)
> SUBPLOT
> Min. :1.000
> 1st Qu.:2.000
> Median :3.000 Median : 97.0 Median : 732.5 Median :6.600
> Mean :3.223 Mean :110.2 Mean : 749.4 Mean :5.538
> 3rd Qu.:4.000 3rd Qu.:167.5 3rd Qu.:1025.5 3rd Qu.:6.900
> Max. :6.000 Max. :284.0 Max. :7032.0 Max. :8.900
> NA's :404 NA's :404
> FCODE88 Flow88 Intactseed88 Dead1988
> Min. :0.0000 Min. : 1.00 Min. : 0 Mode:logical Mode:logical
> 1st Qu.:0.0000 1st Qu.: 4.00 1st Qu.: 0 NA's:1119 NA's:1119
> Median :0.0000 Median : 8.00 Median : 0
> Mean :0.3399 Mean :11.86 Mean : 3
> 3rd Qu.:1.0000 3rd Qu.:15.00 3rd Qu.: 4
> Max. :1.0000 Max. :66.00 Max. :34
> NA's :404 NA's :910 NA's :875
> Missing1988 Seedling1988 Volume89 lnVol89
> Mode:logical Min. :1.000 Min. : 1.8 Min. :0.600
```



| $>$ | 3rd Qu.: 1 | 3rd Qu.: 2.000 |  |
| :--- | :--- | :--- | :--- |
| $>$ | Max. | $: 1$ | Max. |
| $>$ | NA's | $: 1095$ | NA's |
| $>$ | $: 1082$ |  |  |

This dataset includes information on 1,119 individuals, so there are 1,119 rows with data. There are 38 columns. The first two columns give identifying information about each individual (SUBPLOT refers to the patch, and GENET refers to individual identity), with each individual's data entirely restricted to one row. This is followed by four sets of nine columns, each named VolumeXX, lnVolXX, FCODEXX, FlowXX, IntactseedXX, Dead19XX, DormantXX, Missing19XX, and SeedlingXX, where XX corresponds to the year of observation and with years organized consecutively. Thus, columns 3-11 refer to year 1988, columns 12-20 refer to year 1989, etc.

### 7.2.1 Developing stageframes for IPMs

First, we will create a stageframe for this dataset. We will base our stageframe on the life history model provided in Ehrlén (2000), but use a different size classification based on the log leaf volume to allow IPM construction and make all mature stages other than vegetative dormancy reproductive (figure 7.2).


Figure 7.2: Life history model of Lathyrus vernus. Not all adult classes are shown. Survival transitions are indicated with solid arrows, while fecundity transitions are indicated with dashed arrows.

In the stageframe code below, we show that we want an IPM by choosing two stages that serve as the size limits for the IPM's discretized size bin classification. These two size classes should have
exactly the same characteristics in the stageframe other than size. The sizes input into the sizes vector for these two stages should not be midpoints. Instead, the size for the lower limit should be the lower limit of the minimum size bin, while the size input for the upper limit should be the upper limit of the maximum size bin. By choosing these two size limits, we can skip adding and describing the many size classes that will fall between these limits - function sf_create() will create all of these for us. We mark these limits in the vector that we load into the stagenames option using the string "ipm". We then input all other characteristics for these size bins, such as observation status, maturity status, reproductive status, and these characteristics must be the same for both the minimum and maximum size bins. Package lefko3 will then create and name all IPM size classes according to its own conventions. The default number of size classes is 100 bins, and this can be altered using the ipmbins option. Note that this is essentially the same procedure described in section 2.4.

```
sizevector <- c(0, 100, 0, 1, 9)
stagevector <- c("Sd", "Sdl", "Dorm", "ipm", "ipm")
repvector <- c(0, 0, 0, 1, 1)
obsvector <- c(0, 1, 0, 1, 1)
matvector <- c(0, 0, 1, 1, 1)
immvector <- c(1, 1, 0, 0, 0)
propvector <- c(1, 0, 0, 0, 0)
indataset <- c(0, 1, 1, 1, 1)
binvec <- c(0, 100, 0.5, 0.5, 0.5)
comments <- c("Dormant seed", "Seedling", "Dormant", "ipm adult stage",
    "ipm adult stage")
lathframeipm <- sf_create(sizes = sizevector, stagenames = stagevector,
    repstatus = repvector, obsstatus = obsvector, propstatus = propvector,
    immstatus = immvector, matstatus = matvector, comments = comments,
    indataset = indataset, binhalfwidth = binvec, ipmbins = 100, roundsize = 3)
dim(lathframeipm)
> [1] 103 29
```

This stageframe has 103 stages - dormant seed, seedling, vegetative dormancy, and 100 size-classified adult stages. Let's look at just a few key columns.

| > | stage | size | sizebin_min | sizebin_max | comments |
| :---: | :---: | :---: | :---: | :---: | :---: |
| > 1 | Sd | 0.00 | 0.00 | 0.00 | Dormant seed |
| > 2 | Sdl | 100.00 | 0.00 | 200.00 | Seedling |
| > 3 | Dorm | 0.00 | -0.50 | 0.50 | Dormant |
| > 4 | sza_1.0400_0 | 1.04 | 1.00 | 1.08 | ipm adult stage |
| > 5 | sza_1.1200_0 | 1.12 | 1.08 | 1.16 | ipm adult stage |
| $>6$ | sza_1.2000_0 | 1.20 | 1.16 | 1.24 | ipm adult stage |
| > 7 | sza_1.2800_0 | 1.28 | 1.24 | 1.32 | ipm adult stage |
| $>8$ | sza_1.3600_0 | 1.36 | 1.32 | 1.40 | ipm adult stage |
| > 9 | sza_1.4400_0 | 1.44 | 1.40 | 1.48 | ipm adult stage |
| > 10 | sza_1.5200_0 | 1.52 | 1.48 | 1.56 | ipm adult stage |
| > 11 | sza_1.6000_0 | 1.60 | 1.56 | 1.64 | ipm adult stage |
| > 12 | sza_1.6800_0 | 1.68 | 1.64 | 1.72 | ipm adult stage |
| > 13 | sza_1.7600_0 | 1.76 | 1.72 | 1.80 | ipm adult stage |
| > 14 | sza_1.8400_0 | 1.84 | 1.80 | 1.88 | ipm adult stage |
| > 15 | sza_1.9200_0 | 1.92 | 1.88 | 1.96 | ipm adult stage |


| > 16 | sza_2.0000_0 | 2.00 | 1.96 | 2.04 ipm adult stage |
| :---: | :---: | :---: | :---: | :---: |
| > 17 | sza_2.0800_0 | 2.08 | 2.04 | 2.12 ipm adult stage |
| > 18 | sza_2.1600_0 | 2.16 | 2.12 | 2.20 ipm adult stage |
| 19 | sza_2.2400_0 | 2.24 | 2.20 | 2.28 ipm adult stage |
| 20 | sza_2.3200_0 | 2.32 | 2.28 | 2.36 ipm adult stage |
| 21 | sza_2.4000_0 | 2.40 | 2.36 | 2.44 ipm adult stage |
| 22 | sza_2.4800_0 | 2.48 | 2.44 | 2.52 ipm adult stage |
| 23 | sza_2.5600_0 | 2.56 | 2.52 | 2.60 ipm adult stage |
| 24 | sza_2.6400_0 | 2.64 | 2.60 | 2.68 ipm adult stage |
| 25 | sza_2.7200_0 | 2.72 | 2.68 | 2.76 ipm adult stage |
| 26 | sza_2.8000_0 | 2.80 | 2.76 | 2.84 ipm adult stage |
| 27 | sza_2.8800_0 | 2.88 | 2.84 | 2.92 ipm adult stage |
| 28 | sza_2.9600_0 | 2.96 | 2.92 | 3.00 ipm adult stage |
| > 29 | sza_3.0400_0 | 3.04 | 3.00 | 3.08 ipm adult stage |
| > 30 | sza_3.1200_0 | 3.12 | 3.08 | 3.16 ipm adult stage |
| > 31 | sza_3.2000_0 | 3.20 | 3.16 | 3.24 ipm adult stage |
| > 32 | sza_3.2800_0 | 3.28 | 3.24 | 3.32 ipm adult stage |
| > 33 | sza_3.3600_0 | 3.36 | 3.32 | 3.40 ipm adult stage |
| > 34 | sza_3.4400_0 | 3.44 | 3.40 | 3.48 ipm adult stage |
| 35 | sza_3.5200_0 | 3.52 | 3.48 | 3.56 ipm adult stage |
| 36 | sza_3.6000_0 | 3.60 | 3.56 | 3.64 ipm adult stage |
| 37 | sza_3.6800_0 | 3.68 | 3.64 | 3.72 ipm adult stage |
| 38 | sza_3.7600_0 | 3.76 | 3.72 | 3.80 ipm adult stage |
| 39 | sza_3.8400_0 | 3.84 | 3.80 | 3.88 ipm adult stage |
| 40 | sza_3.9200_0 | 3.92 | 3.88 | 3.96 ipm adult stage |
| 41 | sza_4.0000_0 | 4.00 | 3.96 | 4.04 ipm adult stage |
| 42 | sza_4.0800_0 | 4.08 | 4.04 | 4.12 ipm adult stage |
| 43 | sza_4.1600_0 | 4.16 | 4.12 | 4.20 ipm adult stage |
| 44 | sza_4.2400_0 | 4.24 | 4.20 | 4.28 ipm adult stage |
| 45 | sza_4.3200_0 | 4.32 | 4.28 | 4.36 ipm adult stage |
| > 46 | sza_4.4000_0 | 4.40 | 4.36 | 4.44 ipm adult stage |
| > 47 | sza_4.4800_0 | 4.48 | 4.44 | 4.52 ipm adult stage |
| > 48 | sza_4.5600_0 | 4.56 | 4.52 | 4.60 ipm adult stage |
| > 49 | sza_4.6400_0 | 4.64 | 4.60 | 4.68 ipm adult stage |
| > 50 | sza_4.7200_0 | 4.72 | 4.68 | 4.76 ipm adult stage |
| > 51 | sza_4.8000_0 | 4.80 | 4.76 | 4.84 ipm adult stage |
| > 52 | sza_4.8800_0 | 4.88 | 4.84 | 4.92 ipm adult stage |
| > 53 | sza_4.9600_0 | 4.96 | 4.92 | 5.00 ipm adult stage |
| > 54 | sza_5.0400_0 | 5.04 | 5.00 | 5.08 ipm adult stage |
| > 55 | sza_5.1200_0 | 5.12 | 5.08 | 5.16 ipm adult stage |
| > 56 | sza_5.2000_0 | 5.20 | 5.16 | 5.24 ipm adult stage |
| > 57 | sza_5.2800_0 | 5.28 | 5.24 | 5.32 ipm adult stage |
| > 58 | sza_5.3600_0 | 5.36 | 5.32 | 5.40 ipm adult stage |
| > 59 | sza_5.4400_0 | 5.44 | 5.40 | 5.48 ipm adult stage |
| > 60 | sza_5.5200_0 | 5.52 | 5.48 | 5.56 ipm adult stage |
| > 61 | sza_5.6000_0 | 5.60 | 5.56 | 5.64 ipm adult stage |
| > 62 | sza_5.6800_0 | 5.68 | 5.64 | 5.72 ipm adult stage |
| > 63 | sza_5.7600_0 | 5.76 | 5.72 | 5.80 ipm adult stage |
| > 64 | sza_5.8400_0 | 5.84 | 5.80 | 5.88 ipm adult stage |
| > 65 | sza_5.9200_0 | 5.92 | 5.88 | 5.96 ipm adult stage |
| > 66 | sza_6.0000_0 | 6.00 | 5.96 | 6.04 ipm adult stage |


| > 67 | sza_6.0800_0 | 6.08 | 6.04 | 6.12 ipm adult stage |
| :---: | :---: | :---: | :---: | :---: |
| > 68 | sza_6.1600_0 | 6.16 | 6.12 | 6.20 ipm adult stage |
| > 69 | sza_6.2400_0 | 6.24 | 6.20 | 6.28 ipm adult stage |
| > 70 | sza_6.3200_0 | 6.32 | 6.28 | 6.36 ipm adult stage |
| > 71 | sza_6.4000_0 | 6.40 | 6.36 | 6.44 ipm adult stage |
| > 72 | sza_6.4800_0 | 6.48 | 6.44 | 6.52 ipm adult stage |
| > 73 | sza_6.5600_0 | 6.56 | 6.52 | 6.60 ipm adult stage |
| > 74 | sza_6.6400_0 | 6.64 | 6.60 | 6.68 ipm adult stage |
| > 75 | sza_6.7200_0 | 6.72 | 6.68 | 6.76 ipm adult stage |
| $>76$ | sza_6.8000_0 | 6.80 | 6.76 | 6.84 ipm adult stage |
| > 77 | sza_6.8800_0 | 6.88 | 6.84 | 6.92 ipm adult stage |
| $>78$ | sza_6.9600_0 | 6.96 | 6.92 | 7.00 ipm adult stage |
| > 79 | sza_7.0400_0 | 7.04 | 7.00 | 7.08 ipm adult stage |
| > 80 | sza_7.1200_0 | 7.12 | 7.08 | 7.16 ipm adult stage |
| > 81 | sza_7.2000_0 | 7.20 | 7.16 | 7.24 ipm adult stage |
| > 82 | sza_7.2800_0 | 7.28 | 7.24 | 7.32 ipm adult stage |
| > 83 | sza_7.3600_0 | 7.36 | 7.32 | 7.40 ipm adult stage |
| > 84 | sza_7.4400_0 | 7.44 | 7.40 | 7.48 ipm adult stage |
| > 85 | sza_7.5200_0 | 7.52 | 7.48 | 7.56 ipm adult stage |
| > 86 | sza_7.6000_0 | 7.60 | 7.56 | 7.64 ipm adult stage |
| > 87 | sza_7.6800_0 | 7.68 | 7.64 | 7.72 ipm adult stage |
| > 88 | sza_7.7600_0 | 7.76 | 7.72 | 7.80 ipm adult stage |
| > 89 | sza_7.8400_0 | 7.84 | 7.80 | 7.88 ipm adult stage |
| > 90 | sza_7.9200_0 | 7.92 | 7.88 | 7.96 ipm adult stage |
| > 91 | sza_8.0000_0 | 8.00 | 7.96 | 8.04 ipm adult stage |
| > 92 | sza_8.0800_0 | 8.08 | 8.04 | 8.12 ipm adult stage |
| > 93 | sza_8.1600_0 | 8.16 | 8.12 | 8.20 ipm adult stage |
| > 94 | sza_8.2400_0 | 8.24 | 8.20 | 8.28 ipm adult stage |
| > 95 | sza_8.3200_0 | 8.32 | 8.28 | 8.36 ipm adult stage |
| > 96 | sza_8.4000_0 | 8.40 | 8.36 | 8.44 ipm adult stage |
| > 97 | sza_8.4800_0 | 8.48 | 8.44 | 8.52 ipm adult stage |
| > 98 | sza_8.5600_0 | 8.56 | 8.52 | 8.60 ipm adult stage |
| > 99 | sza_8.6400_0 | 8.64 | 8.60 | 8.68 ipm adult stage |
| > 100 | sza_8.7200_0 | 8.72 | 8.68 | 8.76 ipm adult stage |
| > 101 | sza_8.8000_0 | 8.80 | 8.76 | 8.84 ipm adult stage |
| > 102 | sza_8.8800_0 | 8.88 | 8.84 | 8.92 ipm adult stage |
| > 103 | sza_8.9600_0 | 8.96 | 8.92 | 9.00 ipm adult stage |

The function sf_create() has created our mesh points and associated size bins. This is in addition to the discrete stages covering the dormant seed, seedling, and dormant adult stages. Of course, we could have made this even more complex. For example, we could have created two sets of stages to use as the upper and lower bounds of two sets of continuous size states that differ in some key characteristic, such as reproductive status. We also could have set up the IPM using two or three different size metrics and used the ipm option within each or only some of them. This function provides a great deal of flexibility and power to create exactly the life history model that you may want.

### 7.2.2 Formatting demographic data and testing distribution assumptions

Next, we will format the data into hfv format. Because this is an IPM, we need to estimate linear models of vital rates. This will require us either to fix or to remove NAs in size and fecundity, so we will set NAas0 = TRUE. We will also set NRasRep = TRUE because we will assume that all adult stages other than dormancy are reproductive, and there are mature individuals in the dataset that do not
reproduce but need to be included in reproductive stages (setting this option to TRUE makes sure that the reproductive status of non-reproductive individuals in potentially reproductive stages is set to 1 , although the actual fecundity is not altered). Finally, we will ignore patches marked in the dataset and estimate matrices only for the full population in order to preserve statistical power for vital rate modeling in historical IPM analysis.

In the input to verticalize3() below, we utilize a repeating pattern of variable names arranged in the same order for each monitoring occasion. This arrangement allows us to enter only the first variable in each set, as long as noyears and blocksize are set properly and no gaps or shuffles appear in the dataset. The data management functions that we have created for lefko3 do not require such repeating patterns, but they do make the required input in the function much shorter and more succinct. Note also that we will use a new individual identity variable that incorporates the patch identity (indiv_id), to prevent repeat individual identities across patches.

```
lathyrus$indiv_id <- paste(lathyrus$SUBPLOT, lathyrus$GENET)
lathvertipm <- verticalize3(lathyrus, noyears = 4, firstyear = 1988,
    individcol = "indiv_id", blocksize = 9, juvcol = "Seedling1988",
    sizeacol = "lnVol88", repstracol = "FCODE88", fecacol = "Intactseed88",
    deadacol = "Dead1988", nonobsacol = "Dormant1988", stageassign = lathframeipm,
    stagesize = "sizea", censorcol = "Missing1988", censorkeep = NA,
    censorRepeat = TRUE, censor = TRUE, NAasO = TRUE, NRasRep = TRUE)
> Warning: Some stages occurring in the dataset do not match any characteristics
> in the input stageframe.
summary_hfv(lathvertipm)
>
> This hfv dataset contains 2527 rows, 54 variables, 1 population,
> 1 patch, }1053\mathrm{ individuals, and 3 time steps.
> Problems in stage assignment identified in rows:
>
> [1] 1366 2273
> rowid
> Min. : 1.0 Length:2527
> 1st Qu.: 237.0 Class :character
> Median : 522.0 Mode :character Mode :character Mode :character
> Mean : 537.3
> 3rd Qu.: 820.5
> Max. :1118.0
year2 firstseen
    Min. :1988 Min. : 0
    1st Qu.:1988 1st Qu.:1988
    Median :1989 Median :1988
    Mean :1989 Mean :1979
    3rd Qu.:1990 3rd Qu.:1988
    Max. :1990 Max. :1990
        sizea1 size1added repstra1 repstr1added
    Min :0.000 Min. :0.000 Min :0.0000 Min :0.0000
    1st Qu.:0.000 1st Qu.:0.000 1st Qu.:0.0000 1st Qu.:0.0000
    Median :2.200 Median :2.200 Median :0.0000 Median :0.0000
    Mean :2.957 Mean :2.957 Mean :0.1805 Mean :0.1805
    3rd Qu.:6.400 3rd Qu.:6.400 3rd Qu.:0.0000 3rd Qu.:0.0000
    Max. :8.900 Max. :8.900 Max. :1.0000 Max. :1.0000
```




Before we move on to the next steps in analysis, let's take a closer look at fecundity. In this dataset, fecundity is mostly a count of intact seeds, and only differs in six cases where the seed output was estimated based on other models. To see this, try the following code.

```
# Length of fecundity in t+1:
length(lathvertipm$feca3)
> [1] 2527
# Total non-integer entries in fecundity in occasion t+1:
length(which(lathvertipm$feca3 != round(lathvertipm$feca3)))
> [1] 0
# Length of fecundity in t:
```

```
length(lathvertipm$feca2)
> [1] 2527
# Total non-integer entries in fecundity in occasion t:
length(which(lathvertipm$feca2 != round(lathvertipm$feca2)))
> [1] 6
# Length of fecundity in t-1:
length(lathvertipm$feca1)
> [1] 2527
# Total non-integer entries in fecundity in occasion t-1:
length(which(lathvertipm$feca1 != round(lathvertipm$feca1)))
> [1] 6
```

We see that we have quite a bit of fecundity data, and that it is overwhelmingly but not exclusively integer. So, we can either treat fecundity as a continuous variable, or round the values and treat fecundity as a count variable. We will choose the latter approach in this analysis.

```
lathvertipm$feca3 <- round(lathvertipm$feca3)
lathvertipm$feca2 <- round(lathvertipm$feca2)
lathvertipm$feca1 <- round(lathvertipm$feca1)
```

Let's now look at size. Ideally, we would assume the Gaussian distribution for this continuous variable. Let's view a density plot (figure 7.3).

```
plot(density(lathvertipm$sizea2), main = "", xlab = "Log leaf volume",
    bty = "n")
```



Figure 7.3: Probability density associated with primary size, unaltered
This distribution is quite odd, but perhaps close to symmetrical. We have absorbed the size of zero into the Dormant stage, and the remaining sizes are all positive. So, we will try using the Gaussian distribution here.

Although we wish to treat fecundity as a count, it is still not clear what underlying distribution we should use. This package currently allows eight choices: Gaussian, gamma, Poisson, negative binomial, zero-inflated Poisson, zero-inflated negative binomial, zero-truncated Poisson, and zero-truncated negative binomial. To assess which to use, we should first assess whether the mean and variance of the
count are equal using a dispersion test. The Poisson distribution assumes that the mean and variance are equal, and so we can test this assumption using a chi-squared test. If it is not significantly different, then we may use some variant of the Poisson distribution. If the data are significantly over- or under-dispersed, then we should use the negative binomial distribution. If fecundity of zero is possible in reproductive stages, as in cases where reproductive status is defined by flowering rather than by offspring production, then we should also test whether the number of zeros is significantly greater than expected under these distributions, and use a zero-inflated distribution if so (if fecundity does not equal zero in any reproductive individuals at all, then we should use a zero-truncated distribution).

Let's look at a plot of the distribution of fecundity (figure 7.4).

```
hist(subset(lathvertipm, repstatus2 == 1)$feca2, main = "Fecundity",
    xlab = expression(paste("Intact seeds produced in occasion ", italic(t))))
```


## Fecundity



Figure 7.4: Histogram of fecundity

We see that the distribution seems to conform to a classic count variable with a very low mean value. The first bar suggests that there may be too many zeros to use a standard Poisson or negative binomial distribution. But to make that decision, let's formally test the assumptions that the mean equals the variance, and that there are not excess zeros. Both tests use chi-squared distribution-based approaches, with the zero-inflation test based on van den Broek (1995). This is done automatically via the $h f v_{\text {_qc }}()$ function.

```
hfv_qc(lathvertipm, vitalrates = c("surv", "obs", "size", "fec"),
    juvestimate = "Sdl", indiv = "individ", year = "year2")
> Survival:
>
> Data subset has 54 variables and 2246 transitions.
>
> Variable alive3 has O missing values.
> Variable alive3 is a binomial variable.
>
>
```

```
7.2. CREATING IPMS
```

```
> Observation status:
```

> Observation status:
>
>
> Data subset has 54 variables and 2121 transitions.
> Data subset has 54 variables and 2121 transitions.
>
>
> Variable obsstatus3 has O missing values.
> Variable obsstatus3 has O missing values.
> Variable obsstatus3 is a binomial variable.
> Variable obsstatus3 is a binomial variable.
>
>
>
>
Primary size:
Primary size:
>
>
> Data subset has 54 variables and 1916 transitions.
> Data subset has 54 variables and 1916 transitions.
>
>
> Variable sizea3 has 0 missing values.
> Variable sizea3 has 0 missing values.
> Variable sizea3 appears to be a floating point variable.
> Variable sizea3 appears to be a floating point variable.
> 1753 elements are not integers.
> 1753 elements are not integers.
> The minimum value of sizea3 is 1.2 and the maximum is 8.8.
> The minimum value of sizea3 is 1.2 and the maximum is 8.8.
> The mean value of sizea3 is 5.099 and the variance is 3.093.
> The mean value of sizea3 is 5.099 and the variance is 3.093.
> The value of the Shapiro-Wilk test of normality is 0.9551 with P = 7.719e-24.
> The value of the Shapiro-Wilk test of normality is 0.9551 with P = 7.719e-24.
> Variable sizea3 differs significantly from a Gaussian distribution.
> Variable sizea3 differs significantly from a Gaussian distribution.
>
>
> Variable sizea3 is fully positive, lacking even 0s.
> Variable sizea3 is fully positive, lacking even 0s.
>
>
>
>
> Fecundity:
> Fecundity:
>
>
> Data subset has 54 variables and 2246 transitions.
> Data subset has 54 variables and 2246 transitions.
>
>
> Variable feca2 has 0 missing values.
> Variable feca2 has 0 missing values.
> Variable feca2 appears to be an integer variable.
> Variable feca2 appears to be an integer variable.
>
>
> Variable feca2 is fully non-negative.
> Variable feca2 is fully non-negative.
>
>
> Overdispersion test:
> Overdispersion test:
> Mean feca2 is 1.282
> Mean feca2 is 1.282
> The variance in feca2 is 23.21
> The variance in feca2 is 23.21
> The probability of this dispersion level by chance assuming that
> The probability of this dispersion level by chance assuming that
> the true mean feca2 = variance in feca2,
> the true mean feca2 = variance in feca2,
> and an alternative hypothesis of overdispersion, is 0
> and an alternative hypothesis of overdispersion, is 0
> Variable feca2 is significantly overdispersed.
> Variable feca2 is significantly overdispersed.
>
>
> Zero-inflation and truncation tests:
> Zero-inflation and truncation tests:
> Mean lambda in feca2 is 0.2774
> Mean lambda in feca2 is 0.2774
> The actual number of 0s in feca2 is 1980
> The actual number of 0s in feca2 is 1980
> The expected number of 0s in feca2 under the null hypothesis is 623
> The expected number of 0s in feca2 under the null hypothesis is 623
> The probability of this deviation in Os from expectation by chance is 0
> The probability of this deviation in Os from expectation by chance is 0
> Variable feca2 is significantly zero-inflated.
> Variable feca2 is significantly zero-inflated.
>
>
>
>
> Juvenile survival:
> Juvenile survival:
>
>
> Data subset has 54 variables and 281 transitions.

```
> Data subset has 54 variables and 281 transitions.
```

```
>
> Variable alive3 has 0 missing values.
> Variable alive3 is a binomial variable.
>
>
> Juvenile observation status:
>
> Data subset has 54 variables and 210 transitions.
>
> Variable obsstatus3 has 0 missing values.
> Variable obsstatus3 is a binomial variable.
>
>
> Juvenile primary size:
>
> Data subset has 54 variables and 193 transitions.
>
> Variable sizea3 has 0 missing values.
> Variable sizea3 appears to be a floating point variable.
> 183 elements are not integers.
> The minimum value of sizea3 is 0.7 and the maximum is 4.1.
> The mean value of sizea3 is 2.307 and the variance is 0.2051.
> The value of the Shapiro-Wilk test of normality is 0.9273 with P = 3.278e-08.
> Variable sizea3 differs significantly from a Gaussian distribution.
>
> Variable sizea3 is fully positive, lacking even Os.
>
>
> Juvenile maturity status:
>
> Data subset has 54 variables and 210 transitions.
>
> Variable matstatus3 has 0 missing values.
> Variable matstatus3 is a binomial variable.
```

Such significant results for both tests show us that we should use a zero-inflated negative binomial distribution for fecundity.

Now we will create supplement tables to provide extra data for matrix estimation that is not included in the main demographic dataset. Specifically, we will provide the seed dormancy probability and germination rate, which are given as transitions from the dormant seed stage to another year of seed dormancy or to the germinated seedling stage, respectively. We assume that the germination rate is the same regardless of whether seed was produced in the previous year or has been in the seedbank for longer. We will incorporate these terms both as fixed constants for specific transitions within the resulting matrices, and as multipliers for fecundity, since ultimately fecundity will be estimated as the production of seed multiplied by the seed germination rate or the seed dormancy rate. Because some individuals stay in the seedling stage for only one year, and the seed stage itself cannot be observed and so does not exist in the dataset, we will also set a proxy set of transitions so that R assumes that the transitions from seed in occasion $t-1$ to seedling in occasion $t$ to all mature stages in occasion $t+1$ are equal to the equivalent transitions from seedling in both occasions $t-1$ and $t$.

We will start with the ahistorical case, and then move on to the historical case, where we also need to input the corresponding stages in occasion $t-1$ and transition types from occasion $t-1$ to $t$ for each transition. Note the use of the "rep", "mat", and "npr" designations in Stage1 - these are
abbreviations telling $R$ to use all reproductive stages, all mature stages, or all non-propagule stages (mature stages plus the seedling stage) in general, respectively.

```
lathsupp2 <- supplemental(stage3 = c("Sd", "Sdl", "Sd", "Sdl"),
    stage2 = c("Sd", "Sd", "rep", "rep"),
    givenrate \(=c(0.345,0.054, N A, N A)\),
    multiplier \(=c(N A, N A, 0.345,0.054)\),
    type \(=c(1,1,3,3)\), stageframe = lathframeipm, historical = FALSE)
lathsupp3 <- supplemental(stage3 = c("Sd","Sd","Sdl","Sdl","npr","Sd","Sdl"),
    stage2 = c("Sd", "Sd", "Sd", "Sd", "Sdl", "rep", "rep"),
    stage1 = c("Sd", "rep", "Sd", "rep", "Sd", "mat", "mat"),
    eststage3 \(=c(N A, N A, N A, N A, \quad\) npr", NA, NA),
    eststage \(2=c(N A, N A, N A, N A, \quad\) Sdl", NA, NA),
    eststage1 \(=c(N A, N A, N A, N A, \quad\) Sdl", NA, NA),
    givenrate \(=c(0.345,0.345,0.054,0.054, N A, N A, N A)\),
    multiplier \(=c(N A, N A, N A, N A, N A, 0.345,0.054)\),
    type \(=c(1,1,1,1,1,3,3)\), type_t12 = c(1, 2, 1, 2, 1, 1, 1),
    stageframe = lathframeipm, historical = TRUE)
lathsupp2
\begin{tabular}{lrrrrrrrrr}
\(>\) & stage3 & stage2 & stage1 & age2 & eststage3 & eststage2 & eststage1 & estage2 & givenrate \\
\(>1\) & Sd & Sd & <NA> & NA & <NA> & <NA> & <NA> & NA & 0.345 \\
\(>2\) & Sdl & Sd & <NA> & NA & <NA> & <NA> & <NA> & NA & 0.054 \\
\(>3\) & Sd & rep & <NA> & NA & <NA> & <NA> & <NA> & NA & NA \\
\(>4\) & Sdl & rep & <NA> & NA & <NA> & <NA> & <NA> & NA & NA
\end{tabular}
> multiplier convtype convtype_t12
\(\begin{array}{llll}> & \mathrm{NA} & 1 & 1\end{array}\)
\(\begin{array}{llll}>2 & \text { NA } & 1 & 1\end{array}\)
\(\begin{array}{llll}>3 & 0.345 & 3 & 1\end{array}\)
\(\begin{array}{llll}>4 & 0.054 & 3\end{array}\)
lathsupp3
\begin{tabular}{lrrrrrrrrr}
\(>\) & stage3 & stage2 & stage1 & age2 & eststage3 & eststage2 & eststage1 & estage2 & givenrate \\
\(>1\) & Sd & Sd & Sd & NA & <NA> & <NA> & <NA> & NA & 0.345 \\
\(>2\) & Sd & Sd & rep & NA & <NA> & <NA> & <NA> & NA & 0.345 \\
\(>3\) & Sdl & Sd & Sd & NA & <NA> & <NA> & <NA> & NA & 0.054 \\
\(>4\) & Sdl & Sd & rep & NA & <NA> & <NA> & <NA> & NA & 0.054 \\
\(>5\) & npr & Sdl & Sd & NA & npr & Sdl & Sdl & NA & NA \\
\(>6\) & Sd & rep & mat & NA & <NA> & <NA> & <NA> & NA & NA \\
\(>7\) & Sdl & rep & mat & NA & <NA> & <NA> & <NA> & NA & NA
\end{tabular}
> multiplier convtype convtype_t12
\(\begin{array}{llll}> & \text { NA } & 1\end{array}\)
\(\begin{array}{llll}>2 & \text { NA } & 1 & 2\end{array}\)
3 NA 1
4 NA 1
\(\begin{array}{llll}5 & \mathrm{NA} & 1 & 1\end{array}\)
\(\begin{array}{llll}>6 & 0.345 & 3 & 1\end{array}\)
\(\begin{array}{llll}>7 & 0.054 & 3 & 1\end{array}\)
```


### 7.2.3 Estimating vital rate models

Integral projection models require functions of vital rates to populate them. Here, we will develop these functions as linear models using modelsearch(). First, we will create the historical models to assess whether history is a significant influence on vital rates. Note that we have set the appropriate size and fecundity distributions through the settings sizedist = "gamma", fecdist = "negbin", and fec.zero = TRUE. Note also that we will set suite = "size", because our stageframe assumes that all sprouting stages are essentially reproductive. Thus, we cannot test the influence of reproductive status on vital rates since the resulting matrices will not have separate stages for reproductive vs. nonreproductive individuals.

```
lathmodels3ipm <- modelsearch(lathvertipm, historical = TRUE, approach= "mixed",
    suite = "size", vitalrates = c("surv", "obs", "size", "fec"),
    juvestimate = "Sdl", bestfit = "AICc&k", sizedist = "Gaussian",
    fecdist = "negbin", fec.zero = TRUE, indiv = "individ", year = "year2",
    year.as.random = TRUE, juvsize = TRUE, show.model.tables = TRUE,
    quiet = "partial")
```

Now let's see a summary.

```
summary(lathmodels3ipm)
> This LefkoMod object includes 6 linear models.
    Best-fit model criterion used: aicc&k
>
>
>
Survival model:
Generalized linear mixed model fit by maximum likelihood (Laplace
        Approximation) [glmerMod]
    Family: binomial ( logit )
    Formula: alive3 ~ (1 | year2) + (1 | individ)
        Data: subdata
            AIC BIC logLik deviance df.resid
    709.9383 727.0890 -351.9691 703.9383 2243
Random effects:
    Groups Name Std.Dev.
    individ (Intercept) 14.6
    year2 (Intercept) 0.0
    Number of obs: 2246, groups: individ, 931; year2, 3
    Fixed Effects:
    (Intercept)
    10.03
    optimizer (Nelder_Mead) convergence code: O (OK) ; O optimizer warnings; 1 lme4 warnings
>
>
Observation model:
Generalized linear mixed model fit by maximum likelihood (Laplace
        Approximation) [glmerMod]
    Family: binomial ( logit )
    Formula: obsstatus3 ~ sizea1 + sizea2 + (1 | year2) + (1 | individ)
    Data: subdata
```

```
> AIC BIC logLik deviance df.resid
> 1305.000 1333.298-647.500 1295.000 2116
> Random effects:
> Groups Name Std.Dev.
> individ (Intercept) 9.194
> year2 (Intercept) 0.000
Number of obs: 2121, groups: individ, 858; year2, 3
Fixed Effects:
> (Intercept) sizea1 sizea2
> 14.5041 -0.3982 -0.7603
optimizer (Nelder_Mead) convergence code: O (OK) ; 0 optimizer warnings; 1 lme4 warnings
>
>
>
Size model:
Linear mixed model fit by REML ['lmerMod']
Formula: sizea3 ~ sizea1 + sizea2 + (1 | year2) + (1 | individ) + sizea1:sizea2
    Data: subdata
REML criterion at convergence: 5242.508
Random effects:
Groups Name Std.Dev.
individ (Intercept) 0.7265
year2 (Intercept) 0.6006
Residual 0.7323
Number of obs: 1916, groups: individ, 845; year2, 3
Fixed Effects:
    (Intercept) sizea1 sizea2 sizea1:sizea2
        1.04671 0.57845 0.68329 -0.07218
Secondary size model:
[1] 1
>
>
>
> Tertiary size model:
[1] 1
>
>
>
Reproductive status model:
[1] 1
>
>
>
> Fecundity model:
Formula: feca2 ~ sizea1 + sizea2 + (1 | year2) + (1 | individ)
Zero inflation:
~sizea1 + sizea2 + (1 | year2) + (1 | individ) + sizea1:sizea2
Data: subdata
```

```
> AIC BIC logLik df.resid
    2464.397 2517.140 -1220.198 587
Random-effects (co)variances:
Conditional model:
    Groups Name Std.Dev.
    year2 (Intercept) 2.139e-05
    individ (Intercept) 3.504e-01
Zero-inflation model:
    Groups Name Std.Dev.
    year2 (Intercept) 0.5487
    individ (Intercept) 0.4465
Number of obs: 599 / Conditional model: year2, 3; individ, 335 / Zero-inflation model: year2, 3; in
Dispersion parameter for nbinom2 family (): 2.77
Fixed Effects:
Conditional model:
(Intercept) sizea1 sizea2
        -2.49973 0.03908 0.62804
Zero-inflation model:
        (Intercept) sizea1 sizea2 sizea1:sizea2
            0.57865 0.87209 -0.03545 -0.12648
    Juvenile survival model:
Generalized linear mixed model fit by maximum likelihood (Laplace
        Approximation) [glmerMod]
    Family: binomial ( logit )
Formula: alive3 ~ (1 | year2) + (1 | individ)
        Data: subdata
            AIC BIC logLik deviance df.resid
    323.6696 334.5847 -158.8348 317.6696 278
Random effects:
    Groups Name Std.Dev.
    individ (Intercept) 0.0003658
    year2 (Intercept) 0.0000000
Number of obs: 281, groups: individ, 281; year2, 3
Fixed Effects:
(Intercept)
            1.084
optimizer (Nelder_Mead) convergence code: O (OK) ; O optimizer warnings; 1 lme4 warnings
Juvenile observation model:
Generalized linear mixed model fit by maximum likelihood (Laplace
```

```
> Approximation) [glmerMod]
> Family: binomial ( logit )
Formula: obsstatus3 ~ sizea2 + (1 | year2) + (1 | individ)
> Data: subdata
> AIC BIC logLik deviance df.resid
61.3610 74.7495 -26.6805 53.3610 206
Random effects:
Groups Name Std.Dev.
individ (Intercept) 52.8210
year2 (Intercept) 0.1885
Number of obs: 210, groups: individ, 210; year2, 3
Fixed Effects:
> (Intercept) sizea2
> 11.7595 0.6446
>
>
>
Juvenile size model:
[1] 1
>
>
>
> Juvenile secondary size model:
[1] 1
>
>
>
> Juvenile tertiary size model:
[1] 1
Juvenile reproduction model:
[1] 0
>
>
>
> Juvenile maturity model:
[1] 1
>
>
>
>
>
> Number of models in survival table: 4
Number of models in observation table: 5
Number of models in size table: 5
Number of models in secondary size table: 1
```

```
>
Number of models in tertiary size table: 1
>
Number of models in reproduction status table: 1
Number of models in fecundity table: 25
Number of models in juvenile survival table: 2
Number of models in juvenile observation table: 2
Number of models in juvenile size table: 1
Number of models in juvenile secondary size table: 1
Number of models in juvenile tertiary size table: 1
Number of models in juvenile reproduction table: 1
Number of models in juvenile maturity table: 1
General model parameter names (column 1), and
specific names used in these models (column 2):
            parameter_names mainparams
                    time t year2
                    individual individ
                    patch patch
                alive in time t+1 surv3
            observed in time t+1 obs3
                sizea in time t+1 size3
                sizeb in time t+1 sizeb3
                sizec in time t+1 sizec3
            reproductive status in time t+1 repst3
                fecundity in time t+1 fec3
                    fecundity in time t fec2
                    sizea in time t size2
                    sizea in time t-1 size1
                    sizeb in time t sizeb2
                    sizeb in time t-1 sizeb1
                    sizec in time t sizec2
                    sizec in time t-1 sizec1
        reproductive status in time t repst2
        reproductive status in time t-1 repst1
            maturity status in time t+1 matst3
            maturity status in time t matst2
                    age in time t age
                    density in time t density
```

```
> 24 individual covariate a in time t indcova2
> 25 individual covariate a in time t-1 indcova1
> 26 individual covariate b in time t indcovb2
> 27 individual covariate b in time t-1 indcovb1
> 28 individual covariate c in time t indcovc2
> 29 individual covariate c in time t-1 indcovc1
> 30 stage group in time t group2
>31 stage group in time t-1 group1
>
>
>
>
>
> Quality control:
>
> Survival model estimated with 931 individuals and 2246 individual transitions.
> Survival model accuracy is 0.977.
> Observation status model estimated with }858\mathrm{ individuals and 2121 individual transitions.
> Observation status model accuracy is 0.959.
> Primary size model estimated with }845\mathrm{ individuals and 1916 individual transitions.
> Primary size model R-squared is 0.878.
> Secondary size model not estimated.
> Tertiary size model not estimated.
> Reproductive status model not estimated.
> Fecundity model estimated with 335 individuals and 599 individual transitions.
> Fecundity model R-squared is 0.368.
> Juvenile survival model estimated with 281 individuals and 281 individual transitions.
> Juvenile survival model accuracy is 0.747.
> Juvenile observation status model estimated with 210 individuals and 210 individual transitions.
> Juvenile observation status model accuracy is 1.
> Juvenile primary size model not estimated.
> Juvenile secondary size model not estimated.
> Juvenile tertiary size model not estimated.
> Juvenile reproductive status model not estimated.
> Juvenile maturity status model not estimated.
```

We see the influence of history on survival and size and fecundity. So, the historical IPM is the correct choice here, although the $\mathrm{R}^{2}$ for primary size and fecundity are both quite low. However, we will also create an ahistorical IPM for comparison. For that purpose, we will create the ahistorical vital rate model set.

```
lathmodels2ipm <- modelsearch(lathvertipm, historical = FALSE,
    approach = "mixed", suite = "size", juvestimate = "Sdl",
    vitalrates = c("surv", "obs", "size", "fec"), bestfit = "AICc&k",
    sizedist = "gaussian", fecdist = "negbin", fec.zero = TRUE, indiv = "individ",
    year = "year2", year.as.random = TRUE, juvsize = TRUE,
    show.model.tables = TRUE, quiet = "partial")
```

Let's see a summary.

```
summary(lathmodels2ipm)
> This LefkoMod object includes 5 linear models.
Best-fit model criterion used: aicc&k
>
>
>
Survival model:
[1] 1
>
>
>
Observation model:
Generalized linear mixed model fit by maximum likelihood (Laplace
    Approximation) [glmerMod]
Family: binomial ( logit )
Formula: obsstatus3 ~ sizea2 + (1 | year2) + (1 | individ)
            Data: subdata
            AIC BIC logLik deviance df.resid
1330.9225 1353.5610 -661.4612 1322.9225 2117
Random effects:
    Groups Name Std.Dev.
    individ (Intercept) 1.614
    year2 (Intercept) 0.000
Number of obs: 2121, groups: individ, 858; year2, 3
Fixed Effects:
(Intercept) sizea2
3.7622 -0.1323
optimizer (Nelder_Mead) convergence code: O (OK) ; 0 optimizer warnings; 1 lme4 warnings
>
>
> Size model:
Linear mixed model fit by REML ['lmerMod']
Formula: sizea3 ~ sizea2 + (1 | year2) + (1 | individ)
    Data: subdata
REML criterion at convergence: 5713.054
Random effects:
Groups Name Std.Dev.
individ (Intercept) 1.4436
year2 (Intercept) 0.3356
Residual 0.6087
Number of obs: 1916, groups: individ, 845; year2, 3
Fixed Effects:
(Intercept) sizea2
    4.1922 0.1488
Secondary size model:
[1] 1
>
```

```
>
>
Tertiary size model:
[1] 1
>
>
>
Reproductive status model:
[1] 1
>
>
>
Fecundity model:
Formula: feca2 ~ sizea2 + (1 | year2) + (1 | individ)
Zero inflation: ~sizea2 + (1 | year2) + (1 | individ)
Data: subdata
    AIC BIC logLik df.resid
2470.822 2510.380 -1226.411 590
Random-effects (co)variances:
Conditional model:
Groups Name Std.Dev.
year2 (Intercept) 0.1089
individ (Intercept) 0.3613
Zero-inflation model:
    Groups Name Std.Dev.
    year2 (Intercept) 0.6527
    individ (Intercept) 0.4508
Number of obs: 599 / Conditional model: year2, 3; individ, 335 / Zero-inflation model: year2, 3; in
Dispersion parameter for nbinom2 family (): 2.77
Fixed Effects:
Conditional model:
(Intercept) sizea2
    -2.5616 0.6606
Zero-inflation model:
(Intercept) sizea2
    3.9299 -0.5208
>
>
Juvenile survival model:
Generalized linear mixed model fit by maximum likelihood (Laplace
    Approximation) [glmerMod]
    Family: binomial ( logit )
Formula: alive3 ~ (1 | year2) + (1 | individ)
    Data: subdata
```

```
> AIC BIC logLik deviance df.resid
    323.6696 334.5847 -158.8348 317.6696 278
Random effects:
Groups Name Std.Dev.
    individ (Intercept) 0.0003658
    year2 (Intercept) 0.0000000
Number of obs: 281, groups: individ, 281; year2, 3
Fixed Effects:
(Intercept)
> 1.084
optimizer (Nelder_Mead) convergence code: 0 (OK) ; 0 optimizer warnings; 1 lme4 warnings
>
>
>
Juvenile observation model:
Generalized linear mixed model fit by maximum likelihood (Laplace
    Approximation) [glmerMod]
    Family: binomial ( logit )
Formula: obsstatus3 ~ sizea2 + (1 | year2) + (1 | individ)
            Data: subdata
            AIC BIC logLik deviance df.resid
    61.3610 74.7495 -26.6805 53.3610 206
Random effects:
    Groups Name Std.Dev.
    individ (Intercept) 52.8210
    year2 (Intercept) 0.1885
    Number of obs: 210, groups: individ, 210; year2, 3
    Fixed Effects:
    (Intercept) sizea2
        11.7595 0.6446
Juvenile size model:
[1] 1
>
>
>
> Juvenile secondary size model:
[1] 1
>
>
>
> Juvenile tertiary size model:
[1] 1
>
>
>
> Juvenile reproduction model:
[1] 0
>
```

```
>
```

>
>
>
Juvenile maturity model:
Juvenile maturity model:
[1] 1
[1] 1
>
>
>
>
>
>
>
>
>
>
Number of models in survival table: 1
Number of models in survival table: 1
Number of models in observation table: 2
Number of models in observation table: 2
Number of models in size table: 2
Number of models in size table: 2
Number of models in secondary size table: 1
Number of models in secondary size table: 1
Number of models in tertiary size table: 1
Number of models in tertiary size table: 1
Number of models in reproduction status table: 1
Number of models in reproduction status table: 1
Number of models in fecundity table: 4
Number of models in fecundity table: 4
Number of models in juvenile survival table: 2
Number of models in juvenile survival table: 2
Number of models in juvenile observation table: 2
Number of models in juvenile observation table: 2
Number of models in juvenile size table: 1
Number of models in juvenile size table: 1
Number of models in juvenile secondary size table: 1
Number of models in juvenile secondary size table: 1
Number of models in juvenile tertiary size table: 1
Number of models in juvenile tertiary size table: 1
Number of models in juvenile reproduction table: 1
Number of models in juvenile reproduction table: 1
Number of models in juvenile maturity table: 1
Number of models in juvenile maturity table: 1
General model parameter names (column 1), and
General model parameter names (column 1), and
specific names used in these models (column 2):
specific names used in these models (column 2):
parameter_names mainparams
parameter_names mainparams
time t year2
time t year2
individual individ
individual individ
patch patch
patch patch
alive in time t+1 surv3
alive in time t+1 surv3
>
>
>
>
> 6
> 6
> 7

```
> 7
```

```
>8 sizec in time t+1 sizec3
>9 reproductive status in time t+1 repst3
> 10 fecundity in time t+1 fec3
> 11 fecundity in time t fec2
> 12 sizea in time t size2
> 13 sizea in time t-1 size1
> 14 sizeb in time t sizeb2
> 15 sizeb in time t-1 sizeb1
    16 sizec in time t sizec2
            sizec in time t-1 sizec1
        reproductive status in time t repst2
        reproductive status in time t-1 repst1
            maturity status in time t+1 matst3
            maturity status in time t matst2
                age in time t age
            density in time t density
    individual covariate a in time t indcova2
    individual covariate a in time t-1 indcova1
        individual covariate b in time t indcovb2
    individual covariate b in time t-1 indcovb1
        individual covariate c in time t indcovc2
    individual covariate c in time t-1 indcovc1
            stage group in time t group2
            stage group in time t-1 group1
> 31
>
>
>
>
>
> Quality control:
>
> Survival model not estimated.
> Observation status model estimated with }858\mathrm{ individuals and 2121 individual transitions.
> Observation status model accuracy is 0.904.
> Primary size model estimated with }845\mathrm{ individuals and 1916 individual transitions.
> Primary size model R-squared is 0.929.
> Secondary size model not estimated.
> Tertiary size model not estimated.
> Reproductive status model not estimated.
> Fecundity model estimated with 335 individuals and 599 individual transitions.
> Fecundity model R-squared is 0.364.
> Juvenile survival model estimated with 281 individuals and 281 individual transitions.
> Juvenile survival model accuracy is 0.747.
> Juvenile observation status model estimated with 210 individuals and 210 individual transitions.
> Juvenile observation status model accuracy is 1.
> Juvenile primary size model not estimated.
> Juvenile secondary size model not estimated.
> Juvenile tertiary size model not estimated.
> Juvenile reproductive status model not estimated.
> Juvenile maturity status model not estimated.
```

We see a few differences, for example that survival is now set to 1 due to ill-conditioning in some of
the data. Let's go on and make our IPMs.

### 7.2.4 Bringing our discretized IPMs to life

The typical IPM is ahistorical and so will utilize only an ahistorical set of vital rate models to populate its matrices. Let's do that and take a look at the result.

```
lathmat2ipm <- flefko2(stageframe = lathframeipm, modelsuite = lathmodels2ipm,
    supplement = lathsupp2, data = lathvertipm, reduce = FALSE)
summary(lathmat2ipm)
>
> This ahistorical lefkoMat object contains 3 matrices.
>
> Each matrix is square with 103 rows and columns, and a total of 10609 elements.
> A total of }30608\mathrm{ survival transitions were estimated, with 10202.667 per matrix.
> A total of }600\mathrm{ fecundity transitions were estimated, with 200 per matrix.
> This lefkoMat object covers 1 population, 1 patch, and 3 time steps.
>
> The dataset contains a total of }1053\mathrm{ unique individuals and 2527 unique transitions.
>
Vital rate modeling quality control:
>
> Survival not estimated.
Observation estimated with }858\mathrm{ individuals and 2121 individual transitions.
Primary size estimated with }845\mathrm{ individuals and 1916 individual transitions.
Secondary size transition not estimated.
Tertiary size transition not estimated.
Reproduction probability not estimated.
Fecundity estimated with }335\mathrm{ individuals and }599\mathrm{ individual transitions.
Juvenile survival estimated with }281\mathrm{ individuals and 281 individual transitions.
Juvenile observation estimated with 210 individuals and 210 individual transitions.
Juvenile primary size transition not estimated.
Juvenile secondary size transition not estimated.
> Juvenile tertiary size transition not estimated.
> Juvenile reproduction probability not estimated.
Juvenile maturity transition probability not estimated.
Survival probability sum check (each matrix represented by column in order):
> [,1] [,2] [,3]
Min. 0.000 0.000 0.000
1st Qu. 1.000 1.000 1.000
Median 1.000 1.000 1.000
Mean 0.984 0.984 0.984
3rd Qu. 1.000 1.000 1.000
Max. 1.000 1.000 1.000
```

The ahistorical IPM is composed of three matrices, covering each of the time steps. These are large matrices - with 103 rows and columns, they include 10,609 elements each. Of these, on average 10402.67 elements in each matrix are non-zero, meaning that these matrices are not only large but also quite dense ( $98.1 \%$ of elements are estimated).

We will now create the historical suite of matrices covering the years of study. These matrices will be extremely large - large enough that some computers might have difficulty with them. To help, we
will create them in sparse matrix format. If you encounter an error message telling you that you have run out of memory, then please try this on a more powerful computer :).

```
lathmat3ipm <- flefko3(stageframe = lathframeipm, modelsuite = lathmodels3ipm,
    supplement = lathsupp3, data = lathvertipm, reduce = FALSE,
    sparse_output = TRUE)
summary(lathmat3ipm)
>
> This historical lefkoMat object contains 3 matrices.
>
> Each matrix is square with }10609\mathrm{ rows and columns, and a total of 112550881 elements.
A total of 3106216 survival transitions were estimated, with 1035405.333 per matrix.
A total of }60600\mathrm{ fecundity transitions were estimated, with }20200\mathrm{ per matrix.
This lefkoMat object covers 1 population, 1 patch, and 3 time steps.
The dataset contains a total of }1053\mathrm{ unique individuals and 2527 unique transitions.
Vital rate modeling quality control:
>
Survival estimated with }931\mathrm{ individuals and 2246 individual transitions.
Observation estimated with }858\mathrm{ individuals and 2121 individual transitions.
Primary size estimated with }845\mathrm{ individuals and 1916 individual transitions.
Secondary size transition not estimated.
Tertiary size transition not estimated.
Reproduction probability not estimated.
Fecundity estimated with }335\mathrm{ individuals and }599\mathrm{ individual transitions.
Juvenile survival estimated with }281\mathrm{ individuals and 281 individual transitions.
Juvenile observation estimated with 210 individuals and 210 individual transitions.
Juvenile primary size transition not estimated.
Juvenile secondary size transition not estimated.
Juvenile tertiary size transition not estimated.
Juvenile reproduction probability not estimated.
Juvenile maturity transition probability not estimated.
Survival probability sum check (each matrix represented by column in order):
> [,1] [,2] [,3]
Min. 0.000 0.000 0.000
1st Qu. 0.997 1.000 1.000
Median 1.000 1.000 1.000
Mean 0.973 0.972 0.974
3rd Qu. 1.000 1.000 1.000
Max. 1.000 1.000 1.000
```

These are giant matrices. With 10,609 rows and columns, there are a total of 112,550,881 elements per matrix. But they are also amazingly sparse - with $1,055,605$ elements estimated, only $0.9 \%$ of elements per matrix are non-zero. The survival probability sums all look good, so we appear to have no problems with overly large given and proxy survival transitions provided through our supplemental tables.

At this stage, we have created our IPMs. Congratulations! We can also create arithmetic mean matrix versions of each, as below.

```
lath2ipmmean <- lmean(lathmat2ipm)
lath3ipmmean <- lmean(lathmat3ipm)
summary(lath2ipmmean)
>
This ahistorical lefkoMat object contains 1 matrix.
Each matrix is square with 103 rows and columns, and a total of 10609 elements.
A total of 10203 survival transitions were estimated, with }10203\mathrm{ per matrix.
A total of 200 fecundity transitions were estimated, with 200 per matrix.
> This lefkoMat object covers 1 population, 1 patch, and O time steps.
>
> The dataset contains a total of 1053 unique individuals and 2527 unique transitions.
>
Vital rate modeling quality control:
>
Survival not estimated.
Observation estimated with }858\mathrm{ individuals and 2121 individual transitions.
Primary size estimated with }845\mathrm{ individuals and 1916 individual transitions.
Secondary size transition not estimated.
Tertiary size transition not estimated.
Reproduction probability not estimated.
Fecundity estimated with 335 individuals and 599 individual transitions.
Juvenile survival estimated with }281\mathrm{ individuals and 281 individual transitions.
Juvenile observation estimated with 210 individuals and 210 individual transitions.
Juvenile primary size transition not estimated.
Juvenile secondary size transition not estimated.
Juvenile tertiary size transition not estimated.
Juvenile reproduction probability not estimated.
Juvenile maturity transition probability not estimated.
Survival probability sum check (each matrix represented by column in order):
[,1]
Min. 0.000
1st Qu. 1.000
Median 1.000
Mean 0.984
3rd Qu. 1.000
Max. 1.000
ummary(lath3ipmmean)
>
This historical lefkoMat object contains 1 matrix.
Each matrix is square with }10609\mathrm{ rows and columns, and a total of 112550881 elements.
A total of 1036775 survival transitions were estimated, with }1036775\mathrm{ per matrix.
A total of 20200 fecundity transitions were estimated, with 20200 per matrix.
This lefkoMat object covers 1 population, 1 patch, and O time steps.
The dataset contains a total of 1053 unique individuals and 2527 unique transitions.
Vital rate modeling quality control:
```

```
>
> Survival estimated with 931 individuals and 2246 individual transitions.
> Observation estimated with }858\mathrm{ individuals and 2121 individual transitions.
> Primary size estimated with }845\mathrm{ individuals and 1916 individual transitions.
> Secondary size transition not estimated.
> Tertiary size transition not estimated.
> Reproduction probability not estimated.
> Fecundity estimated with 335 individuals and 599 individual transitions.
> Juvenile survival estimated with }281\mathrm{ individuals and 281 individual transitions.
> Juvenile observation estimated with 210 individuals and 210 individual transitions.
> Juvenile primary size transition not estimated.
> Juvenile secondary size transition not estimated.
> Juvenile tertiary size transition not estimated.
> Juvenile reproduction probability not estimated.
> Juvenile maturity transition probability not estimated.
>
> Survival probability sum check (each matrix represented by column in order):
> [,1]
> Min. 0.000
> 1st Qu. 0.999
> Median 1.000
> Mean 0.973
> 3rd Qu. 1.000
> Max. 1.000
```


### 7.3 Quality control

IPMs are difficult to inspect because of their size. Package lefko3 includes a number of ways to assess the overall quality of an IPM. Here, we will cover three main methods, each covering a different aspect of the process. First, we may look at quality control information about our vital rate models. Let's look at a summary of the historical vital rate models.

```
summary(lathmodels3ipm)
> This LefkoMod object includes 6 linear models.
> Best-fit model criterion used: aicc&k
>
>
>
> Survival model:
> Generalized linear mixed model fit by maximum likelihood (Laplace
> Approximation) [glmerMod]
> Family: binomial ( logit )
> Formula: alive3 ~ (1 | year2) + (1 | individ)
> Data: subdata
> AIC BIC logLik deviance df.resid
> 709.9383 727.0890 -351.9691 703.9383 2243
> Random effects:
> Groups Name Std.Dev.
> individ (Intercept) 14.6
> year2 (Intercept) 0.0
> Number of obs: 2246, groups: individ, 931; year2, 3
```

```
Fixed Effects:
(Intercept)
10.03
optimizer (Nelder_Mead) convergence code: O (OK) ; O optimizer warnings; 1 lme4 warnings
>
>
>
Observation model:
Generalized linear mixed model fit by maximum likelihood (Laplace
        Approximation) [glmerMod]
    Family: binomial ( logit )
Formula: obsstatus3 ~ sizea1 + sizea2 + (1 | year2) + (1 | individ)
            Data: subdata
            AIC BIC logLik deviance df.resid
1305.000 1333.298-647.500 1295.000 2116
Random effects:
    Groups Name Std.Dev.
    individ (Intercept) 9.194
    year2 (Intercept) 0.000
Number of obs: 2121, groups: individ, 858; year2, 3
Fixed Effects:
(Intercept) sizea1 sizea2
    14.5041 -0.3982 -0.7603
optimizer (Nelder_Mead) convergence code: O (OK) ; 0 optimizer warnings; 1 lme4 warnings
>
>
>
Size model:
Linear mixed model fit by REML ['lmerMod']
Formula: sizea3 ~ sizea1 + sizea2 + (1 | year2) + (1 | individ) + sizea1:sizea2
    Data: subdata
REML criterion at convergence: 5242.508
Random effects:
    Groups Name Std.Dev.
individ (Intercept) 0.7265
year2 (Intercept) 0.6006
Residual 0.7323
Number of obs: 1916, groups: individ, 845; year2, 3
Fixed Effects:
    (Intercept) sizea1 sizea2 sizea1:sizea2
        1.04671 0.57845 0.68329 -0.07218
Secondary size model:
[1] 1
Tertiary size model:
[1] 1
```

```
>
>
>
Reproductive status model:
[1] 1
```



```
>
>
Fecundity model:
Formula: feca2 ~ sizea1 + sizea2 + (1 | year2) + (1 | individ)
Zero inflation:
~sizea1 + sizea2 + (1 | year2) + (1 | individ) + sizea1:sizea2
Data: subdata
AIC BIC logLik df.resid
2464.397 2517.140 -1220.198 587
Random-effects (co)variances:
Conditional model:
    Groups Name Std.Dev.
    year2 (Intercept) 2.139e-05
    individ (Intercept) 3.504e-01
Zero-inflation model:
    Groups Name Std.Dev.
    year2 (Intercept) 0.5487
    individ (Intercept) 0.4465
Number of obs: 599 / Conditional model: year2, 3; individ, 335 / Zero-inflation model: year2, 3; in
Dispersion parameter for nbinom2 family (): 2.77
Fixed Effects:
Conditional model:
(Intercept) sizea1 sizea2
    -2.49973 0.03908 0.62804
Zero-inflation model:
    (Intercept) sizea1 sizea2 sizea1:sizea2
            0.57865 0.87209 -0.03545 -0.12648
    Juvenile survival model:
Generalized linear mixed model fit by maximum likelihood (Laplace
        Approximation) [glmerMod]
    Family: binomial ( logit )
Formula: alive3 ~ (1 | year2) + (1 | individ)
        Data: subdata
            AIC BIC logLik deviance df.resid
    323.6696 334.5847-158.8348 317.6696 278
Random effects:
```

```
Groups Name Std.Dev.
individ (Intercept) 0.0003658
year2 (Intercept) 0.0000000
Number of obs: 281, groups: individ, 281; year2, 3
Fixed Effects:
> (Intercept)
1.084
optimizer (Nelder_Mead) convergence code: O (OK) ; O optimizer warnings; 1 lme4 warnings
Juvenile observation model:
Generalized linear mixed model fit by maximum likelihood (Laplace
    Approximation) [glmerMod]
    Family: binomial ( logit )
Formula: obsstatus3 ~ sizea2 + (1 | year2) + (1 | individ)
        Data: subdata
        AIC BIC logLik deviance df.resid
    61.3610 74.7495 -26.6805 53.3610 206
Random effects:
    Groups Name Std.Dev.
    individ (Intercept) 52.8210
    year2 (Intercept) 0.1885
Number of obs: 210, groups: individ, 210; year2, 3
Fixed Effects:
(Intercept) sizea2
11.7595 0.6446
Juvenile size model:
[1] 1
> Juvenile secondary size model:
[1] 1
> Juvenile tertiary size model:
[1] 1
> Juvenile reproduction model:
[1] 0
> Juvenile maturity model:
```

$>$
$>$
$>$
$>$
$>$
$>$
$>$
$>$
$>$
$>$
$>$
$>$
$>$
$>$
$>$

```
[1] 1
>
>
>
>
>
Number of models in survival table: 4
Number of models in observation table: 5
Number of models in size table: 5
Number of models in secondary size table: 1
Number of models in tertiary size table: 1
Number of models in reproduction status table: 1
Number of models in fecundity table: 25
Number of models in juvenile survival table: 2
Number of models in juvenile observation table: 2
Number of models in juvenile size table: 1
Number of models in juvenile secondary size table: 1
Number of models in juvenile tertiary size table: 1
Number of models in juvenile reproduction table: 1
Number of models in juvenile maturity table: 1
> General model parameter names (column 1), and
specific names used in these models (column 2):
            parameter_names mainparams
                                    time t year2
                    individual individ
                                    patch patch
4 ralive in time t+1 surv3
5 observed in time t+1 obs3
> 6 sizea in time t+1 size3
> 7 sizeb in time t+1 sizeb3
> sizec in time t+1 sizec3
>9 reproductive status in time t+1 repst3
10
                        fecundity in time t+1 fec3
```

```
>11 fecundity in time t fec2
> 12 sizea in time t size2
> 13 sizea in time t-1 size1
> 14 sizeb in time t sizeb2
> 15 sizeb in time t-1 sizeb1
> 16 sizec in time t sizec2
17 sizec in time t-1 sizec1
> 18 reproductive status in time t repst2
> 19 reproductive status in time t-1 repst1
> 20 maturity status in time t+1 matst3
> 21 maturity status in time t matst2
>22 age in time t age
>23 density in time t density
> 24 individual covariate a in time t indcova2
> 25 individual covariate a in time t-1 indcova1
> 26 individual covariate b in time t indcovb2
> 27 individual covariate b in time t-1 indcovb1
> 28 individual covariate c in time t indcovc2
> 29 individual covariate c in time t-1 indcovc1
    30 stage group in time t group2
    31 stage group in time t-1 group1
>
>
>
>
>
> Quality control:
>
> Survival model estimated with 931 individuals and 2246 individual transitions.
> Survival model accuracy is 0.977.
> Observation status model estimated with }858\mathrm{ individuals and 2121 individual transitions.
> Observation status model accuracy is 0.959.
> Primary size model estimated with }845\mathrm{ individuals and 1916 individual transitions.
> Primary size model R-squared is 0.878.
> Secondary size model not estimated.
> Tertiary size model not estimated.
> Reproductive status model not estimated.
> Fecundity model estimated with 335 individuals and 599 individual transitions.
> Fecundity model R-squared is 0.368.
> Juvenile survival model estimated with 281 individuals and 281 individual transitions.
> Juvenile survival model accuracy is 0.747.
> Juvenile observation status model estimated with 210 individuals and 210 individual transitions.
> Juvenile observation status model accuracy is 1.
> Juvenile primary size model not estimated.
> Juvenile secondary size model not estimated.
> Juvenile tertiary size model not estimated.
> Juvenile reproductive status model not estimated.
> Juvenile maturity status model not estimated.
```

At the very bottom of our output is a section labelled Quality control. We see, first of all, a statement of which of our fourteen possible vital rate models were estimated. For each estimated model, we see the number of individuals and actual transitions used to estimate the respective model.

In general, the higher the number of individuals and transitions used to estimate the model, the better the quality of the model and the higher the statistical power. The former number, the number of individuals, particularly gives us a sense of the overall level of pseudoreplication that might be inherent in our analysis, since transitions from the same individual are obviously related and not statistically independent of one another.

Our output also includes information on the accuracy of binomial models and simple $R^{2}$ of size and fecundity models. Accuracy is calculated as the proportion of predicted responses from a binomial model equal to the actual responses given each data point. Simple $\mathrm{R}^{2}$ is calculated as $1-\frac{\sum_{i}\left(y_{i}-E\left(y_{i}\right)\right)^{2}}{\sum_{i}\left(y_{i}-\bar{y}\right)^{2}}$. Accuracy and simple $\mathrm{R}^{2}$ both vary from 0.0 to 1.0 , and the higher the number the better the quality of the model. What is a "good" quality of model is difficult to say, but prediction should probably not be attempted with vital rate models under $90 \%$ accuracy or simple $R^{2}$. Naturally, such values may be difficult to achieve in most analyses.

The next method of assessing quality control focuses on the IPMs, themselves. Let's take a look at a summary of the historical IPM.

```
summary(lathmat3ipm)
>
> This historical lefkoMat object contains 3 matrices.
>
> Each matrix is square with }10609\mathrm{ rows and columns, and a total of 112550881 elements.
> A total of 3106216 survival transitions were estimated, with 1035405.333 per matrix.
> A total of }60600\mathrm{ fecundity transitions were estimated, with }20200\mathrm{ per matrix.
> This lefkoMat object covers 1 population, 1 patch, and 3 time steps.
>
> The dataset contains a total of }1053\mathrm{ unique individuals and 2527 unique transitions.
>
> Vital rate modeling quality control:
>
> Survival estimated with }931\mathrm{ individuals and 2246 individual transitions.
> Observation estimated with }858\mathrm{ individuals and 2121 individual transitions.
> Primary size estimated with }845\mathrm{ individuals and 1916 individual transitions.
> Secondary size transition not estimated.
> Tertiary size transition not estimated.
> Reproduction probability not estimated.
> Fecundity estimated with }335\mathrm{ individuals and }599\mathrm{ individual transitions.
> Juvenile survival estimated with }281\mathrm{ individuals and 281 individual transitions.
> Juvenile observation estimated with 210 individuals and 210 individual transitions.
> Juvenile primary size transition not estimated.
> Juvenile secondary size transition not estimated.
> Juvenile tertiary size transition not estimated.
> Juvenile reproduction probability not estimated.
> Juvenile maturity transition probability not estimated.
>
> Survival probability sum check (each matrix represented by column in order):
> [,1] [,2] [,3]
> Min. 0.000 0.000 0.000
> 1st Qu. 0.997 1.000 1.000
> Median 1.000 1.000 1.000
> Mean 0.973 0.972 0.974
> 3rd Qu. 1.000 1.000 1.000
> Max. 1.000 1.000 1.000
```

Some of the output should be familiar, particularly the output related to the vital rate models. The key output for us to look at here is at the bottom, under Survival probability sum check. The columns in our $U$ matrices should always sum to values between 0 and 1 . The reason is that the sum of each column should equal the probability of survival from whatever stage the column is associated with in time $t$ to time $t+1$, regardless of what stage the organism is in in the latter time. In at least three circumstances, these sums may be greater than 1.0 , and the user would need to correct their IPMs in these cases to prevent odd analytical results and erroneous inferences. The first circumstance is through the incorporation via supplement tables of fixed transition probabilities or proxy values that are too high. Fixing an IPM in this case would mean reducing these fixed or proxy values in the supplement table. The second circumstance is through the use of the midpoint method in size transition probability estimation. The correct way to fix this is to use the cumulative density function (CDF) method instead of the midpoint method. Fortunately, lefko3 uses the CDF method by default. The third circumstance is through the incorporation of sizes not observed, or representing strong outlier sizes. In these cases, it is possible for at least some of the resulting probabilities to be estimated at unnatural levels. In this third case, the way to correct the problem is to remove the outlier size from the classification in the stageframe.

Finally, there is at least one more method that we can use to assess the overall quality of an IPM. That method is to assess the overall structure of the IPM. The best way to do this is to inspect the elements themselves, perhaps by opening the IPM matrix in R Studio, or exporting it to Microsoft Excel or another spreadsheet program for assessment. A more visual approach assessing just the structure itself is to use the image3() function, which provides users with a means of assessing whether the overall structure of the model "looks right". For example, here we look at an image of the $1^{\text {st }}$ matrix in our historical IPM (figure 7.5).

```
image3(lathmat3ipm, used = 1)
> [[1]]
```



Figure 7.5: Matrix image of the first historical IPM. Red area corresponds to non-zero elements

We can also focus in on just the survival or fecundity transitions, as below. Note that in an ahistorical IPM, we expect all fecundity values to be located toward the top of the matrix (figure 7.6), while in a historical MPM they will be distributed in other sections. Let's take a look at the historical case.

```
image3(lathmat2ipm, used = 1, type = "F")
> [[1]]
```



Figure 7.6: Matrix image of the first fecundity matrix in our ahistorical IPM. Red area corresponds to non-zero elements

Other approaches to quality control are provided for other aspects of matrix analysis in lefko3.
At this point, we may move ahead and analyze the IPMs in the same ways that we might analyze other kinds of MPMs.

### 7.4 Points to remember

1. IPMs assume a continuous distribution for size and integrate vital rates across this size metric typically assuming a Gaussian size distribution. While theoretically different from MPMs, ultimately IPMs are discretized in ways that allow them to be created and analyzed exactly the same way as function-based MPMs.
2. Package lefko3 allows the development of even extremely large numbers of size bins across the range of a continuous size metric, using the "ipm" shorthand in the sf_create() function. With only a few lines, a stageframe with hundreds of discretized size bins can be created and used to create an IPM for analysis.
3. Quality control tools include the linear model accuracy and pseudo- $\mathrm{R}^{2}$ output from modelsearch(), the output from summary() calls for IPMs created with flefko2() or flefko3(), and visualization with image3().

## Chapter 8

## Population Projection I: Deterministic Analysis

For a moment, nothing happened. Then, after a second or so, nothing continued to happen. - Douglas Adams, The Hitchhiker's Guide to the Galaxy

Deterministic analysis refers to matrix projection conducted assuming that a single matrix encompasses all of the important demographic variation in a population. This single matrix is projected forward an extremely large number of time steps or to a limit of infinity, and the asymptotic properties of the projection are assessed. Projecting forward by the same matrix means that instead of going one time step forward, where $\mathbf{n}_{\mathbf{t}+\mathbf{1}}=\mathbf{A} \mathbf{n}_{\mathbf{t}}$, we find ourselves analyzing the result of the following equation.

$$
\begin{equation*}
\mathbf{n}_{\mathbf{t}}=\mathbf{A}^{\mathbf{t}} \mathbf{n}_{\mathbf{0}, t \rightarrow \infty} \tag{8.1}
\end{equation*}
$$

The asymptotic properties of this projection conform to the results of eigenanalysis, where after some time we see the right eigenvector associated with the dominant eigenvalue yielding the equilibrial stage structure $\mathbf{w}$, and the left eigenvector associated with the dominant eigenvalue yielding the reproductive value vector $\mathbf{v}$. The dominant eigenvalue is $\lambda$, and is equal to the asymptotic population growth rate, and we also see the following conditions.

$$
\begin{gather*}
\mathbf{A} \mathbf{w}=\lambda \mathbf{w}  \tag{8.2}\\
\mathbf{v}^{*} \mathbf{A}=\lambda \mathbf{v}^{*} \tag{8.3}
\end{gather*}
$$

Here, $\mathbf{v}^{*}$ is the complex conjugate of the left eigenvector and $\mathbf{w}$ is the right eigenvector associated with $\lambda$ (Caswell, 2001). Once we have the projected population growth rate $\lambda$, the stable stage vector, and the reproductive value vector, we can use these terms to derive further values, such as the sensitivity or elasticity of $\lambda$ to changes in matrix elements.

Package lefko3 includes a number of functions to aid analysis once matrices are created. These may be of greater utility in some circumstances than established $R$ functions such as eigen(), because our functions are made to handle even extremely large, sparse matrices and can handle many kinds of unusual situations that might arise in analysis. Currently, we include functions to estimate elementwise arithmetic mean matrices, population growth rate, stable stage structure, reproductive value, sensitivity and elasticity, and life table response experiments (LTRE and sLTRE). We also include two basic projection functions that show the expected numbers of individuals in each stage forward in time under deterministic, cyclical, and stochastic conditions, with or without density dependence (these two functions will be covered in chapter 10 on General Projection Simulation).

In this chapter, we will utilize the raw MPMs developed in chapter 4 to illustrate the sorts of deterministic analyses that we can perform. Particularly, we will use the lefkoMat objects named cypmatrix2r, cypmatrix2rp, cypmatrix3r, and cypmatrix3rp. However, the function-based MPMs, IPMs, and age-by-stage MPMs that we have created can also be used in all cases.

Let's start off by taking a look at summaries of these four objects.

```
summary(cypmatrix2r)
>
> This ahistorical lefkoMat object contains 5 matrices.
>
> Each matrix is square with }11\mathrm{ rows and columns, and a total of 121 elements.
> A total of 120 survival transitions were estimated, with 24 per matrix.
> A total of 40 fecundity transitions were estimated, with }8\mathrm{ per matrix.
> This lefkoMat object covers 1 population, 1 patch, and 5 time steps.
>
> The dataset contains a total of 74 unique individuals and 320 unique transitions.
>
> Survival probability sum check (each matrix represented by column in order):
> [,1] [,2] [,3] [,4] [,5]
> Min. 0.000 0.050 0.050 0.000 0.050
> 1st Qu. 0.100 0.140 0.140 0.100 0.140
> Median 0.689 0.870 0.864 0.610 0.882
> Mean 0.552 0.629 0.629 0.528 0.627
> 3rd Qu. 1.000 1.000 1.000 0.960 1.000
> Max. 1.000 1.000 1.000 1.000 1.000
summary(cypmatrix2rp)
>
> This ahistorical lefkoMat object contains 15 matrices.
>
> Each matrix is square with }11\mathrm{ rows and columns, and a total of 121 elements.
> A total of 266 survival transitions were estimated, with 17.733 per matrix.
> A total of 70 fecundity transitions were estimated, with 4.667 per matrix.
> This lefkoMat object covers 1 population, 3 patches, and 5 time steps.
>
> The dataset contains a total of 74 unique individuals and 320 unique transitions.
>
> Survival probability sum check (each matrix represented by column in order):
> [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11] [,12]
> Min. 0.000 0.000 0.000 0.000 0.000 0.000 0.050 0.050 0.000 0.050 0.000 0.000
> 1st Qu. 0.075 0.025 0.075 0.025 0.075 0.075 0.140 0.140 0.100 0.140 0.100 0.100
> Median 0.180 0.100 0.180 0.100 0.180 0.180 0.909 0.778 0.686 0.857 0.750 0.575
> Mean 0.457 0.361 0.471 0.328 0.417 0.464 0.631 0.611 0.530}00.631 0.562 0.523
> 3rd Qu. 0.955 0.769 1.000 0.592 0.781 1.000 1.000 1.000 0.955 1.000 1.000 1.000
> Max. 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000
> [,13] [,14] [,15]
> Min. 0.000 0.000 0.000
> 1st Qu. 0.075 0.075 0.100
> Median 0.180 0.180 0.750
> Mean 0.432 0.450 0.562
> 3rd Qu. 0.875 1.000 1.000
> Max. 1.000 1.000 1.000
summary(cypmatrix3r)
```

```
>
> This historical lefkoMat object contains 4 matrices.
>
> Each matrix is square with 121 rows and columns, and a total of 14641 elements.
> A total of 242 survival transitions were estimated, with 60.5 per matrix.
> A total of 54 fecundity transitions were estimated, with 13.5 per matrix.
> This lefkoMat object covers 1 population, 1 patch, and 4 time steps.
>
> The dataset contains a total of 74 unique individuals and 320 unique transitions.
>
> Survival probability sum check (each matrix represented by column in order):
> [,1] [,2] [,3] [,4]
> Min. 0.000 0.000 0.000 0.000
> 1st Qu. 0.000 0.000 0.000 0.000
> Median 0.000 0.000 0.000 0.000
> Mean 0.173 0.179 0.166 0.198
> 3rd Qu. 0.100 0.100 0.100 0.100
> Max. 1.000 1.000 1.000 1.000
summary(cypmatrix3rp)
>
> This historical lefkoMat object contains 12 matrices.
>
> Each matrix is square with }121\mathrm{ rows and columns, and a total of 14641 elements.
> A total of }516\mathrm{ survival transitions were estimated, with 43 per matrix.
> A total of }70\mathrm{ fecundity transitions were estimated, with 5.833 per matrix.
> This lefkoMat object covers 1 population, 3 patches, and 4 time steps.
>
> The dataset contains a total of 74 unique individuals and 320 unique transitions.
>
> Survival probability sum check (each matrix represented by column in order):
> [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11]
> Min. 0.000 0.0000 0.0000 0.000 0.000 0.000 0.00 0.000 0.000 0.0000 0.000
> 1st Qu. 0.000 0.0000 0.0000 0.000 0.000 0.000 0.00 0.000 0.000 0.0000 0.000
>Median 0.000 0.0000 0.0000 0.000 0.000 0.000 0.00 0.000 0.000 0.0000 0.000
> Mean 0. 0.107 0.0945 0.0851 0.101 0.158 0.158 0.14 0.169 0.119 0.0851 0.119
> 3rd Qu. 0.000 0.0000 0.0000 0.000 0.100 0.100 0.05 0.100 0.000 0.0000 0.000
> Max. 1.000 1.0000 1.0000 1.000 1.000 1.000 1.00 1.000 1.000 1.0000 1.000
> [,12]
> Min. 0.000
> 1st Qu. 0.000
> Median 0.000
> Mean 0.144
> 3rd Qu. 0.000
> Max. 1.000
```

We see that these four MPMs cover three patches of one population, and that cypmatrix2r and cypmatrix 3 r do not distinguish the patches while cypmatrix2rp and cypmatrix3rp do. Objects cypmatrix $2 r$ and cypmatrix $2 r p$ are ahistorical, while cypmatrix $3 r$ and cypmatrix $3 r p$ are historical. Because these are raw MPMs and there are six total monitoring occasions covered, we see five matrices estimated in the ahistorical population-level set and four matrices estimated in the historical population-level set. The patch-level MPMs create five and four matrices per patch for the ahistorical and historical cases, respectively. There are 11 life history stages, which we see below.



Let's move on now to our first deterministic analysis: the estimation of the population growth rate $\lambda$.

### 8.1 Population growth rate

The deterministic population growth rate is estimated with function lambda3(). This function estimates the population growth rate $(\lambda)$ as the dominant eigenvalue of each matrix provided (technically, the largest real part of the estimated eigenvalues for each matrix). Where matrices are large, as in most historical or age-by-stage cases, lambda3() first converts matrices to sparse format in order to speed up estimation. This is all done under the hood, so the user need not do anything prior to analysis to prepare for this.

First, let's look at the population growth rate estimates at the whole-population level, using the ahistorical cypmatrix2r.

```
cyp2r_lam <- lambda3(cypmatrix2r)
cyp2r_lam
> pop patch year2 lambda
> 1 1 1 1 2004 1.1204918
> 2 1 1 2005 1.0906393
> 3 1 1 2006 1.1685560
>4 1 1 2007 0.9519705
> 5 1 1 1 2008 1.1092330
```

Since we have five matrices in that MPM, we see five estimates of $\lambda$. These range from 0.952 in 2007, to 1.169 in 2006. It may be interesting to compare these to the historical case, as below.

```
cyp3r_lam <- lambda3(cypmatrix3r)
cyp3r_lam
```

| $>$ | pop | patch | year2 | lambda |
| :--- | ---: | ---: | ---: | ---: |
| $>$ | 1 | 1 | 1 | 2005 |
| $>$ | 0.9794481 |  |  |  |
| $>$ | 2 | 1 | 1 | 2006 |
| $>$ | 3 | 1 | 1 | 2007 |
| $>$ | 0.0000000 |  |  |  |
| $>$ | 4 | 1 | 1 | 2008 |

We see one fewer estimate of $\lambda$ because there is one fewer matrix. Estimated $\lambda$ also appears to be somewhat lower in the historical case. Indeed, mean $\lambda \pm 1$ se for years 2005 through 2008 is $1.088 \pm$ 0.041 and $0.989 \pm 0.029$ for the ahistorical and historical cases, respectively. Generally, results like this suggest that individual history influences population dynamics via some sort of long-term cost, and so should not be ignored.

Let's move on now and look at $\lambda$ for the patch-level MPMs. We may wish to compare the ahistorical estimates to the historical estimates. Here, we will use lambda3() to develop these estimates for us, and then we will plot them (figure 8.1).

```
cyp2rp_lam <- lambda3(cypmatrix2rp)
cyp3rp_lam <- lambda3(cypmatrix3rp)
plot(lambda ~ year2, data = subset(cyp2rp_lam, patch == "A"),
    ylim = c(0.4, 1.4), type = "l", lwd = 2, bty = "n")
lines(lambda ~ year2, data = subset(cyp2rp_lam, patch == "B"), type = "l",
    lwd = 2, lty = 2)
lines(lambda ~ year2, data = subset(cyp2rp_lam, patch == "C"), type = "l",
    lwd = 2, lty = 3)
lines(lambda ~ year2, data = subset(cyp3rp_lam, patch == "A"), type = "l",
    lwd = 2, lty = 1, col = "red")
lines(lambda ~ year2, data = subset(cyp3rp_lam, patch == "B"), type = "l",
    lwd = 2, lty = 2, col = "red")
lines(lambda ~ year2, data = subset(cyp3rp_lam, patch == "C"), type = "l",
    lwd = 2, lty = 3, col = "red")
legend("bottomleft", c("A ahistorical", "B ahistorical", "C ahistorical",
            "A historical", "B historical", "C historical"), lty = c(1, 2, 3, 1, 2, 3),
    col = c("black", "black", "black", "red", "red", "red"), lwd = 2, cex = 0.7,
    bty = "n")
```

Here we see that $\lambda$ is quite different between the ahistorical and historical cases in the raw MPMs. This should give us pause and encourage us to consider using historical approaches to correct for the influence of individual history.

We may now wish to compare $\lambda$ for the mean matrices, as well. Let's do so, as below.

```
cyp2r_mean <- lmean(cypmatrix2r)
cyp2rp_mean <- lmean(cypmatrix2rp)
cyp3r_mean <- lmean(cypmatrix3r)
cyp3rp_mean <- lmean(cypmatrix3rp)
# Overall raw ahistorical lambda
lambda3(cyp2r_mean)
> pop patch lambda
>1 1 1 1.082189
# Raw ahistorical lambda by patch
lambda3(cyp2rp_mean)
```



Figure 8.1: Lambda of ahistorical vs. historical raw MPMs

```
> pop patch lambda
> 1 1 A 0.9835717
> 2 1 B 1.1051470
> 3 1 C 1.0717513
>4 1 0 1.0450267
# Overall raw historical lambda
lambda3(cyp3r_mean)
> pop patch lambda
> 1 1 1 1.011018
# Raw historical lambda by patch
lambda3(cyp3rp_mean)
> pop patch lambda
> 1 1 A 0.8510445
> 2 1 B 0.9481901
> 3 1 C 0.9671030
>4 1 0 0.8946030
```

As we saw before, our historical estimates are lower than our ahistorical estimates. We also see that $\lambda$ for the patch-weighted mean matrix (marked as patch 0 in the patch-level output) is lower than the whole-population $\lambda$ in both ahistorical and historical cases. The latter result suggests that the patches have different levels of influence on overall population dynamics because of different numbers of individuals.

### 8.2 Stable stage distribution and reproductive value

The stable stage distribution is a vector showing the expected proportion of the population composed of each stage after we have projected the population forward in time sufficiently to create asymptotic conditions. The reproductive value vector gives the expected asymptotic contribution of an individual in each stage to future generations, and so has a relationship (though a complex one) to natural selection (Caswell, 2001). In ahistorical analysis, the stable stage distribution and the reproductive value vector are estimated as the standardized right and left eigenvectors, respectively, associated with the dominant
eigenvalue of the matrix. Standardization of the stable stage distribution is handled by dividing each respective element of the right eigenvector by the sum of the elements in that eigenvector, yielding a vector that sums to 1.0. Standardization of the reproductive value vector is handled by dividing each element in the left eigenvector by the value of the first non-zero element in that eigenvector, which is also often though not always the lowest value. This estimation makes a number of key assumptions, and we urge readers to read Caswell (2001) for further details.

The methods mentioned above apply to historical matrices as well (deVries and Caswell, 2018). However, as described, these methods only provide the stable stage distribution and reproductive values of stage pairs, since each row and column in a historical matrix corresponds to a stage pair rather than an individual stage. We provide two functions, stablestage3() and repvalue3() to allow the estimation of these vectors for both ahistorical and historical MPMs. When provided with a historical MPM, these functions also estimate historically-corrected stable stage distributions and reproductive value vectors, which provide the stable stage distribution and reproductive value vector for the original stages in the associated life history model corrected for individual history. These vectors are comparable to the ahistorical stable stage distributions and reproductive value vectors. The historically-corrected stable stage proportion for stage $j$ is estimated as the sum of all stable stage proportions for stage $j$ in occasion $t$ across all stages in occasion $t-1$.

$$
\begin{equation*}
S S_{j}=\sum_{l=1}^{m} w_{j l} \tag{8.4}
\end{equation*}
$$

Here, $l$ is stage in occasion $t-1, m$ is the number of stages, $S S_{j}$ is the stable stage proportion of stage $j$, and $w_{j l}$ is the stable stage proportion of stage pair $j l$ given as the standardized right eigenvector corresponding to the dominant eigenvalue of the hMPM.

The historically-corrected reproductive value of stage $j$ is calculated as the sum of all reproductive values for stage $j$ in occasion $t$ across all stages in occasion $t-1$, weighted by the stable stage proportion of the respective stage pair divided by the stable stage proportion of stage $j$ at occasion $t$ (Ehrlén, 2000).

$$
\begin{equation*}
R V_{j}=\sum_{l=1}^{m} v_{j l}\left(w_{j l} / S S_{j}\right) \tag{8.5}
\end{equation*}
$$

Here, $R V_{j}$ refers to reproductive value of stage $j$, and $v_{j l}$ refers to the reproductive value of the stage pair as given by the standardized left eigenvector of the dominant eigenvalue of the historical MPM. Individual history can alter the stable stage distribution and reproductive value vector quite drastically from those from ahistorical analyses.

Now let's take a look at the stable stage distributions at the population level. We will start by looking just at the ahistorical equilibrium.

```
tm2ss_r <- stablestage3(cyp2r_mean)
tm2ss_r
> matrix stage_id stage ss_prop
> 1 1 1 SD 4.709506e-01
> 2 1 2 P1 4.796543e-01
> 3 1 3 P2 4.432261e-02
> 4 1 4 P3 4.095645e-03
> 5 1 5 SL 1.983961e-04
> 6 1 % 6 D 4.309986e-05
> 7 1 % 7 XSm 2.759814e-04
> 8 1 8 Sm 3.049081e-04
> 9 1 9 Md 9.705559e-05
> 10 1 10 Lg 4.938104e-05
> 11 1 11 XLg 8.078062e-06
```

The output shows us the stage information and corresponding equilibrial proportion (ss_prop) for each matrix, of which there is only one. For each matrix, the proportions should sum to 1.0. Here, it appears that $48 \%$ of the stable stage structure is $1^{\text {st }}$ year protocorms, and $47 \%$ is composed of dormant seeds, with all other stages comprising the remaining $5 \%$.

Now let's compare this output to the historical case.



| > 94 | 1 |  | 6 | 9 | D | Md $0.000000 \mathrm{e}+00$ |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| > 95 | 1 |  | 7 | 9 | XSm | Md 1.004197e-05 |
| > 96 | 1 |  | 8 | 9 | Sm | Md 2.949146e-05 |
| > 97 | 1 |  | 9 | 9 | Md | Md 6.204565e-05 |
| > 98 | 1 |  | 10 | 9 | Lg | Md 5.996768e-06 |
| > 99 | 1 |  | 11 | 9 | XLg | Md $0.000000 \mathrm{e}+00$ |
| > 100 | 1 |  | 1 | 10 | SD | Lg 3.337106e-02 |
| > 101 | 1 |  | 2 | 10 | P1 | Lg 3.337106e-02 |
| > 102 | 1 |  | 3 | 10 | P2 | Lg 0.000000e+00 |
| > 103 | 1 |  | 4 | 10 | P3 | Lg $0.000000 \mathrm{e}+00$ |
| > 104 | 1 |  | 5 | 10 | SL | Lg $0.000000 \mathrm{e}+00$ |
| > 105 | 1 |  | 6 | 10 | D | Lg $0.000000 \mathrm{e}+00$ |
| > 106 | 1 |  | 7 | 10 | XSm | Lg $0.000000 \mathrm{e}+00$ |
| > 107 | 1 |  | 8 | 10 | Sm | Lg 1.194258e-06 |
| > 108 | 1 |  | 9 | 10 | Md | Lg 2.677112e-06 |
| > 109 | 1 |  | 10 | 10 | Lg | Lg 5.257711e-06 |
| > 110 | 1 |  | 11 | 10 | XLg | Lg 6.918967e-07 |
| > 111 | 1 |  | 1 | 11 | SD | XLg 4.477104e-03 |
| > 112 | 1 |  | 2 | 11 | P1 | XLg 4.477104e-03 |
| > 113 | 1 |  | 3 | 11 | P2 | XLg $0.000000 \mathrm{e}+00$ |
| > 114 | 1 |  | 4 | 11 | P3 | XLg $0.000000 \mathrm{e}+00$ |
| > 115 | 1 |  | 5 | 11 | SL | XLg $0.000000 \mathrm{e}+00$ |
| > 116 | 1 |  | 6 | 11 | D | XLg $0.000000 \mathrm{e}+00$ |
| > 117 | 1 |  | 7 | 11 | XSm | XLg $0.000000 \mathrm{e}+00$ |
| > 118 | 1 |  | 8 | 11 | Sm | XLg $0.000000 \mathrm{e}+00$ |
| > 119 | 1 |  | 9 | 11 | Md | XLg $0.000000 \mathrm{e}+00$ |
| > 120 | 1 |  | 10 | 11 | Lg | XLg 1.692245e-07 |
| > 121 | 1 |  | 11 | 11 | XLg | XLg 3.421782e-07 |
| > |  |  |  |  |  |  |
| > \$ahist |  |  |  |  |  |  |
| $>$ |  | stage_id | stage | ss_ |  |  |
| $>1$ | 1 | 1 |  | 4.731057 |  |  |
| $>2$ | 1 | 2 | P1 | 4.740135 |  |  |
| $>3$ | 1 | 3 | P2 | 4.688477 |  |  |
| $>4$ | 1 | 4 | P3 | 4.637381 |  |  |
| $>5$ | 1 | 5 |  | 2.412744 |  |  |
| $>6$ | 1 | 6 | D | 8.679096 |  |  |
| $>7$ | 1 | 7 | XSm | 4.812309 |  |  |
| $>8$ | 1 | 8 |  | 4.179082 |  |  |
| $>9$ | 1 | 9 | Md | 1.126409 |  |  |
| > 10 | 1 | 10 |  | 1.777762 |  |  |
| > 11 | 1 | 11 | XLg | 1.034075 |  |  |

We see two data frames, the first covering the expected equilibrial distribution of stage pairs, and the second showing the corrected equilibrial stage distribution. In both cases, the proportions sum to 1.0 within each matrix, of which there is only one. The amount of information here is difficult to digest at one glance, so let's compare the ahistorical stage distribution to the historically corrected stage distribution via a bar plot (figure 8.2).

```
ss_put_together <- cbind.data.frame(tm2ss_r$ss_prop, tm3ss_r$ahist$ss_prop)
names(ss_put_together) <- c("ahistorical", "historical")
rownames(ss_put_together) <- tm2ss_r$stage
```

```
barplot(t(ss_put_together), beside=T, ylab = "Proportion", xlab = "Stage",
    ylim = c(0, 0.85), col = c("black", "orangered"), bty = "n")
legend("topright", c("ahistorical", "historical"), col = c("black","orangered"),
    pch = 15, cex = 0.9, bty = "n")
```



Figure 8.2: Ahistorical vs. historically-corrected stable stage distribution
Overall, these are very similar patterns. Particularly, both ahistorical and historical analyses suggest that dormant seeds and first year protocorms take up the greatest share of the stable stage structure, with second year protocorms coming next. Adults contribute little to the stable stage structure. However, the historical model predicts a slightly greater share of the population composed of dormant seeds and $2^{\text {nd }}$ year protocorms, and a slightly lower share composed of $1^{\text {st }}$ year protocorms.

Next let's look at the reproductive values associated with both ahistorical and historical approaches. We will initially look at just the ahistorical case.

| > | matrix | stage_id | stage | rep_value |
| :---: | :---: | :---: | :---: | :---: |
| > 1 | 1 | 1 | SD | 1.00000 |
| > 2 | 1 | 2 | P1 | 10.02189 |
| $>3$ | 1 | 3 | P2 | 108.45573 |
| $>4$ | 1 | 4 | P3 | 1173.69567 |
| $>5$ | 1 | 5 | SL | 25403. 20404 |
| $>6$ | 1 | 6 | D | 39100.90885 |
| > 7 | 1 | 7 | XSm | 37601.08685 |
| $>8$ | 1 | 8 | Sm | 55396.33397 |
| > 9 | 1 | 9 | Md | 85190.53793 |
| > 10 | 1 | 10 | Lg | 128406.01230 |
| > 11 | 1 | 11 | XLg | 179758.08291 |

We have a wide range in reproductive values among the stages, with the smallest value for dormant seeds and the largest for extra large adults. Note that the reproductive value vector is the left eigenvector divided by its first non-zero element, which here is the dormant seed stage. This strongly suggests that the largest adults have a disproportionately large impact on future generations.

Let's now look at the historical reproductive values.


| > 48 | 1 | 4 | 5 | P3 | SL $0.000000 \mathrm{e}+00$ |
| :---: | :---: | :---: | :---: | :---: | :---: |
| > 49 | 1 | 5 | 5 | SL | SL $1.924265 \mathrm{e}+04$ |
| > 50 | 1 | 6 | 5 | D | SL 1.600483e+04 |
| $>51$ | 1 | 7 | 5 | XSm | SL $2.685861 \mathrm{e}+04$ |
| > 52 | 1 | 8 | 5 | Sm | SL 4.755014e+04 |
| $>53$ | 1 | 9 | 5 | Md | SL $0.000000 \mathrm{e}+00$ |
| > 54 | 1 | 10 | 5 | Lg | SL $0.000000 \mathrm{e}+00$ |
| > 55 | 1 | 11 | 5 | XLg | SL $0.000000 \mathrm{e}+00$ |
| > 56 | 1 | 1 | 6 | SD | D $0.000000 \mathrm{e}+00$ |
| > 57 | 1 | 2 | 6 | P1 | D $0.000000 \mathrm{e}+00$ |
| > 58 | 1 | 3 | 6 | P2 | D $0.000000 \mathrm{e}+00$ |
| > 59 | 1 | 4 | 6 | P3 | D $0.000000 \mathrm{e}+00$ |
| > 60 | 1 | 5 | 6 | SL | D $0.000000 \mathrm{e}+00$ |
| > 61 | 1 | 6 | 6 | D | D $4.034282 \mathrm{e}+03$ |
| > 62 | 1 | 7 | 6 | XSm | D $1.631493 \mathrm{e}+04$ |
| > 63 | 1 | 8 | 6 | Sm | D $2.683487 \mathrm{e}+04$ |
| > 64 | 1 | 9 | 6 | Md | D $0.000000 \mathrm{e}+00$ |
| > 65 | 1 | 10 | 6 | Lg | D $3.449945 \mathrm{e}+04$ |
| > 66 | 1 | 11 | 6 | XLg | D $0.000000 \mathrm{e}+00$ |
| > 67 | 1 | 1 | 7 | SD | XSm 1.019782e+00 |
| > 68 | 1 | 2 | 7 | P1 | XSm 9.310181e+00 |
| > 69 | 1 | 3 | 7 | P2 | XSm 0.000000e+00 |
| > 70 | 1 | 4 | 7 | P3 | XSm 0.000000e+00 |
| > 71 | 1 | 5 | 7 | SL | XSm 0.000000e+00 |
| > 72 | 1 | 6 | 7 | D | XSm 1.600483e+04 |
| > 73 | 1 | 7 | 7 | XSm | XSm 2.728434e+04 |
| > 74 | 1 | 8 | 7 | Sm | XSm 5.010449e+04 |
| > 75 | 1 | 9 | 7 | Md | XSm 6.833780e+04 |
| > 76 | 1 | 10 | 7 | Lg | XSm 1.518934e+04 |
| > 77 | 1 | 11 | 7 | XLg | XSm 0.000000e+00 |
| > 78 | 1 | 1 | 8 | SD | Sm 1.019782e+00 |
| > 79 | 1 | 2 | 8 | P1 | Sm 9.310181e+00 |
| > 80 | 1 | 3 | 8 | P2 | Sm 0.000000e+00 |
| > 81 | 1 | 4 | 8 | P3 | Sm 0.000000e+00 |
| > 82 | 1 | 5 | 8 | SL | Sm 0.000000e+00 |
| > 83 | 1 | 6 | 8 | D | Sm 1.519080e+04 |
| > 84 | 1 | 7 | 8 | XSm | Sm 3.623157e+04 |
| > 85 | 1 | 8 | 8 | Sm | Sm 7.229060e+04 |
| > 86 | 1 | 9 | 8 | Md | Sm 1.037423e+05 |
| > 87 | 1 | 10 | 8 | Lg | Sm 1.421550e+04 |
| > 88 | 1 | 11 | 8 | XLg | Sm 0.000000e+00 |
| > 89 | 1 | 1 | 9 | SD | Md 1.019782e+00 |
| > 90 | 1 | 2 | 9 | P1 | Md 9.310181e+00 |
| > 91 | 1 | 3 | 9 | P2 | Md $0.000000 \mathrm{e}+00$ |
| > 92 | 1 | 4 | 9 | P3 | Md $0.000000 \mathrm{e}+00$ |
| > 93 | 1 | 5 | 9 | SL | Md $0.000000 \mathrm{e}+00$ |
| > 94 | 1 | 6 | 9 | D | Md $0.000000 \mathrm{e}+00$ |
| > 95 | 1 | 7 | 9 | XSm | Md $4.524192 \mathrm{e}+04$ |
| > 96 | 1 | 8 | 9 | Sm | Md $6.755724 \mathrm{e}+04$ |
| > 97 | 1 | 9 | 9 | Md | Md 1.381815e+05 |
| > 98 | 1 | 10 | 9 | Lg | Md 8.161812e+04 |


| > 99 | 1 |  | 11 | 9 | XLg |  | $0.000000 \mathrm{e}+00$ |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| > 100 |  |  | 1 | 10 | SD |  | $1.019782 \mathrm{e}+00$ |
| > 101 |  |  | 2 | 10 | P1 |  | $9.310181 \mathrm{e}+00$ |
| > 102 |  |  | 3 | 10 | P2 |  | $0.000000 \mathrm{e}+00$ |
| > 103 |  |  | 4 | 10 | P3 | Lg | $0.000000 \mathrm{e}+00$ |
| > 104 |  |  | 5 | 10 | SL |  | $0.000000 \mathrm{e}+00$ |
| > 105 |  |  | 6 | 10 | D | Lg | $0.000000 \mathrm{e}+00$ |
| > 106 |  |  | 7 | 10 | XSm |  | $0.000000 \mathrm{e}+00$ |
| > 107 |  |  | 8 | 10 | Sm |  | $5.355023 \mathrm{e}+04$ |
| > 108 |  |  | 9 | 10 | Md |  | $6.142678 \mathrm{e}+04$ |
| > 109 |  |  | 10 | 10 | Lg |  | $1.395183 \mathrm{e}+05$ |
| > 110 |  |  | 11 | 10 | XLg | Lg | 8.345149e+04 |
| > 111 |  |  | 1 | 11 | SD | XLg | $1.019782 \mathrm{e}+00$ |
| > 112 |  |  | 2 | 11 | P1 | XLg | $9.310181 \mathrm{e}+00$ |
| > 113 |  |  | 3 | 11 | P2 | XLg | 0.000000 +00 |
| > 114 |  |  | 4 | 11 | P3 | XLg | $0.000000 \mathrm{e}+00$ |
| > 115 |  |  | 5 | 11 | SL | XLg | $0.000000 \mathrm{e}+00$ |
| > 116 |  |  | 6 | 11 | D | XLg | $0.000000 \mathrm{e}+00$ |
| > 117 |  |  | 7 | 11 | XSm | XLg | $0.000000 \mathrm{e}+00$ |
| > 118 |  |  | 8 | 11 | Sm | XLg | $0.000000 \mathrm{e}+00$ |
| > 119 |  |  | 9 | 11 | Md | XLg | $0.000000 \mathrm{e}+00$ |
| > 120 |  |  | 10 | 11 | Lg | XLg | $6.790669 \mathrm{e}+04$ |
| > 121 |  |  | 11 | 11 | XLg | XLg | $7.189852 \mathrm{e}+04$ |
| > |  |  |  |  |  |  |  |
| > \$ahist |  |  |  |  |  |  |  |
| > |  | stage_id | stage rep_value |  |  |  |  |
| $>1$ | 1 | 1 | SD 1.000000e+00 |  |  |  |  |
| $>2$ |  | 2 | P1 9.146789e+00 |  |  |  |  |
| > 3 | 1 | 3 | P2 9.247569e+01 |  |  |  |  |
| > 4 | 1 | 4 | P3 9.349460e+02 |  |  |  |  |
| > 5 | 1 | 5 | SL $1.890495 \mathrm{e}+04$ |  |  |  |  |
| $>6$ | 1 | 6 | D $1.524115 \mathrm{e}+04$ |  |  |  |  |
| > 7 | 1 | 7 | XSm 2.843405e+04 |  |  |  |  |
| $>8$ | 1 | 8 | Sm 5.915940e+04 |  |  |  |  |
| > 9 | 1 | 9 | Md 1.175879e+05 |  |  |  |  |
| > 10 | 1 | 10 | Lg 7.540168e+04 |  |  |  |  |
| > 11 | 1 | 11 | XLg 7.823111e+04 |  |  |  |  |

As in the stable stage output, we see two data frames. The first shows the reproductive values of historical stage pairs, where the has been standardized by dividing the vector against its minimum element. Most reproductive values are zero, generally when referring to an impossible stage pairing (e.g. an individual cannot be a small adult in one year and a $3^{\text {rd }}$ year protocorm in the next). The biologically possible pairs are generally positive, and rather comparable to the ahistorical case. The second data frame shows the historically-corrected reproductive values for each stage in the stageframe. Note that individual history has reduced the reproductive values of most stages. Let's compare the ahistorical reproductive values to the historically-corrected reproductive values, in a side-by-side pair of plots (figure 8.3).

```
tm2rv_r <- repvalue3(cyp2r_mean)
tm3rv_r <- repvalue3(cyp3r_mean)
tm2rv_plot <- as.matrix(tm2rv_r$rep_value)
```

```
rownames(tm2rv_plot) <- tm2rv_r$stage
tm3rv_plot <- as.matrix(tm3rv_r$ahist$rep_value)
rownames(tm3rv_plot) <- tm3rv_r$ahist$stage
par(mfrow=c(1,2))
barplot(t(tm2rv_plot), ylab = "Rep value", xlab = "Stage", col = "black",
    ylim = c(0, 200000), bty = "n")
title("a)", adj = 0)
legend("topleft", c("ahistorical", "historical"), col = c("black", "orangered"),
    pch = 15, bty = "n")
barplot(t(tm3rv_plot), ylab = "", xlab = "Stage", col = "orangered",
    ylim = c(0, 200000), bty = "n")
title("b)", adj = 0)
```



Figure 8.3: Ahistorical (a) vs. historically-corrected (b) deterministic reproductive values

We see some big differences here. In the ahistorical case, reproductive value increases with size, reaching its peak in the largest adults. In contrast, in the historical case, the greatest reproductive values are associated with medium adults, decreasing again in large and extra large adults. So, history appears to have large effects here. Given that reproductive value has implications for natural selection, we might hypothesize that size-related traits might evolve differently under these two models. Interesting results in need of further study!

### 8.3 Sensitivity analysis

Package lefko3 contains functions allowing users to conduct deterministic and stochastic sensitivity and elasticity analyses. Sensitivity and elasticity analysis are forms of perturbation analysis, and we urge readers to consult Caswell (2001) and Caswell (2019) to become fully acquainted with the topic. Here, we discuss just the most important aspects to understand these analyses as conducted using lefko3.

The sensitivity of $\lambda$, the deterministic population growth rate, to a specific element in a projection matrix can be calculated using eigenanalysis, as below.

$$
\begin{equation*}
\frac{\partial \lambda}{\partial a_{k j}}=\frac{\bar{v}_{k} w_{j}}{\langle\mathbf{w}, \mathbf{v}\rangle} \tag{8.6}
\end{equation*}
$$

Here, $\mathbf{w}$ is the stable stage distribution vector calculated as the right eigenvector of the dominant eigenvalue of the matrix (standardized to sum to 1.0 ), $\mathbf{v}$ is the reproductive value vector calculated as the associated left eigenvector (standardized by dividing by the value of the first non-zero or minimum value), and $\bar{v}_{k}$ is the complex conjugate of element $k$ of $\mathbf{v} . k$ refers to stage in occasion $t+1$ and in the ahMPM refers to the corresponding row, while $j$ refers to stage in occasion $t$ and in an ahMPM refers to the corresponding column. The term $<\mathbf{w}, \mathbf{v}>$ refers to the scalar product of these vectors.

In the hMPM case, the basic method is essentially the same, although the equation itself changes somewhat to account for stage $l$ in occasion $t-1$. Thus, the sensitivity of $\lambda$ to each matrix element in an hMPM is given as

$$
\begin{equation*}
\frac{\partial \lambda}{\partial a_{k j l}}=\frac{\bar{v}_{k j} w_{j l}}{\langle\mathbf{w}, \mathbf{v}\rangle} \tag{8.7}
\end{equation*}
$$

Here, $k$ and $j$ are as before, $l$ is stage in occasion $t-1, k j$ refers both to the stage pair in occasions $t+1$ and $t$ as well as the corresponding row in the historical matrix, and $j l$ refers both to the stage pair in occasions $t$ and $t-1$ as well as the corresponding column in the historical matrix. Historicallycorrected sensitivities may also be estimated for the basic stages of the life history model using the historically-corrected stable stage distributions and reproductive values given in equations 8.4 and 8.5.

Let's analyze the sensitivity of $\lambda$ to the elements of our raw Cypripedium candidum MPMs. First, let's look at the raw ahistorical MPM.

```
tm2sens_r <- sensitivity3(cyp2r_mean)
tm2sens_r
> $h_sensmats
> NULL
>
> $ah_sensmats
> $ah_sensmats[[1]]
> [,1] [,2] [,3] [,4] [,5]
> [1,] 7.251380e-03 7.385393e-03 6.824496e-04 6.306198e-05 3.054770e-06
> [2,] 7.267251e-02 7.401557e-02 6.839433e-03 6.320001e-04 3.061456e-05
> [3,] 7.864537e-01 8.009882e-01 7.401557e-02 6.839433e-03 3.313073e-04
> [4,] 8.510913e+00 8.668204e+00 8.009882e-01 7.401557e-02 3.585370e-03
> [5,] 1.842083e+02 1.876126e+02 1.733641e+01 1.601976e+00 7.760094e-02
> [6,] 2.835355e+02 2.887756e+02 2.668440e+01 2.465781e+00 1.194443e-01
> [7,] 2.726598e+02 2.776988e+02 2.566085e+01 2.371199e+00 1.148627e-01
> [8,] 4.016999e+02 4.091237e+02 3.780521e+01 3.493403e+00 1.692231e-01
> [9,] 6.177489e+02 6.291656e+02 5.813825e+01 5.372284e+00 2.602375e-01
> [10,] 9.311208e+02 9.483289e+02 8.763064e+01 8.097538e+00 3.922508e-01
> [11,] 1.303494e+03 1.327584e+03 1.226758e+02 1.133590e+01 5.491196e-01
> [,6] [,7] [,8] [,9] [,10]
> [1,] 6.636226e-07 4.249376e-06 4.694769e-06 1.494397e-06 7.603360e-07
> [2,] 6.650750e-06 4.258676e-05 4.705044e-05 1.497667e-05 7.620002e-06
> [3,] 7.197367e-05 4.608692e-04 5.091746e-04 1.620759e-04 8.246280e-05
> [4,] 7.788909e-04 4.987474e-03 5.510230e-03 1.753967e-03 8.924031e-04
> [5,] 1.685814e-02 1.079478e-01 1.192622e-01 3.796246e-02 1.931497e-02
> [6,] 2.594825e-02 1.661545e-01 1.835697e-01 5.843227e-02 2.972983e-02
```

```
[7,] 2.495293e-02 1.597811e-01 1.765284e-01 5.619094e-02 2.858946e-02
    [8,] 3.676226e-02 2.353998e-01 2.600730e-01 8.278409e-02 4.211983e-02
    [9,] 5.653436e-02 3.620066e-01 3.999499e-01 1.273085e-01 6.477344e-02
[10,] 8.521313e-02 5.456454e-01 6.028365e-01 1.918895e-01 9.763172e-02
[11,] 1.192915e-01 7.638596e-01 8.439226e-01 2.686299e-01 1.366765e-01
    [,11]
    [1,] 1.243806e-07
    [2,] 1.246528e-06
[3,] 1.348978e-05
[4,] 1.459849e-04
[5,] 3.159665e-03
    [6,] 4.863393e-03
    [7,] 4.676844e-03
    [8,] 6.890227e-03
    [9,] 1.059605e-02
[10,] 1.597121e-02
[11,] 2.235841e-02
>
>
> $hstages
> NULL
>
> $ahstages
\begin{tabular}{|c|c|c|c|c|c|c|c|}
\hline > 1 & 1 & SD & 0.0 & NA & NA & 0 & NA \\
\hline \(>2\) & 2 & P1 & 0.0 & NA & NA & 0 & NA \\
\hline \(>3\) & 3 & P2 & 0.0 & NA & NA & 0 & NA \\
\hline \(>4\) & 4 & P3 & 0.0 & NA & NA & 0 & NA \\
\hline \(>5\) & 5 & SL & 0.0 & NA & NA & 0 & NA \\
\hline \(>6\) & 6 & D & 0.0 & NA & NA & 0 & NA \\
\hline \(>7\) & 7 & XSm & 1.0 & NA & NA & 0 & NA \\
\hline \(>8\) & 8 & Sm & 3.0 & NA & NA & 0 & NA \\
\hline > 9 & 9 & Md & 6.0 & NA & NA & 0 & NA \\
\hline & 10 & Lg & 11.0 & NA & NA & 0 & NA \\
\hline > 11 & 11 & XLg & 19.5 & NA & NA & 0 & NA \\
\hline
\end{tabular}
> repstatus obsstatus propstatus immstatus matstatus entrystage indataset
\begin{tabular}{llllllll}
\(>\) & 1 & 0 & 0 & 1 & 0 & 0 & 1 \\
\(>\) & 2 & 0 & 0 & 0 & 1 & 0 & 0 \\
\(>\) & 3 & 0 & 0 & 0 & 1 & 0 & 0 \\
\(\gg\) & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
\(>\) & 5 & 0 & 0 & 0 & 1 & 0 & 0 \\
\(>\) & 6 & 0 & 0 & 0 & 0 & 1 & 0 \\
\(>7\) & 1 & 1 & 0 & 0 & 1 & 0 & 0 \\
\(>8\) & 1 & 1 & 0 & 0 & 1 & 0 & 1 \\
\(>\) & 9 & 1 & 1 & 0 & 0 & 1 & 0 \\
\(>10\) & 1 & 1 & 0 & 0 & 1 & 0 & 1 \\
\(>\) & 11 & 1 & 1 & 0 & 0 & 1 & 0
\end{tabular}
> binhalfwidth_raw sizebin_min sizebin_max sizebin_center sizebin_width
> 1 0.0 0.0 0.0.0 0.0
> 2 0.0 0.0 0.0
llllll
```



The output is a list with several elements. The first element is called $h_{-}$sensmats, and it would hold the sensitivity matrices of the A matrices in the lefkoMat object input if our MPM was historical. Here, our MPM is actually ahistorical, so this element is empty. The second element, ah_sensmats, is a list of sensitivity matrices with each matrix corresponding to each matrix of our input MPM in order.

If we had input a historical MPM, then this element would show matrices of the historically-corrected sensitivities. The order of stage pairs would be shown in element h_stages if the input MPM was historical. Element ah_stages always shows the stageframe underlying the MPM.

Sensitivity matrices can be hard to read, because all of the elements are estimated to have non-zero sensitivities even if the matrix element must equal zero. It usually helps to analyze such matrices with particular questions in mind. For example, what element is $\lambda$ most sensitive to changes in? We can address this by searching for the maximum sensitivity element, as below.

```
# The highest deterministic sensitivity value
max(tm2sens_r$ah_sensmats[[1]][which(cyp2r_mean$A[[1]] > 0)])
> [1] 1.601976
# This value is associated with element
max_sens <- intersect(which(tm2sens_r$ah_sensmats[[1]] ==
    max(tm2sens_r$ah_sensmats[[1]][which(cyp2r_mean$A[[1]] > 0)])),
    which(cyp2r_mean$A[[1]] > 0))
max_sens
> [1] 38
# This element is associated with column
ceiling(max_sens / dim(tm2sens_r$ahstages)[1])
> [1] 4
# and row
max_sens %% dim(tm2sens_r$ahstages) [1]
> [1] 5
```

The highest sensitivity value appears to be associated with element 38 , which is the transition from $3^{\text {rd }}$ year protocorm to seedling. This element is one that we included as a given rate via supplement table, and its high sensitivity means that these assumed, fixed transition probabilities have a great impact on our analyses.

How does all of this compare with the historical sensitivity analysis? Let's take a look, remembering that we will now get a historical sensitivity matrix with 121 rows and columns each corresponding to stage pairs, as well as a historically-corrected sensitivity matrix with 11 rows and columns each corresponding to stages.

```
tm3sens_r <- sensitivity3(cyp3r_mean)
# Highest sensitivity in the historical matrix
max(tm3sens_r$h_sensmats[[1]][which(cyp3r_mean$A[[1]] > 0)])
> [1] 1.210279
# This value is associated with element
max_sens1 <- intersect(which(tm3sens_r$h_sensmats[[1]] ==
        max(tm3sens_r$h_sensmats[[1]][which(cyp3r_mean$A[[1]] > 0)])),
    which(cyp3r_mean$A[[1]] > 0))
max_sens1
> [1] 3063
# This element is associated with column
ceiling(max_sens1 / dim(tm3sens_r$hstages)[1])
> [1] 26
# and row
```

```
max_sens1 %% dim(tm3sens_r$hstages) [1]
> [1] 38
# Highest sensitivity in the historically-corrected matrix
max(tm3sens_r$ah_sensmats[[1]][which(cyp2r_mean$A[[1]] > 0)])
> [1] 1.210279
# This value is associated with element
max_sens2 <- intersect(which(tm3sens_r$ah_sensmats[[1]] ==
    max(tm3sens_r$ah_sensmats[[1]][which(cyp2r_mean$A[[1]] > 0)])),
    which(cyp2r_mean$A[[1]] > 0))
max_sens2
> [1] 38
# This element is associated with column
ceiling(max_sens2 / dim(tm3sens_r$ahstages)[1])
> [1] 4
# and row
max_sens2 %% dim(tm3sens_r$ahstages) [1]
> [1] 5
```

Here we find that $\lambda$ in the hMPM is most sensitive to element 3063. This corresponds to the the transition from the $26^{\text {th }}$ stage pair ( $2^{\text {nd }}$ year protocorm in time $t-1$ and $3^{\text {rd }}$ year protocorm in time $t$ ), to the $38^{\text {th }}$ stage pair ( $3^{\text {rd }}$ year protocorm in time $t$ and seedling in time $t+1$ ). Further, the historically-corrected highest sensitivity is the same transition as in the ahistorical case. So, once again we find that we find here that early-life transitions have strong additive influences on $\lambda$.

Incidentally, the user may wish to know how we can easily understand which a transition each element corresponds to. For example, how do we know that element 3063 corresponds to the transition from the $26^{\text {th }}$ stage pair to the $38^{\text {th }}$ stage pair? The element index 3063 assumes a column vectorized matrix, meaning that we can think of the matrix as a column vector where the left-most column is stacked on top of the second column, and this combined vector is stacked on top of the third column, etc. In the historical matrix, there are 121 rows and columns, so to determine which column we are in, we need to divide the element index by the number of columns and round up to the nearest integer, as below.

```
ceiling(3063 / 121)
> [1] 26
```

To determine the row, we divide the matrix element index by the number of columns and take the remainder.

```
3063 %% 121
```

> [1] 38

### 8.4 Elasticity analysis

Sensitivities show how much a small, additive change in a matrix element can alter $\lambda$. They do so by estimating the local slope of $\lambda$ given in terms of the matrix element (Caswell, 2001). Problematically, sensitivities are also typically non-zero even for impossible transitions represented by zeros in the matrix. Elasticities, in contrast, show the influence of a small, proportional change in a matrix element
on $\lambda$, essentially given as the local slope of $\log \lambda$ given in terms of the $\log$ of the matrix element. This results in impossible transitions yielding elasticity values of zero, although it also yields somewhat different inferences from sensitivity analysis.

In the ahistorical case, the elasticity, $e_{k j}$, of $\lambda$ to change in a matrix element $a_{k j}$ is given as the following.

$$
\begin{equation*}
e_{k j}=\frac{a_{k j}}{\lambda} \frac{\partial \lambda}{\partial a_{k j}} \tag{8.8}
\end{equation*}
$$

The historical case is just an extension of the above to include stage $l$ in occasion $t-1$.

$$
\begin{equation*}
e_{k j l}=\frac{a_{k j l}}{\lambda} \frac{\partial \lambda}{\partial a_{k j l}} \tag{8.9}
\end{equation*}
$$

Here, we use row and column definitions equivalent to those used in the historical sensitivity case. This makes the elasticity a function of the sensitivity of $\lambda$ to the matrix element, as well as of the value of the matrix element itself. Thus, structural zeros in the hMPM must have elasticity equal to 0 , although it is entirely possible that they have non-zero sensitivities.

Elasticities calculated for hMPMs can be compared to those calculated in ahMPMs easily because elasticities may be added by stage or transition type, with all elasticities corresponding to a specific matrix necessarily summing to 1.0 . We can calculate the stage pair elasticities resulting from a historical MPM analysis in the following manner, with all symbol definitions as before.

$$
\begin{equation*}
e_{k j}=\sum_{i=1}^{m} e_{k j i} \tag{8.10}
\end{equation*}
$$

These stage pair elasticity values are historically-corrected, and so can be added together in matrices of equivalent dimension to the ahistorical matrices and compared against the latter. We have provided a summary () function for elasticity output in lefko3 that automatically groups the resulting elasticity values by the kind of ahistorical or historical transition.

Let's now assess the elasticity of $\lambda$ to matrix elements in our MPMs, comparing the ahistorical to the historically-corrected case. We'll start off by just looking at the ahistorical case.

```
tm2elas_r <- elasticity3(cyp2r_mean)
tm2elas_r
> $h_elasmats
> NULL
>
> $ah_elasmats
> $ah_elasmats[[1]]
> [,1] [,2] [,3] [,4] [,5] [,6]
> [1,] 0.0005360529 0.00000000 0.00000000 0.00000000 0.000000000 0.000000000
> [2,] 0.0067153268 0.00000000 0.00000000 0.00000000 0.000000000 0.000000000
> [3,] 0.0000000000 0.07401557 0.00000000 0.00000000 0.000000000 0.000000000
> [4,] 0.0000000000 0.00000000 0.07401557 0.00000000 0.000000000 0.000000000
> [5,] 0.0000000000 0.00000000 0.00000000 0.07401557 0.003585370 0.000000000
> [6,] 0.0000000000 0.00000000 0.00000000 0.00000000 0.005105945 0.001598504
> [7,] 0.0000000000 0.00000000 0.00000000 0.00000000 0.038053219 0.007493797
> [8,] 0.0000000000 0.00000000 0.00000000 0.00000000 0.030856409 0.011606513
> [9,] 0.0000000000 0.00000000 0.00000000 0.00000000 0.000000000 0.000000000
> [10,] 0.0000000000 0.00000000 0.00000000 0.00000000 0.000000000 0.005249431
> [11,] 0.0000000000 0.00000000 0.00000000 0.00000000 0.000000000 0.000000000
> [,7] [,8] [,9] [,10] [,11]
> [1,] 0.000174604 0.002179009 0.002174921 0.001497188 0.0006896056
```

```
[2,] 0.001749861 0.021837778 0.021796810 0.015004649 0.0069111490
[3,] 0.000000000 0.000000000 0.000000000 0.000000000 0.0000000000
[4,] 0.000000000 0.000000000 0.000000000 0.000000000 0.0000000000
[5,] 0.000000000 0.000000000 0.000000000 0.000000000 0.0000000000
> [6,] 0.010146698 0.009097098 0.000000000 0.000000000 0.0000000000
> [7,] 0.075620578 0.033709675 0.004903878 0.000000000 0.0000000000
> [8,] 0.061318847 0.126000606 0.025286479 0.005004124 0.0000000000
> [9,] 0.006386165 0.052442610 0.056793876 0.011685801 0.0000000000
> [10,] 0.004384395 0.014806203 0.016352487 0.052411728 0.0044274750
> [11,] 0.000000000 0.000000000 0.000000000 0.012028230 0.0103301811
>
>
> $hstages
NULL
>
$agestages
X1
1 NA
$ahstages
\begin{tabular}{|c|c|c|c|c|c|c|c|}
\hline > 1 & 1 & SD & 0.0 & NA & NA & 0 & NA \\
\hline \(>2\) & 2 & P1 & 0.0 & NA & NA & 0 & NA \\
\hline \(>3\) & 3 & P2 & 0.0 & NA & NA & 0 & NA \\
\hline \(>4\) & 4 & P3 & 0.0 & NA & NA & 0 & NA \\
\hline \(>5\) & 5 & SL & 0.0 & NA & NA & 0 & NA \\
\hline \(>6\) & 6 & D & 0.0 & NA & NA & 0 & NA \\
\hline \(>7\) & 7 & XSm & 1.0 & NA & NA & 0 & NA \\
\hline \(>8\) & 8 & Sm & 3.0 & NA & NA & 0 & NA \\
\hline > 9 & 9 & Md & 6.0 & NA & NA & 0 & NA \\
\hline > 10 & 10 & Lg & 11.0 & NA & NA & 0 & NA \\
\hline > 11 & 11 & XLg & 19.5 & NA & NA & 0 & NA \\
\hline
\end{tabular}
repstatus obsstatus propstatus immstatus matstatus entrystage indataset
> 1 0
> 2 0 0 0 0
> 3 0 0
>4 0
>5 0
>6 0
>7 1 1 1 0
> 8 1 1 1 0
> 9 <rllllll
> 10 >
> > 11 binhalfwidth_raw sizebin_min sizebin_max sizebin_center sizebin_width
> 1 0.0 0.0 0.0 0.0
> 2 0.0 0.0 0.0
> 3 0.0 0.0 0.0 0
> 4 0.0 0.0 0.0
> 5 0.0 0.0 0.0
>6 0.5 0.0.5 0.5 0.5 0.0
```



The output is a list with a similar order to the output of the sensitivity3() function, although here we see elasticity values instead of sensitivity values. The elasticity matrix shown in element ah_elasmats has many zeros, and these zeros correspond to zero elements in the original matrix. As before, we may wish to know which element $\lambda$ is most elastic to.

```
max_elas <- which(tm2elas_r$ah_elasmats[[1]] == max(tm2elas_r$ah_elasmats[[1]]))
# Highest ahistorical elasticity is
tm2elas_r$ah_elasmats[[1]][max_elas]
> [1] 0.1260006
# This value is associated with element
max_elas
> [1] }8
# This element is associated with column
ceiling(max_elas / dim(tm2elas_r$ahstages)[1])
> [1] 8
# and row
max_elas %% dim(tm2elas_r$ahstages) [1]
> [1] 8
```

Interestingly, our ahistorical elasticity analysis suggests that $\lambda$ is most elastic to element 85 , which refers to stasis as a small adult. So, we see greater importance of adult stages in our elasticity analysis, rather than juvenile stages as we say with sensitivity analysis. But let's see how far this pattern holds.

Let's now compare to the historical case. First, we will conduct the elasticity analysis. As the output is far too large for the printed page, we have hashtagged the output, but feel free to remove hashtag in the second line and see what the output looks like.

```
tm3elas_r <- elasticity3(cyp3r_mean)
#tm3elas_r
```

The output is a good deal longer in this case, because the historical MPM yields both a historical elasticity matrix and a historically-corrected matrix. The output must also hold the index to historical stage pairs and the original stageframe. Let's take a look at which elements exert the strongest proportional influence on $\lambda$.

```
max_elas1 <- which(tm3elas_r$h_elasmats[[1]] == max(tm3elas_r$h_elasmats[[1]]))
max_elas2 <- which(tm3elas_r$ah_elasmats[[1]] == max(tm3elas_r$ah_elasmats[[1]]))
# Highest historical elasticity is
tm3elas_r$h_elasmats[[1]][max_elas1]
> [1] 0.1238486
#This value is associated with element
max_elas1
> [1] 10249
# This element is associated with column
ceiling(max_elas1 / dim(tm3elas_r$hstages)[1])
> [1] }8
# and row
max_elas1 %% dim(tm3elas_r$hstages) [1]
> [1] }8
# Highest historically-corrected elasticity is
tm3elas_r$ah_elasmats[[1]][max_elas2]
> [1] 0.1968263
```

```
# This value is associated with element
max_elas2
> [1] }8
# This element is associated with column
ceiling(max_elas2 / dim(tm3elas_r$ahstages)[1])
> [1] 8
# and row
max_elas2 %% dim(tm3elas_r$ahstages) [1]
> [1] 8
```

Here we see that ahistorical and historical analyses generally agree about which transitions $\lambda$ is most elastic in response to: stasis as a small adult. This is a very different result from the ahistorical sensitivity analysis, and is worth exploring.

Now let's compare the elasticity of population growth rate in relation to the core life history stages, via a bar plot comparison (figure 8.4).

```
elas_put_together <- cbind.data.frame(colSums(tm2elas_r$ah_elasmats[[1]]),
    colSums(tm3elas_r$ah_elasmats[[1]]))
names(elas_put_together) <- c("ahistorical", "historical")
rownames(elas_put_together) <- tm2elas_r$ahstages$stage
barplot(t(elas_put_together), beside=T, ylab = "Elasticity", xlab = "Stage",
    col = c("black", "orangered"), bty = "n")
legend("topleft", c("ahistorical", "historical"),
    col = c("black", "orangered"), pch = 15, bty = "n")
```



Figure 8.4: Ahistorical vs. historically-corrected deterministic elasticity to stage
Elasticity patterns in the two analyses look somewhat different. Both ahistorical and historical analyses show that $\lambda$ is most elastic to shifts in transitions from the small adult stage, followed by the extra small and medium adult stages. However, historical analysis suggests a far greater importance to the small adult stage, and generally lower importance to juvenile stages and to large and extra large adults. So, once again, we see an important impact of individual history here.

Finally, let's look at the elasticity sums of different transition types (figure 8.5).

```
tm2elas_r_sums <- summary(tm2elas_r)
tm3elas_r_sums <- summary(tm3elas_r)
elas_sums_together <- cbind.data.frame(tm2elas_r_sums$ahist[,2],
    tm3elas_r_sums$ahist[,2])
names(elas_sums_together) <- c("ahistorical", "historical")
rownames(elas_sums_together) <- tm2elas_r_sums$ahist$category
barplot(t(elas_sums_together), beside = T, ylab = "Elasticity",
    xlab = "Transition", col = c("black", "orangered"), bty = "n")
legend("topright", c("ahistorical", "historical"),
    col = c("black", "orangered"), pch = 15, bty = "n")
```



Figure 8.5: Ahistorical vs. historically-corrected elasticity to transitions

Fecundity has the least impact in both cases, although it does influence ahistorical analysis a reasonable amount. Stasis, in contrast, exerts the greatest influence, while growth is more influential than shrinkage and fecundity. We can also look at historical transitions, as below (figure 8.6).

```
elas_hist2plot <- as.matrix(tm3elas_r_sums$hist[,2])
rownames(elas_hist2plot) <- tm3elas_r_sums$hist$category
par(mar = c(7, 4, 2, 2) + 0.2)
barplot(t(elas_hist2plot), ylab = "Elasticity", xlab = "", xaxt = "n",
    col = "orangered", bty = "n")
text(cex=0.6, x=seq(from = -0.3, to = 1.15*length(tm3elas_r_sums$hist$category),
    by = 1.17), y=-0.08, tm3elas_r_sums$hist$category, xpd=TRUE, srt=45)
```

We can see that full stasis (i.e. staying in the same stage from occasion $t-1$ to $t+1$ ) is associated with the greatest summed elasticity values, and growth from occasion $t-1$ to $t$ followed by stasis to occasion $t+1$ is the next most important. Transitions associated with fecundity are associated with the lowest summed elasticity values, except for fecundity followed by growth.


Figure 8.6: Elasticity of lambda to historical transitions

### 8.5 Further analyses

Deterministic life table response experiments (LTREs) are also possible in lefko3, and we describe them in a later chapter 11.

### 8.6 Points to remember

1. Deterministic analysis refers to matrix projection in which a single matrix is projected forward indefinitely. Its asymptotic properties yield values such as the stable stage structure and reproductive values of each stage, as well as the population growth rate under the stable structure.
2. Package lefko3 can conduct all major deterministic analyses even for large numbers of massive matrices, such as historical IPMs and age-by-stage MPMs. This includes even sensitivity and elasticity analyses.
3. A simple means of assessing the influence of history on population dynamics is through the comparison of ahistorical projection analyses to the equivalent historically-corrected analyses, since the latter interprets the results of historical analyses but framed within the stageframe and matrix dimensions of the former.

## Chapter 9

## Population Projection II: Temporal Environmental Stochasticity

Each of us a cell of awareness imperfect and incomplete<br>Genetic blends with uncertain ends<br>on a fortune hunt that's far too fleet

- Rush, Freewill (1980)

Stochasticity refers to randomness. In population ecology, stochasticity is studied in two dominant forms: environmental and demographic (Caswell, 2001). The former refers to temporal randomness in the environment leading to random changes in the projection matrix over time via changes affecting the entire population. This randomness can be temporal or spatial, or both. The latter refers to random divergence in vital rates caused by the differing fates of individuals within populations of finite size. In this chapter, we will concern ourselves only with the former kind of stochasticity.

Most of the literature on environmental stochasticity in matrix projection analysis is focused on randomness with respect to time. Temporal randomness implies that the choice of matrix for a particular point in time, or the changes that occur in the matrix in that time, do not depend on any other time. Thus, the matrix used for projection at some point in time is drawn at random from a distribution of such matrices, which are all independent and identically distributed (Caswell, 2001). Alternatively, the kernel used to populate the matrices has a set of terms that vary across time, and these terms are shuffled randomly at each time step. Thus, the choice of matrix is not predictable knowing the composition of the matrix at one point in time tells you nothing of its composition at any other point in time.

How exactly does such randomness change the projection? In comparison to equation 8.1 , which shows a deterministic projection, an environmental stochastic projection is given as the following.

$$
\begin{equation*}
\mathbf{n}_{\mathbf{t}}=\mathbf{A}_{\mathbf{t}} \mathbf{A}_{\mathbf{t}-\mathbf{1}} \cdots \mathbf{A}_{\mathbf{1}} \mathbf{n}_{\mathbf{0}} \tag{9.1}
\end{equation*}
$$

Here, each $\mathbf{A}_{\mathbf{i}}$ is a randomly sampled matrix, each of which has the same dimensions. If we project the population forward a large number of time steps, then we generally find certain stable conditions forming in the long run. These stable conditions are predicted by the strong and weak ergodic theorems (Caswell, 2001).

Package lefko3 includes a number of functions to facilitate matrix projection analysis assuming temporal stochasticity (we shall refer to this in general as stochastic analysis from now on). Currently, we include functions to estimate stochastic population growth rate, long-run stage structure, long-run reproductive value, stochastic sensitivity and elasticity, and stochastic life table response experiments. Two projection functions are also included that can be used to assess more interesting questions.

Let's utilize the raw MPMs developed in chapter 4 to illustrate the sorts of stochastic analyses that we can perform. These are the same matrices utilized in the last chapter (chapter 8). Particularly, we will use the lefkoMat objects named cypmatrix $2 r$, cypmatrix2rp, cypmatrix $3 r$, and cypmatrix $3 r p$. However, the function-based MPMs, IPMs, and age-by-stage MPMs that we have created can also be used in all cases.

Let's start off by taking a look at summaries of these objects.

```
summary(cypmatrix2r)
>
> This ahistorical lefkoMat object contains 5 matrices.
>
> Each matrix is square with }11\mathrm{ rows and columns, and a total of 121 elements.
> A total of 120 survival transitions were estimated, with 24 per matrix.
> A total of 40 fecundity transitions were estimated, with 8 per matrix.
> This lefkoMat object covers 1 population, 1 patch, and 5 time steps.
>
> The dataset contains a total of 74 unique individuals and 320 unique transitions.
>
> Survival probability sum check (each matrix represented by column in order):
> [,1] [,2] [,3] [,4] [,5]
> Min. 0.000 0.050 0.050 0.000 0.050
> 1st Qu. 0.100 0.140 0.140 0.100 0.140
> Median 0.689 0.870 0.864 0.610 0.882
Mean 0.552 0.629 0.629 0.528 0.627
3rd Qu. 1.000 1.000 1.000 0.960 1.000
Max. 1.000 1.000 1.000 1.000 1.000
summary(cypmatrix2rp)
>
> This ahistorical lefkoMat object contains 15 matrices.
>
Each matrix is square with }11\mathrm{ rows and columns, and a total of 121 elements.
A total of 266 survival transitions were estimated, with 17.733 per matrix.
A total of 70 fecundity transitions were estimated, with 4.667 per matrix.
This lefkoMat object covers 1 population, 3 patches, and 5 time steps.
The dataset contains a total of }74\mathrm{ unique individuals and 320 unique transitions.
Survival probability sum check (each matrix represented by column in order):
> [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11] [,12]
>Min. 0.000 0.000 0.000 0.000 0.000 0.000 0.050 0.050 0.000 0.050 0.000 0.000
> 1st Qu. 0.075 0.025 0.075 0.025 0.075 0.075 0.140 0.140 0.100 0.140 0.100 0.100
> Median 0.180 0.100 0.180 0.100 0.180 0.180 0.909 0.778 0.686 0.857 0.750 0.575
> Mean 0.457 0.361 0.471 0.328 0.417 0.464 0.631 0.611 0.530}00.631 0.562 0.523
3rd Qu. 0.955 0.769 1.000 0.592 0.781 1.000 1.000 1.000 0.955 1.000 1.000 1.000
Max. 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000
    [,13] [,14] [,15]
Min. 0.000 0.000 0.000
1st Qu. 0.075 0.075 0.100
Median 0.180 0.180 0.750
Mean 0.432 0.450 0.562
3rd Qu. 0.875 1.000 1.000
Max. 1.000 1.000 1.000
```

```
summary(cypmatrix3r)
>
> This historical lefkoMat object contains 4 matrices.
>
> Each matrix is square with }121\mathrm{ rows and columns, and a total of 14641 elements.
> A total of 242 survival transitions were estimated, with 60.5 per matrix.
> A total of 54 fecundity transitions were estimated, with 13.5 per matrix.
> This lefkoMat object covers 1 population, 1 patch, and 4 time steps.
>
> The dataset contains a total of 74 unique individuals and 320 unique transitions.
>
> Survival probability sum check (each matrix represented by column in order):
> [,1] [,2] [,3] [,4]
> Min. 0.000 0.000 0.000 0.000
> 1st Qu. 0.000 0.000 0.000 0.000
> Median 0.000 0.000 0.000 0.000
> Mean 0.173 0.179 0.166 0.198
> 3rd Qu. 0.100 0.100 0.100 0.100
> Max. 1.000 1.000 1.000 1.000
summary(cypmatrix3rp)
>
> This historical lefkoMat object contains 12 matrices.
>
> Each matrix is square with }121\mathrm{ rows and columns, and a total of 14641 elements.
> A total of 516 survival transitions were estimated, with 43 per matrix.
> A total of 70 fecundity transitions were estimated, with 5.833 per matrix.
> This lefkoMat object covers 1 population, 3 patches, and 4 time steps.
>
> The dataset contains a total of 74 unique individuals and 320 unique transitions.
>
> Survival probability sum check (each matrix represented by column in order):
> [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11]
> Min. 0.000 0.0000 0.0000 0.000 0.000 0.000 0.00 0.000 0.000 0.0000 0.000
> 1st Qu. 0.000 0.0000 0.0000 0.000 0.000 0.000 0.00 0.000 0.000 0.0000 0.000
> Median 0.000 0.0000 0.0000 0.000 0.000 0.000 0.00 0.000 0.000 0.0000 0.000
> Mean 0.107 0.0945 0.0851 0.101 0.158 0.158 0.14 0.169 0.119 0.0851 0.119
> 3rd Qu. 0.000 0.0000 0.0000 0.000 0.100 0.100 0.05 0.100 0.000 0.0000 0.000
>Max. 1.000 1.0000 1.0000 1.000 1.000 1.000 1.00 1.000 1.000 1.0000 1.000
> [,12]
> Min. 0.000
> 1st Qu. 0.000
> Median 0.000
> Mean 0.144
> 3rd Qu. 0.000
> Max. 1.000
```

We see that these four MPMs cover three patches of one population, and that cypmatrix2r and cypmatrix $3 r$ do not distinguish the patches while cypmatrix $2 r p$ and cypmatrix $3 r p$ do. Objects cypmatrix2r and cypmatrix2rp are ahistorical, while cypmatrix3r and cypmatrix3rp are historical. Because these are raw MPMs and there are six total monitoring occasions covered, we see five matrices estimated in the ahistorical population-level set and four matrices estimated in the historical population-level set. The patch-level MPMs create five and four matrices per patch for the ahistorical
and historical cases, respectively. Finally, there are 11 life history stages, which we see below.


| > 1 |  | NA | A | NA |  | NA |  | NA | NA |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| $>2$ |  | NA | A | NA |  | NA |  | NA | NA |
| $>3$ |  | NA | A | NA |  | NA |  | NA | NA |
| $>4$ |  | NA | A | NA |  | NA |  | NA | NA |
| > 5 |  | NA | A | NA |  | NA |  | NA | NA |
| $>6$ |  | NA | A | NA |  | NA |  | NA | NA |
| > 7 |  | NA | A | NA |  | NA |  | NA | NA |
| $>8$ |  | NA | A | NA |  | NA |  | NA | NA |
| > 9 |  | NA | A | NA |  | NA |  | NA | NA |
| > 10 |  | NA | A | NA |  | NA |  | NA | NA |
| > 11 |  | NA | A | NA |  | NA |  | NA | NA |
| $>$ | group |  |  |  | mments | ve |  |  |  |
| $>1$ | 0 |  |  | Dormant | t seed | 1 | 0 |  |  |
| $>2$ | 0 |  | 1st | yr prot | tocorm | 1 | 0 |  |  |
| > 3 | 0 |  | 2nd | yr prot | tocorm | 1 | 0 |  |  |
| $>4$ | 0 |  | 3 rd | yr prot | tocorm | 1 | 0 |  |  |
| > 5 | 0 |  |  |  | edling | 1 | 0 |  |  |
| $>6$ | 0 |  |  | Dormant | adult | 1 | 0 |  |  |
| > 7 | 0 | Extra sma | mall adu | ult (1 sho | shoot) | 1 | 0 |  |  |
| $>8$ | 0 | Small | 1 adult | (2-4 sh | shoots) | 1 | 0 |  |  |
| > 9 | 0 | Medium | m adult | (5-7 sh | shoots) | 1 | 0 |  |  |
| > 10 | 0 | Large | adult | (8-14 sh | shoots) | 1 | 0 |  |  |
| > 11 | 0 | Extra large | e adult | (>14 sh | shoots) | 1 | 0 |  |  |

Let's move on now to the first of the stochastic analyses: the estimation of the stochastic population growth rate, $a$.

### 9.1 Stochastic population growth rate

Function slambda3() estimates the log stochastic population growth rate in its instantaneous form $\left(a=\log \lambda_{S}\right)$. This is estimated as the mean $\log$ discrete population growth rate across a user-specified number of random draws of the annual matrices provided as input in the function.

$$
\begin{equation*}
a=\log \lambda_{S}=\frac{1}{T} \sum_{t=1}^{T} \log \frac{N_{t}}{N_{t-1}} \tag{9.2}
\end{equation*}
$$

Here, $N_{t}$ is the population size in occasion $t$, and $T$ is the number of occasions projected forward, set to 10,000 by default. Function slambda3() does not shuffle across patches or populations, instead shuffling within patches, or shuffling annual matrices calculated as element-wise means of patch matrices within the same population and the same time. The methodology is based on Morris and Doak (2002), though accounts for spatial averaging of patches and can easily handle large and sparse matrices.

Let's estimate the stochastic population growth rate. We will use set. seed() prior to each run to make sure that we get the same result as what you will see (function set.seed() allows us to set the seed for the pseudo-random number generator in this function, thus yielding the same "random" sequence of matrix draws).

```
# Raw ahistorical stochastic log lambda
set.seed(42)
cypmatrix2rp_slam <- slambda3(cypmatrix2rp)
cypmatrix2rp_slam
> pop patch a var sd se
```

```
llllllllll
llllllllll
llllllllllllllllllll
llllllllll
# Raw historical stochastic log lambda
set.seed(42)
cypmatrix3rp_slam <- slambda3(cypmatrix3rp)
cypmatrix3rp_slam
> pop patch a var se
llllllllll
l 1 1 B -0.07213781 2.4096979 1.5523202 0.015523202
llllllllll
>4 1 0
```

The output above is for the ahistorical and historical patch-level MPMs. The final entry in each case, labeled as patch 0 , corresponds to the population (where the population MPM is one in which the matrix for each year is an equally weighted mean of each patch's matrix for that year). Unlike in deterministic analysis, population growth rate is not estimated for each matrix, but rather for each patch or population. This is because the growth rate is estimated by a random shuffle of the matrices, with the stochastic growth rate equal to the average log change in projected population size in the latter portion of the shuffled set. Since we did not change any of the defaults, R shuffled the matrices 10,000 times, and the resulting mean log stochastic growth rate and its associated variance, standard deviation, and standard error are calculated over these 10,000 times in the projection.

Users will notice differences between ahistorical and historical $\log \lambda_{S}$ values. Ahistorical population growth rate estimates are often higher than those for historical MPMs. The reasons are not yet clear, but they may have to do with the life histories that have been used in analysis so far - population growth in long-lived herbaceous perennial plants is typically driven by survival transitions in adults, and the impact of fecundity on growth rate appears even lower when individual history is incorporated into analysis. This may also be due to long-term trade-offs captured within the historical models but not within ahistorical models. Of note is the growth trade-off - we have found that large individuals growing to large size in time $t$ from small size in time $t$ - 1 have lower survival probability to time $t+1$ (Shefferson et al., 2014). Finally, the data are cut more finely to determine elements in the historical case, leading to a greater influence of artificial zeros (some zeros are artificial in the sense that they are zero simply because the dataset is too small to yield individuals moving through particular transitions).

It may be interesting in this case to compare our estimates of the population growth rate to the deterministic case. To compare them, we must first estimate the deterministic population growth rate $\lambda$ for the temporal mean matrices. Then, we need either to take the natural logarithm of $\lambda$, or to raise the natural number $e$ to the power of $a$ given in the slambda3() output. Let's do the latter. We will put our results together into a data frame using the cbind.data.frame() function, as below.

```
cyp2rp_mean <- lmean(cypmatrix2rp)
cyp3rp_mean <- lmean(cypmatrix3rp)
cyp2rp_mean_lam <- lambda3(cyp2rp_mean)
cyp3rp_mean_lam <- lambda3(cyp3rp_mean)
cypmatrix2rp_slam$expa <- exp(cypmatrix2rp_slam$a)
cypmatrix3rp_slam$expa <- exp(cypmatrix3rp_slam$a)
lambda_comparison <- cbind.data.frame(pop = cyp2rp_mean_lam$pop,
    patch = cyp2rp_mean_lam$patch, lambda_ah = cyp2rp_mean_lam$lambda,
    lambda_h = cyp3rp_mean_lam$lambda, expa_ah = cypmatrix2rp_slam$expa,
    expa_h = cypmatrix3rp_slam$expa)
```

```
lambda_comparison
> pop patch lambda_ah lambda_h expa_ah expa_h
> 1 1 A 0.9835717 0.8510445 0.9670638 0.8311838
> 2 1 B 1.1051470 0.9481901 1.1021705 0.9304027
> 3 1 C C 1.0717513 0.9671030 1.0686343 0.9459268
>4 1 0 1.0450267 0.8946030 1.0431169 0.8820991
```

Temporal variation in fitness leads to lower long-term fitness, and we see this in the general pattern of lower stochastic population growth rate than deterministic growth rate. There also appears to be a general trend of historical matrices yielding lower deterministic and stochastic population growth rates.

### 9.2 Long-run average stage distribution and reproductive value

Package lefko3 also handles the estimation of stochastic long-run stage distribution and long-run reproductive value vectors, which are the stochastic equivalents of the deterministic stable stage equilibrium and asymptotic reproductive value vector. It handles this through the projection of shuffled temporal matrices, typically 10,000 occasions forward though the exact number can be set as needed. According to the stochastic strong and weak ergodic theorems, this random shuffling should eventually yield a roughly stationary distribution of stage proportions and stage reproductive values. Thus, the stochastic stable structure is estimated as the arithmetic mean vector of the predicted stage proportion vectors across the final portion of such a projection (in a 10,000 time step projection, this is typically the final 1000 or so time steps).

The stochastic long-run reproductive value is more complicated. Imagine a vector, $\mathbf{x}(t)$, that includes the expected number of offspring produced by each stage in occasion $t+1$. This is referred to as the undiscounted reproductive value vector because it is not standardized (Caswell, 2001), and so is likely to eventually move to an extreme value such as infinity. Vector $\mathbf{x}(t)$ is related to other occasions in this projection, as in the following.

$$
\begin{equation*}
\mathbf{x}^{\top}(t)=\mathbf{x}^{\top}(t+1) \mathbf{A}_{t} \tag{9.3}
\end{equation*}
$$

Standardizing leads to the discounted reproductive value vector, below.

$$
\begin{equation*}
\mathbf{v}(t)=\frac{\mathbf{x}(t)}{\|\mathbf{x}(t)\|} \tag{9.4}
\end{equation*}
$$

This reproductive value vector can then be projected as well, as follows.

$$
\begin{equation*}
\mathbf{v}(t)=\frac{\mathbf{v}^{\top}(t+1) \mathbf{A}_{t}}{\left\|\mathbf{v}^{\top}(t+1) \mathbf{A}_{t}\right\|} \tag{9.5}
\end{equation*}
$$

The reproductive value vector is projected backwards in time. Under the default 10,000 time steps, package lefko3 takes its average in the final 1000 steps of the backward projection (smaller numbers of steps are used in cases where the projection is limited to less than 2000 steps).

Now let's take a look at the long-run stage distributions at the population level, comparing the deterministic to the stochastic, and the ahistorical to the historical. We will focus only on the populationlevel MPMs (note that we need to create the mean matrices for the deterministic analysis of the population-level MPMs, and do so within the first two lines). We will first create the stable stage vectors, and then put them all into a common data frame, as below.

```
cyp2r_mean <- lmean(cypmatrix2r)
cyp3r_mean <- lmean(cypmatrix3r)
tm2ss_r <- stablestage3(cyp2r_mean)
tm3ss_r <- stablestage3(cyp3r_mean)
tm2ss_rs <- stablestage3(cypmatrix2r, stochastic = TRUE, seed = 42)
tm3ss_rs <- stablestage3(cypmatrix3r, stochastic = TRUE, seed = 42)
ss_put_together <- cbind.data.frame(tm2ss_r$ss_prop, tm3ss_r$ahist$ss_prop,
    tm2ss_rs$ss_prop, tm3ss_rs$ahist$ss_prop)
names(ss_put_together) <- c("det ahist", "det hist", "sto ahist", "sto hist")
rownames(ss_put_together) <- tm2ss_r$stage
ss_put_together
> det ahist det hist sto ahist sto hist
> SD 4.709506e-01 4.731057e-01 4.568960e-01 4.565872e-01
> P1 4.796543e-01 4.740135e-01 4.696325e-01 4.584031e-01
> P2 4.432261e-02 4.688477e-02 6.502990e-02 7.295521e-02
> P3 4.095645e-03 4.637381e-03 6.863520e-03 9.297594e-03
> SL 1.983961e-04 2.412744e-04 3.354917e-04 5.336241e-04
> D 4.309986e-05 8.679096e-05 9.244503e-05 2.547176e-04
> XSm 2.759814e-04 4.812309e-04 4.249543e-04 9.095035e-04
> Sm 3.049081e-04 4.179082e-04 4.652088e-04 7.679929e-04
> Md 9.705559e-05 1.126409e-04 1.714340e-04 2.549410e-04
> Lg 4.938104e-05 1.777762e-05 7.269498e-05 3.391366e-05
> XLg 8.078062e-06 1.034075e-06 1.583885e-05 2.185849e-06
```

Now let's compare the stage distributions via a plot (figure 9.1).

```
ss_put_together <- cbind.data.frame(tm2ss_r$ss_prop, tm3ss_r$ahist$ss_prop,
    tm2ss_rs$ss_prop, tm3ss_rs$ahist$ss_prop)
names(ss_put_together) <- c("det ahist", "det hist", "sto ahist", "sto hist")
rownames(ss_put_together) <- tm2ss_r$stage
barplot(t(ss_put_together), beside=T, ylab = "Proportion", xlab = "Stage",
    ylim = c(0, 0.95), col = c("black", "orangered", "grey", "darkred"),
    bty = "n")
legend("topright", c("det ahist", "det hist", "sto ahist", "sto hist"),
    col = c("black", "orangered", "grey", "darkred"), pch = 15, cex = 0.9,
    bty = "n")
```

Overall, these are similar patterns. However, deterministic analyses suggest a greater share of the population should be composed of dormant seeds and $1^{\text {st }}$ year protocorms, and a lower share should be composed of $2^{\text {nd }}$ year protocorms protocorms.

Let's look at stochastic vs. deterministic reproductive values from ahistorical and historical approaches. First let's calculate the reproductive values under each scenario.

```
tm2rv_r <- repvalue3(cyp2r_mean)
tm3rv_r <- repvalue3(cyp3r_mean)
tm2rv_rs <- repvalue3(cypmatrix2r, stochastic = TRUE, seed = 42)
tm3rv_rs <- repvalue3(cypmatrix3r, stochastic = TRUE, seed = 42)
```



Figure 9.1: Ahistorical vs. historically-corrected stable and long-run mean stage distribution

```
tm2rv_rplot <- as.matrix(tm2rv_r$rep_value)
rownames(tm2rv_rplot) <- tm2rv_r$stage
tm2rv_rsplot <- as.matrix(tm2rv_rs$rep_value)
rownames(tm2rv_rsplot) <- tm2rv_rs$stage
tm3rv_rplot <- as.matrix(tm3rv_r$ahist$rep_value)
rownames(tm3rv_rplot) <- tm3rv_r$ahist$stage
tm3rv_rsplot <- as.matrix(tm3rv_rs$ahist$rep_value)
rownames(tm3rv_rsplot) <- tm3rv_rs$ahist$stage
```

Now let's compare them graphically (figure 9.2).

```
par(mfrow=c (2,2))
barplot(t(tm2rv_rplot), ylab = "Rep value", xlab = "", col = "black", bty = "n")
title("a)", adj = 0)
legend("topleft", c("det ahist", "det hist", "sto ahist", "sto hist"), cex =0.8,
    col = c("black", "orangered", "grey", "darkred"), pch = 15, bty = "n")
barplot(t(tm3rv_rplot), ylab = "", xlab = "", col = "orangered", bty = "n")
title("b)", adj = 0)
barplot(t(tm2rv_rsplot), ylab = "Rep value", xlab = "Stage", col = "grey",
    bty = "n")
title("c)", adj = 0)
barplot(t(tm3rv_rsplot), ylab = "", xlab = "Stage", col = "darkred", bty = "n")
title("d)", adj = 0)
```

We see some big differences here, particularly between ahistorical and historical analyses. Indeed, in the ahistorical case, reproductive value increases with size, reaching its peak in the largest adults. In contrast, in the historical case, the greatest reproductive values are associated with medium adults. So, history appears to have large effects here, while temporal stochasticity does not seem to have much influence.


Figure 9.2: Ahistorical (a, c) vs. historically-corrected (b, d) deterministic (a, b) and stochastic (c, d) reproductive values

### 9.3 Stochastic sensitivity analysis

Package lefko3 contains functions allowing users to conduct deterministic and stochastic sensitivity and elasticity analyses. Sensitivity and elasticity analysis are forms of perturbation analysis, and we urge readers to consult Caswell (2001) and Caswell (2019) to become fully acquainted with the topic. Here, we discuss just the most important aspects to understand these analyses as conducted using lefko3.

Stochastic sensitivity analysis assesses the impact of small, absolute changes in matrix elements on the $\log$ stochastic growth rate, $a=\log \lambda_{S}$ (Caswell, 2001). The stochastic sensitivity of $a$ to an element $a_{k j}$ is as follows.

$$
\begin{equation*}
\frac{\partial \log \lambda_{S}}{\partial a_{k j}}=\lim _{T \rightarrow \infty} \frac{1}{T} \sum_{t=0}^{T-1} \frac{\mathbf{v}(t+1) \mathbf{w}^{\top}(t)}{\mathbf{v}^{\top}(t+1) \mathbf{w}(t+1)} \tag{9.6}
\end{equation*}
$$

Here, $t$ refers to a specific occasion and $T$ refers to the number of occasions projected. Stochastic sensitivities for hMPMs may be converted to historically-corrected format as in the deterministic case.

Let's now try a stochastic sensitivity analysis. We'll start off by running the ahistorical MPM.

```
set.seed(42)
tm2sens_rs <- sensitivity3(cypmatrix2r, stochastic = TRUE)
# The highest stochastic sensitivity value
max(tm2sens_rs$ah_sensmats[[1]][which(cyp2r_mean$A[[1]] > 0)])
> [1] 1.066749
# This value is associated with element
intersect(which(tm2sens_rs$ah_sensmats[[1]] ==
    max(tm2sens_rs$ah_sensmats[[1]][which(cyp2r_mean$A[[1]] > 0)])),
    which(cyp2r_mean$A[[1]] > 0))
> [1] 38
```

The highest sensitivity value is associated with element 38 . Since there are 11 stages, and the element number refers to the position of the element within a vectorized form of the matrix, we can infer that $\log \lambda$ is most sensitive to the transition from $3^{\text {rd }}$ year protocorm to seedling. Check this by typing ceiling (38/11) to get the column number and $38 \% \% 11$ to get the row number for element 38 .

```
# From stage
    cypmatrix2r$ahstages[ceiling(38/dim(cypmatrix2r$ahstages) [1]),"stage"]
> [1] "P3"
# To stage
    cypmatrix2r$ahstages[38 %% dim(cypmatrix2r$ahstages)[1],"stage"]
> [1] "SL"
```

Let's compare this to the historical case. Note that the historical sensitivity matrices will be output in sparse format by default, so we need to use as.matrix() in some instances to make the code work properly.

```
set.seed(42)
tm3sens_rs <- sensitivity3(cypmatrix3r, stochastic = TRUE)
# The highest stochastic sensitivity value
max(tm3sens_rs$h_sensmats[[1]][which(cyp3r_mean$A[[1]] > 0)])
> [1] 1.576569
```

```
# This value is associated with element
intersect(which(as.matrix(tm3sens_rs$h_sensmats[[1]]) ==
    max(tm3sens_rs$h_sensmats[[1]][which(cyp3r_mean$A[[1]] > 0)])),
    which(cyp3r_mean$A[[1]] > 0))
> [1] 3063
```

Here we find that $\log \lambda$ is most sensitive to element 3063 . We can use the same trick as with the ahistorical case to determine which transition this is, but we need to use the hstages portion of the historical MPM rather than the ahstages element. The hstages element outlines the 121 stage-pairs corresponding to the rows and column in these matrices. So, let's figure this out, as below.

```
# From stage in t-1:
cypmatrix3r$hstages[ceiling(3063/dim(cypmatrix3r$hstages)[1]), "stage_1"]
> [1] "P2"
# From stage in t:
cypmatrix3r$hstages[ceiling(3063/dim(cypmatrix3r$hstages)[1]), "stage_2"]
> [1] "P3"
# To stage in t:
cypmatrix3r$hstages[3063 %% dim(cypmatrix3r$hstages)[1],"stage_1"]
> [1] "P3"
# To stage in t+1:
cypmatrix3r$hstages[3063 %% dim(cypmatrix3r$hstages)[1],"stage_2"]
> [1] "SL"
```

This transition corresponds to the transition from the $26^{\text {th }}$ stage pair ( $2^{\text {nd }}$ year and $3^{\text {rd }}$ year protocorms in times $t-1$ and $t$, respectively), to the $38^{\text {th }}$ stage pair ( $3^{\text {rd }}$ year protocorm and seedling in times $t$ and $t+1$, respectively). This is similar to our results from the deterministic sensitivity analysis in the last chapter (chapter 8).

### 9.4 Stochastic elasticity analysis

Elasticity analyses assess the impacts of small, proportional changes in matrix elements on population growth rate. In stochastic elasticity analysis, the population growth rate used is the log stochastic growth rate, $a=\log \lambda_{S}$ (Caswell, 2001). The stochastic elasticity of $a=\log \lambda_{S}$ to changes in an element is given as follows.

$$
\begin{equation*}
\frac{\partial \log \lambda_{S}}{\partial \log a_{k j}}=\lim _{T \rightarrow \infty} \frac{1}{T} \sum_{t=0}^{T-1} \frac{\left(\mathbf{v}(t+1) \mathbf{w}^{\top}(t)\right) \circ \mathbf{A}_{\mathbf{t}}}{\mathbf{v}^{\top}(t+1) \mathbf{w}(t+1)} \tag{9.7}
\end{equation*}
$$

Here, $\mathbf{A}_{\mathbf{t}}$ refers to the A matrix corresponding to occasion $t$. Stochastic elasticity values for hMPMs may be converted to historically-corrected format as in the deterministic case, and may also be summed as before. Elements estimated to be exactly zero must also have elasticity values equal to zero, making their interpretation even more different than sensitivity values.

Let's assess the elasticity of $a=\log \lambda_{S}$ to matrix elements, comparing the ahistorical to the historically-corrected case in stochastic analyses.

```
set.seed(42)
tm2elas_rs <- elasticity3(cypmatrix2r, stochastic = TRUE)
set.seed(42)
tm3elas_rs <- elasticity3(cypmatrix3r, stochastic = TRUE)
```

```
# Max ahistorical stoch elasticity occurs in element
which(tm2elas_rs$ah_elasmats[[1]] == max(tm2elas_rs$ah_elasmats[[1]]))
> [1] 121
# Max historically-corrected stoch elasticity occurs in element
which(as.matrix(tm3elas_rs$ah_elasmats[[1]]) == max(tm3elas_rs$ah_elasmats[[1]]))
> [1] 73
# Max historical stoch elasticity occurs in element
which(as.matrix(tm3elas_rs$h_elasmats[[1]]) == max(tm3elas_rs$h_elasmats[[1]]))
> [1] }878
```

The ahistorical and historical analyses generally agree that $\log \lambda_{S}$ is most elastic to the ahistorical stasis transition from Small adult in time $t$ to Small adult in time $t+1$, and the historical stasis transition as Small adult in times $t-1, t$, and $t+1$. This is a different result from sensitivity analysis, where the ahistorical and historical stochastic sensitivity analyses suggested that population growth rate was most sensitivity to the transition from $3^{\text {rd }}$ year protocorm to seedling (with $2^{\text {nd }}$ year protocorm in time $t$ - 1 in the historical case).

Let's compare the elasticity of population growth to life history stages (figure 9.3).

```
elas_put_together <- cbind.data.frame(colSums(tm2elas_rs$ah_elasmats[[1]]),
    colSums(as.matrix(tm3elas_rs$ah_elasmats[[1]])))
names(elas_put_together) <- c("sto ahist", "sto hist")
rownames(elas_put_together) <- tm2elas_rs$ahstages$stage
barplot(t(elas_put_together), beside=T, ylab = "Elasticity", xlab = "Stage",
    col = c("grey", "darkred"), ylim = c(0, 0.50), bty = "n")
legend("topright", c("sto ahist", "sto hist"), col = c("grey", "darkred"),
    pch = 15, bty = "n")
```



Figure 9.3: Ahistorical vs. historically-corrected stochastic elasticity to stage
Elasticity patterns in these plots are somewhat similar, though with some key differences. The ahistorical analysis suggests that $\log \lambda_{S}$ is most elastic in response to changes in transitions from the extra large adult stage, followed by extra small and small adults. The historical analysis suggests that $\log \lambda_{S}$ is most elastic in response to changes in extra small and small adults, followed by extra large adults. Dormant seeds and dormant adults appear to have little influence.

Finally, let's look at the elasticity sums of different transition types (figure 9.4).

```
tm2elas_rs_sums <- summary(tm2elas_rs)
tm3elas_rs_sums <- summary(tm3elas_rs)
elas_sums_together <- cbind.data.frame(tm2elas_rs_sums$ahist[,2],
    tm3elas_rs_sums$ahist[,2])
names(elas_sums_together) <- c("sto ahist", "sto hist")
rownames(elas_sums_together) <- tm2elas_rs_sums$ahist$category
barplot(t(elas_sums_together), beside=T, ylab = "Elasticity",
    xlab = "Transition", col = c("grey", "darkred"), ylim = c(0, 0.60), bty = "n")
legend("topright", c("sto ahist", "sto hist"), col = c("grey", "darkred"),
    pch = 15, bty = "n")
```



Figure 9.4: Ahistorical vs. historically-corrected elasticity of $a$ to transitions

Both ahistorical an historical analyses show reasonably similar values for transition types, and suggest that $\log \lambda_{S}$ is most elastic in response to stasis transitions. In both analyses, shrinkage has the least influence (figure 9.5).

```
elas_hist2plot <- as.matrix(tm3elas_rs_sums$hist[,2])
rownames(elas_hist2plot) <- tm3elas_rs_sums$hist$category
par(mar = c(7, 4, 2, 2) + 0.2)
barplot(t(elas_hist2plot), ylab = "Elasticity", xlab = "", xaxt = "n",
    col = c("orangered", "darkred"), bty = "n")
text(cex=0.6, x=seq(from = 0, to = 1.1*length(tm3elas_rs_sums$hist$category),
    by = 1.15), y=-0.088, tm3elas_rs_sums$hist$category, xpd=TRUE, srt=45)
```

We can see that full stasis across occasions $t-1$, $t$, and $t+1$ is associated with the greatest summed elasticity, followed by growth to stasis and stasis to growth. Transitions associated with fecundity are typically associated with the lowest summed elasticity values, except for fecundity to growth.

Package lefko3 also includes two functions to conduct general projection simulations, including stochastic simulations. Please see chapter 10 for more.


Figure 9.5: Elasticity of $a$ to historical transitions

### 9.5 Points to remember

1. Stochastic analysis typically refers to matrix projection in which matrices are shuffled randomly a large number of times. The properties of the projection in the long-term have a tendency to be approximately stationary. These analyses assess the impacts of random environmental shifts across time on population dynamics. Although stochasticity may also be demographic or spatial, we concern ourselves only with temporal stochasticity in this chapter because of its dominance in the literature.
2. Package lefko3 can conduct all major stochastic analyses even for large numbers of massive matrices, such as historical IPMs and age-by-stage MPMs. This includes even sensitivity and elasticity analyses.
3. In most cases, stochastic analysis in lefko3 is handled with the same functions as deterministic analysis, but using the option stochastic = TRUE.

## Chapter 10

## Population Projection III: Projection Simulations

Anyone who believes in indefinite growth of anything physical on a physically finite planet is either a madman or an economist.

- Kenneth Boulding

So far we have covered a number of standard approaches to matrix projection analysis. Basic deterministic analyses and projection assuming temporal environmental stochasticity dominate the MPM literature. However, these analyses have strong assumptions that are often not met in real life. Indeed, lack of concordance between these assumptions and reality may be behind the general failure of matrix-based population viability analyses to predict population dynamics successfully beyond just a few time steps into the future (Crone et al., 2013).

Package lefko3 provides some general projection options for users interested in more complex analyses. For example, users can project function-based matrices using a particular set of individual covariate values, or conduct density-dependent analyses using a variety of density dependence functions, or even run cyclical projections or replicate stochastic simulations. Further, package lefko3 provides the added capability to enforce substochasticity in all projections, thus preventing analysis from involving impossible rates and probabilities.

Chapter 9 introduced an equation showing how matrices are projected under the assumption of temporal environmental stochasticity, shown below.

$$
\begin{equation*}
\mathbf{n}_{\mathbf{t}}=\mathbf{A}_{\mathbf{t}} \mathbf{A}_{\mathbf{t}-\mathbf{1}} \cdots \mathbf{A}_{1} \mathbf{n}_{\mathbf{0}} \tag{10.1}
\end{equation*}
$$

This is really a general projection equation, where each $\mathbf{A}_{\mathbf{i}}$ simply represents the matrix to be used at time $i$. We can project forward any number of times, perhaps only a small number to see how a particular management regime might impact population dynamics in the immediate term, or perhaps a very large number to see what long-term patterns we find, such as cycles, plateaus, or chaos. These matrices can be chosen randomly or set to a specific order.

Package lefko3 provides two functions for general projection: projection3() and f_projection3(). The former is used to project matrices that have already been developed and exist within a lefkoMat object. The latter is used to take vital rate models and build new function-based matrices for each time step. Both functions have been developed to run as binaries in R, making them run as fast as possible.

Many kinds of projections are possible with these functions, including deterministic, ordered / cyclical, and stochastic, both with and without density dependence. Let's now to turn to density dependence, in particular, since we have not discussed this topic before.

### 10.1 Density dependence

Density dependent matrix projections involve the modification of matrix elements or vital rates by some function of population size. Many different functions can be applied to incorporate density dependence into matrix projection. The first projections typically applied a function such as the logistic function to all matrix elements, generally multiplying each matrix element by some function of density at either the current time or some time in the past (e.g., Leslie, 1959). More recent approaches typically separate density dependent functions by vital rate, and alter only certain specific matrix elements (e.g., Jensen, 1995).

To provide the greatest flexibility, lefko3 currently implements density dependence on matrix elements using the density_input() function, which allows users to list all matrix elements that should be subject to density dependence, and to stipulate the characteristics of the density dependence (the function density_vr() can be used to incorporate density dependence characteristics on vital rates, if a function-based MPM is used). Density dependence is assigned to specific elements or sets of elements by noting the transitions to be targeted, using the same format as in function supplemental (). Shorthand abbreviations can be used, such as all for all stages, rep for reproductive stages, nrep for mature but non-reproductive stages, etc. A default time delay of one time step is applied, and this can be set to any positive integer. Currently, lefko3 includes four density dependence functions that can be chosen, with projections capable of including multiple functions and time delays. The first is the two-parameter Ricker function, given below.

$$
\begin{equation*}
\phi_{t+1}=\phi_{t} \times \alpha e^{-\beta n_{t}} \tag{10.2}
\end{equation*}
$$

Here, $\alpha$ and $\beta$ are the density dependence parameters, and $\beta$ in particular gives the strength of density dependence. This equation is among the most interesting in the density dependence literature, because it is capable of producing plateaus, single or multi-period oscillations, and even chaos.

The second density dependence function in lefko3 is the two-parameter Beverton-Holt function, shown below.

$$
\begin{equation*}
\phi_{t+1}=\phi_{t} \times \frac{\alpha}{1+\beta n_{t}} \tag{10.3}
\end{equation*}
$$

Here, $\alpha$ and $\beta$ are, once again, the density dependence parameters, and $\beta$ in particular gives the strength of density dependence. This function generally asymptotes at an equilibrium density, and so is not capable of producing some of the complex patterns that the Ricker function is capable of.

The third function implemented is the Usher function, given below.

$$
\begin{equation*}
\phi_{t+1}=\phi_{t} \times \frac{1}{1+e^{\alpha n_{t}+\beta}} \tag{10.4}
\end{equation*}
$$

Here, both $\alpha$ and $\beta$ give the strength of density dependence, the former via an interaction with density and the latter via an addition to the exponential effect.

Finally, lefko3 also implements a form of the logistic function, given below.

$$
\begin{equation*}
\phi_{t+1}=\phi_{t} \times\left(1-\frac{n_{t}}{K}\right) \tag{10.5}
\end{equation*}
$$

The logistic function classically takes only one parameter as input, the carrying capacity $K$. For this function, the user may also stipulate whether $K$ is a hard limit, or whether the time lag in density can result in overshooting $K$. These can both be set using alpha to set $K$ and beta to set whether $K$ is a hard limit in the density_input () function.

Users working with density dependence forms that involve user-provided exponents should be aware of the limits of their computers to handle particularly large and particularly small numbers. Generally speaking, the largest number capable of being handled by most computers is likely to be $1.7976931 \times$ $10^{308}$. This roughly corresponds to $e^{709.7}$. The minimum limit of $2.2250739 \times 10^{-308}$ corresponds to $e^{-708.3}$ If these limits are crossed, then projections will likely still run but yield absurd results, such as switching to negative numbers of individuals in a single time step.

### 10.1.1 Substochasticity

Functions projection3() and f_projection3() allow the user to stipulate whether the projection should force matrices to remain substochastic. Substochasticity refers to the condition that estimated probabilities stay within the bounds of probabilities, meaning that they should not equal values below 0.0 or above 1.0. Values outside this range may occur when using some density dependence functions, for example when survival transitions are subject to the Ricker function.

Two forms of substochastic enforcement are included in these functions: mild and hard. In the mild form, matrix elements used as survival-transition probabilities are prevented from moving outside of the interval $[0,1]$. However, simply adjusting matrix elements to the interval $[0,1]$ may still lead to total survival probabilities greater than 1.0 , potentially leading to biased analyses. This happens because the column sums in each survival-transition matrix, $\mathbf{U}$, must equal the overall time-step survival of each stage, age-stage combination, stage-pair, or whatever aspect of the life history model that each column represents. That is why we have also developed the hard form of substochasticity, in which survival-transition matrix column sums are kept within the interval [0, 1] by adjusting matrix elements by proportionate amounts. Both forms of substochastic enforcement also prevent fecundity from becoming negative. The default setting does not enforce substochasticity, but we encourage users to try enforcing substochasticity when experimenting with density dependence, as logically impossible values may occur, and the consequences of incorporating logically impossible values in analyses of population dynamics are largely unknown (Caswell, 2001).

Let's consider substochastic forcing mathematically. Both forms of substochastic enforcement start by identifying which elements are negative, and changing them to 0 . This is done first within the $\mathbf{U}$ matrix, and then in the $\mathbf{F}$ matrix. Thus, if $U_{i j}$ is the survival transition element in the $i$ th row and the $j$ th column, then we have the following.

$$
U_{i j}=\left\{\begin{array}{cc}
U_{i j} & U_{i j}>0  \tag{10.6}\\
0 & U_{i j}<0
\end{array}\right.
$$

The above applies for all $U_{i j}$. The equation is the same for all $F_{i j}$.
Once this operation is complete, mild substochastic forcing changes all survival-transition elements above 1.0 to 1.0 , as in the following for all $U_{i j}$.

$$
U_{i j}=\left\{\begin{array}{cc}
U_{i j} & U_{i j}<1  \tag{10.7}\\
1 & U_{i j}>1
\end{array}\right.
$$

Under hard substochastic forcing, we instead identify survival-transition column sums greater than 1.0 and alter survival-transitions, as below for all $U_{i j}$ and with $m$ referring to the number of rows (and hence columns, since the matrices must be square).

$$
U_{i j}=\left\{\begin{array}{cc}
U_{i j} & \sum_{i}^{m} U_{i j}<1  \tag{10.8}\\
\frac{U_{i j}}{\sum_{i}^{m} U_{i j}} & \sum_{i}^{m} U_{i j}>1
\end{array}\right.
$$

### 10.2 Projecting existing MPMs

In this first half of the chapter, let's explore general projection simulations with projection3() using our raw MPMs developed with the Cypripedium candidum dataset. Here we see the life history model that we will use, which was originally introduced in chapter 3 .

As it has been several chapters since we have seen that code, we show the full code to build our raw Cypripedium MPMs below.


Figure 10.1: Life history model of Cypripedium candidum for use in raw MPMs. The blue box shows the stages that were actually monitored within the field study.

```
rm(list=ls(all=TRUE))
library(lefko3)
data(cypdata)
sizevector <- c(0, 0, 0, 0, 0, 0, 1, 3, 6, 11, 19.5)
stagevector <- c("SD", "P1", "P2", "P3", "SL", "D", "XSm", "Sm", "Md", "Lg",
    "XLg")
repvector <- c(0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1)
obsvector <- c(0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1)
matvector <- c(0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1)
immvector <- c(0, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0)
propvector <- c(1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
indataset <- c(0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1)
binvec <- c(0, 0, 0, 0, 0, 0.5, 0.5, 1.5, 1.5, 3.5, 5)
comments <- c("Dormant seed", "1st yr protocorm", "2nd yr protocorm",
    "3rd yr protocorm", "Seedling", "Dormant adult",
    "Extra small adult (1 shoot)", "Small adult (2-4 shoots)",
    "Medium adult (5-7 shoots)", "Large adult (8-14 shoots)",
    "Extra large adult (>14 shoots)")
cypframe_raw <- sf_create(sizes = sizevector, stagenames = stagevector,
    repstatus = repvector, obsstatus = obsvector, matstatus = matvector,
    propstatus = propvector, immstatus = immvector, indataset = indataset,
    binhalfwidth = binvec, comments = comments)
```

```
cypraw_v1 <- verticalize3(data = cypdata, noyears = 6, firstyear = 2004,
```

    patchidcol = "patch", individcol = "plantid", blocksize = 4,
    ```
    sizeacol = "Inf2.04", sizebcol = "Inf.04", sizeccol = "Veg.04",
    repstracol = "Inf.04", repstrbcol = "Inf2.04", fecacol = "Pod.04",
    stageassign = cypframe_raw, stagesize = "sizeadded", NAas0 = TRUE,
    NRasRep = TRUE)
seeds_per_fruit <- 5000
sl_mult <- 0.7
cypsupp2_raw <- supplemental(stage3 = c("SD", "P1", "P2", "P3", "SL", "SL", "D",
        "XSm", "Sm", "SD", "P1"),
    stage2= c("SD", "SD", "P1", "P2", "P3", "SL", "SL", "SL", "SL", "rep", "rep"),
    eststage3 = c(NA, NA, NA, NA, NA, NA, "D", "XSm", "Sm", NA, NA),
    eststage2 = c(NA, NA, NA, NA, NA, NA, "XSm", "XSm", "XSm", NA, NA),
    givenrate =c(0.08,0.10, 0.10, 0.10, 0.05, 0.05, NA, NA, NA, NA, NA),
    multiplier = c(NA, NA, NA, NA, NA, NA, sl_mult, sl_mult, sl_mult,
        0.5 * seeds_per_fruit, 0.5 * seeds_per_fruit),
    type =c(1, 1, 1, 1, 1, 1, 1, 1, 1, 3, 3),
    stageframe = cypframe_raw, historical = FALSE)
cypsupp3_raw <- supplemental(stage3 = c("SD", "SD", "P1", "P1", "P2", "P2",
        "P3", "SL", "SL", "SL", "D", "XSm", "Sm", "D", "XSm", "Sm", "mat", "mat",
        "mat", "SD", "P1"),
    stage2 = c("SD", "SD", "SD", "SD", "P1", "P1", "P2", "P3", "SL", "SL", "SL",
        "SL", "SL", "SL", "SL", "SL", "D", "XSm", "Sm", "rep", "rep"),
    stage1 = c("SD", "rep", "SD", "rep", "SD", "rep", "P1", "P2", "P3", "SL",
        "SL", "SL", "SL", "P3", "P3", "P3", "SL", "SL", "SL", "mat", "mat"),
    eststage3 = c(NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, "D", "XSm", "Sm", "D",
        "XSm", "Sm", "mat", "mat", "mat", NA, NA),
    eststage2 = c(NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, "XSm", "XSm", "XSm",
        "XSm", "XSm", "XSm", "D", "XSm", "Sm", NA, NA),
    eststage1 = c(NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, "XSm", "XSm", "XSm",
        "XSm", "XSm", "XSm", "XSm", "XSm", "XSm", NA, NA),
    givenrate = c(0.08, 0.10, 0.10, 0.10, 0.10, 0.10, 0.10, 0.05, 0.05, 0.05, NA,
        NA, NA, NA, NA, NA, NA, NA, NA, NA, NA),
    multiplier = c(NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, sl_mult, sl_mult,
        sl_mult, sl_mult, sl_mult, sl_mult, 1, 1, 1, 0.5 * seeds_per_fruit,
        0.5 * seeds_per_fruit),
    type =c(1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 3, 3),
    stageframe = cypframe_raw, historical = TRUE)
cypmatrix2r <- rlefko2(data = cypraw_v1, stageframe = cypframe_raw,
    year = "all", stages = c("stage3", "stage2"),
    size = c("size3added", "size2added"), supplement = cypsupp2_raw,
    yearcol = "year2", indivcol = "individ")
cypmatrix2rp <- rlefko2(data = cypraw_v1, stageframe = cypframe_raw,
    year = "all", patch = "all", stages = c("stage3", "stage2"),
    size = c("size3added", "size2added"), supplement = cypsupp2_raw,
    yearcol = "year2", patchcol = "patchid", indivcol = "individ")
cypmatrix3r <- rlefko3(data = cypraw_v1, stageframe = cypframe_raw,
    year = "all", stages = c("stage3", "stage2", "stage1"),
```

```
    size = c("size3added", "size2added", "size1added"), supplement = cypsupp3_raw,
    yearcol = "year2", indivcol = "individ", sparse_output = TRUE)
cypmatrix3rp <- rlefko3(data = cypraw_v1, stageframe = cypframe_raw,
    year = "all", patch = "all", stages = c("stage3", "stage2", "stage1"),
    size = c("size3added", "size2added", "size1added"), supplement = cypsupp3_raw,
    yearcol = "year2", patchcol = "patchid", indivcol = "individ",
    sparse_output = TRUE)
```

We see that we have four MPMs. The first two, cypmatrix2r and cypmatrix2rp, are ahistorical MPMs at the population and patch level, respectively. The next two, cypmatrix3r and cypmatrix3rp, are historical MPMs at the population and patch level, respectively. Let's see their summaries to jumpstart our memories about them.

```
summary(cypmatrix2r)
>
> This ahistorical lefkoMat object contains 5 matrices.
>
> Each matrix is square with }11\mathrm{ rows and columns, and a total of 121 elements.
> A total of }120\mathrm{ survival transitions were estimated, with 24 per matrix.
> A total of 40 fecundity transitions were estimated, with 8 per matrix.
> This lefkoMat object covers 1 population, 1 patch, and 5 time steps.
>
> The dataset contains a total of }74\mathrm{ unique individuals and 320 unique transitions.
>
> Survival probability sum check (each matrix represented by column in order):
> [,1] [,2] [,3] [,4] [,5]
> Min. 0.000 0.050 0.050 0.000 0.050
> 1st Qu. 0.100 0.140 0.140 0.100 0.140
> Median 0.689 0.870 0.864 0.610 0.882
> Mean 0.552 0.629 0.629 0.528 0.627
> 3rd Qu. 1.000 1.000 1.000 0.960 1.000
> Max. 1.000 1.000 1.000 1.000 1.000
summary(cypmatrix2rp)
>
> This ahistorical lefkoMat object contains 15 matrices.
>
> Each matrix is square with }11\mathrm{ rows and columns, and a total of 121 elements.
> A total of 266 survival transitions were estimated, with 17.733 per matrix.
> A total of }70\mathrm{ fecundity transitions were estimated, with 4.667 per matrix.
> This lefkoMat object covers 1 population, 3 patches, and 5 time steps.
>
> The dataset contains a total of 74 unique individuals and 320 unique transitions.
>
> Survival probability sum check (each matrix represented by column in order):
> [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11] [,12]
> Min. 0.000 0.000 0.000 0.000 0.000 0.000 0.050 0.050 0.000 0.050 0.000 0.000
> 1st Qu. 0.075 0.025 0.075 0.025 0.075 0.075 0.140 0.140 0.100 0.140 0.100 0.100
> Median 0.180 0.100 0.180 0.100 0.180 0.180 0.909 0.778 0.686 0.857 0.750 0.575
> Mean 0.457 0.361 0.471 0.328 0.417 0.464 0.631 0.611 0.530 0.631 0.562 0.523
> 3rd Qu. 0.955 0.769 1.000 0.592 0.781 1.000 1.000 1.000 0.955 1.000 1.000 1.000
> Max. 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000
```

```
> [,13] [,14] [,15]
> Min. 0.000 0.000 0.000
> 1st Qu. 0.075 0.075 0.100
> Median 0.180 0.180 0.750
> Mean 0.432 0.450 0.562
> 3rd Qu. 0.875 1.000 1.000
> Max. 1.000 1.000 1.000
summary(cypmatrix3r)
>
> This historical lefkoMat object contains 4 matrices.
>
> Each matrix is square with }121\mathrm{ rows and columns, and a total of 14641 elements.
> A total of 242 survival transitions were estimated, with 60.5 per matrix.
> A total of 54 fecundity transitions were estimated, with 13.5 per matrix.
> This lefkoMat object covers 1 population, 1 patch, and 4 time steps.
>
> The dataset contains a total of 74 unique individuals and 320 unique transitions.
>
> Survival probability sum check (each matrix represented by column in order):
> [,1] [,2] [,3] [,4]
> Min. 0.000 0.000 0.000 0.000
> 1st Qu. 0.000 0.000 0.000 0.000
> Median 0.000 0.000 0.000 0.000
> Mean 0.173 0.179 0.166 0.198
> 3rd Qu. 0.100 0.100 0.100 0.100
> Max. 1.000 1.000 1.000 1.000
summary(cypmatrix3rp)
>
> This historical lefkoMat object contains 12 matrices.
>
> Each matrix is square with }121\mathrm{ rows and columns, and a total of 14641 elements.
> A total of 516 survival transitions were estimated, with 43 per matrix.
> A total of 70 fecundity transitions were estimated, with 5.833 per matrix.
> This lefkoMat object covers 1 population, 3 patches, and 4 time steps.
>
> The dataset contains a total of 74 unique individuals and 320 unique transitions.
>
> Survival probability sum check (each matrix represented by column in order):
> [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11]
>Min. 0.000 0.0000 0.0000 0.000 0.000 0.000 0.00 0.000 0.000 0.0000 0.000
> 1st Qu. 0.000 0.0000 0.0000 0.000 0.000 0.000 0.00 0.000 0.000 0.0000 0.000
> Median 0.000 0.0000 0.0000 0.000 0.000 0.000 0.00 0.000 0.000 0.0000 0.000
> Mean 0.107 0.0945 0.0851 0.101 0.158 0.158 0.14 0.169 0.119 0.0851 0.119
> 3rd Qu. 0.000 0.0000 0.0000 0.000 0.100 0.100 0.05 0.100 0.000 0.0000 0.000
> Max. 1.000 1.0000 1.0000 1.000 1.000 1.000 1.00 1.000 1.000 1.0000 1.000
> [,12]
>Min. 0.000
> 1st Qu. 0.000
> Median 0.000
> Mean 0.144
> 3rd Qu. 0.000
```

> Max. 1.000
Users of lefko3 might wish to conduct several styles of projections. In this first half of the chapter, we will focus on the following types of projection with existing lefkoMat objects:

1. Deterministic projections of a specific time length
2. Ordered or cyclical projections of a specific time length
3. Replicated simulations assuming temporal stochasticity
4. Density dependent deterministic projections
5. Ordered or cyclical density dependent simulations
6. Replicated simulations assuming temporal stochasticity and density dependence

Projection styles $1,2,4$, and 5 assume no randomness and so we assume that users are not interested in producing replicates for those styles. However, both projection3() and f_projection3() allow replication for all styles. Let's begin with the first style of projection.

### 10.2.1 Deterministic projections of a specific time length

One of the most common projection styles involves projecting an arithmetic mean matrix deterministically. Perhaps we wish to project the mean forward by ten time steps simply to see the short-term predicted trend, or perhaps we wish to project many thousands of time steps in order to see the asymptotic properties of the matrix. Let's start off by building arithmetic mean matrices for the ahistorical and historical population-level MPMs.

```
cypmean2 <- lmean(cypmatrix2r)
cypmean3 <- lmean(cypmatrix3r)
summary(cypmean2)
>
> This ahistorical lefkoMat object contains 1 matrix.
>
> Each matrix is square with }11\mathrm{ rows and columns, and a total of 121 elements.
> A total of 33 survival transitions were estimated, with 33 per matrix.
> A total of 10 fecundity transitions were estimated, with 10 per matrix.
> This lefkoMat object covers 1 population, 1 patch, and O time steps.
>
> The dataset contains a total of 74 unique individuals and 320 unique transitions.
>
> Survival probability sum check (each matrix represented by column in order):
> [,1]
Min. 0.050
1st Qu. 0.140
Median 0.800
Mean 0.593
3rd Qu. 0.921
Max. 1.000
ummary(cypmean3)
>
This historical lefkoMat object contains 1 matrix.
>
Each matrix is square with }121\mathrm{ rows and columns, and a total of 14641 elements.
```

```
> A total of 99 survival transitions were estimated, with 99 per matrix.
> A total of 28 fecundity transitions were estimated, with 28 per matrix.
> This lefkoMat object covers 1 population, 1 patch, and 0 time steps.
>
> The dataset contains a total of }74\mathrm{ unique individuals and 320 unique transitions.
>
> Survival probability sum check (each matrix represented by column in order):
> [,1]
> Min. 0.000
> 1st Qu. 0.000
> Median 0.000
> Mean 0.179
> 3rd Qu. 0.200
> Max. 1.000
```

Now let's project these two mean matrices forward. Since these are only single matrices, we do not need to indicate any specific year or patch to project. We will instruct $R$ to project the ahistorical mean ten time steps forward. Then, we will take a look at the resulting object.

```
cypproj2 <- projection3(cypmean2, times = 10)
> Warning: This function takes annual matrices as input. This lefkoMat object appears to be a set of
cypproj2
> $projection
> $projection[[1]]
> $projection[[1]][[1]]
> [,1] [,2] [,3] [,4] [,5] [,6]
> [1,] 1 1.025278e+04 8976.9428602 7579.2162919 6559.5257069 5791.2902872
> [2,] 1 1.025280e+04 9181.9984786 7758.7551491 6711.1100327 5922.4808013
> [3,] 1 1.000000e-01 1025.2800919 918.1998479 775.8755149 671.1110033
```



```
> [5,] 1 1.000000e-01 0.0100000 0.0010000 5.1264505 4.8473218
> [6,] 1 2.326441e-01 0.2158192 0.2037291 0.1945073 0.4237303
> [7,] 1 1.496794e+00 1.3301888 1.2290474 1.1663551 2.9546675
> [8,] 1 1.804320e+00 1.8827350 1.8582091 1.7999057 2.7390456
> [9,] 1 8.390066e-01 0.8995651 
> [10,] 1 1.075116e+00 0.9570211 0.8349439 0.7303300 0.6449596
> [11,] 
> [1,] 5777.0586419 6330.3074391 7144.4860609 8023.4195510 8864.9356819
> [2,] 5892.8844477 6445.8486120 7271.0922097 8166.3092722 9025.4040729
> [3,] 592.2480801 589.2884448 644.5848612 
> [4,] 67.1111003 59.2248080 58.9288445 64.4584861 
> [l[\mp@code{[] 4.1217437 3.5616422 }
> [lllllllllll
> [7,] 4.0352491 4.6021723 4.9080956 5.0742834 5.2639516
> [8,] 3.7343588 4.5180195 5.0926985 5.5058964 5.8572084
> [9,] 0.9816220 1.2047260 1.4431232 1.6600921 1.8434275
> [10,] 
> [11,] 0.1524758 0.1366669 0.1329884 0.1382168
>
>
>
```

```
> $stage_dist
> $stage_dist[[1]]
> [1] 0
>
>
> $rep_value
> $rep_value[[1]]
> [1] 0
>
>
> $pop_size
> $pop_size[[1]]
> [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8]
> [1,] 11 20511.92 19189.93 16364.03 14148.47 12475.11 12343.56 13440.07
> [,9] [,10] [,11]
[1,] 15135.32 16998.41 18797.93
>
>
$labels
    pop patch
> 1 1 1
>
> $ahstages
\begin{tabular}{|c|c|c|c|c|c|c|c|}
\hline > 1 & 1 & SD & - 0.0 & NA & NA & 0 & NA \\
\hline \(>2\) & 2 & P1 & 0.0 & NA & NA & 0 & NA \\
\hline 3 & 3 & P2 & 0.0 & NA & NA & 0 & NA \\
\hline 4 & 4 & P3 & 0.0 & NA & NA & 0 & NA \\
\hline \(>5\) & 5 & SL & 0.0 & NA & NA & 0 & NA \\
\hline 6 & 6 & D & 0.0 & NA & NA & 0 & NA \\
\hline 7 & 7 & XSm & 1.0 & NA & NA & 0 & NA \\
\hline 8 & 8 & Sm & 3.0 & NA & NA & 0 & NA \\
\hline 9 & 9 & Md & 6.0 & NA & NA & 0 & NA \\
\hline > 10 & 10 & Lg & 11.0 & NA & NA & 0 & NA \\
\hline > 11 & 11 & XLg & 19.5 & NA & NA & 0 & NA \\
\hline
\end{tabular}
> repstatus obsstatus propstatus immstatus matstatus entrystage indataset
> 1 
> 2 0
> 3 0
>4 0
>5 0
> 6 0
> 7
> 8 1 1 1 0
\begin{tabular}{llllllll}
\(>\) & 9 & 1 & 1 & 0 & 0 & 1 & 0 \\
\(\gg 10\) & 1 & 1 & 0 & 0 & 1 & 0 & 1 \\
\(>\) & 11 & 1 & 1 & 0 & 0 & 1 & 0
\end{tabular}
> binhalfwidth raw sizebin min sizebin max sizebin center sizebin width
> 1 0.0 0.0 0.0.0 0.0
> 2 0.0 0.0 0.0
llllll
```



```
> 1 NA
>
> $control
> [1] 1 10
>
> attr(,"class")
> [1] "lefkoProj"
```

Here we can see the structure of the resulting object, which is of class lefkoProj. Its elements include:

1. projection - A list with two levels of elements. The top-level elements correspond to the patches or population-patch combinations in the input lefkoMat object, with as many elements as there are population-patch combinations. Each of these top-level elements is also a list with number of elements equal to the number of replicates (option nreps, which defaults to 1). Each element of this lower-level list is a matrix with rows corresponding to the MPM stages (if ahistorical; paired stages if historical; ages if Leslie; and age-stages if age-by-stage), and columns corresponding to the projected times (total number of columns equal to the number of time steps plus 1, with the first time referred to as time 0). The elements of the matrix show the projected number of individuals in the stages, ages, age-stages, or historical stage-pairs in that time.
2. stage_dist - A list showing the stage distribution projected per time, with the same order and structure as projection. Only estimated if growthonly = FALSE.
3. rep__value - A list showing the reproductive value of each stage projected per time, with the same order and structure as projection. Only estimated if growthonly $=$ FALSE.
4. pop_size - A list with number of elements equal to the number of projected patches or population-patches. Each element of this list is a matrix showing the total population size at each time (column) per replicate (row).
5. labels - A data frame giving the population, patch, and year of each matrix in the lefkoMat object used as input, in order.
6. ahstages - The stageframe used in analysis, though modified and edited according to MPM creation conventions.
7. hstages - A data frame matrix showing the pairing of ahistorical stages used to create historical stage pairs (only if historical).
8. agestages - A data frame showing the order of age-stage pairs (only if age-by-stage).
9. control - An integer vector indicating the number of replicates and the number of time steps.
10. density - The data frame input under the density option. Only provided if input by the user.

Let's also conduct a historical projection, and then compare summaries.

```
cypproj3 <- projection3(cypmean3, times = 10)
Warning: This function takes annual matrices as input. This lefkoMat object appears to be a set of
summary(cypproj2)
>
> The input lefkoProj object covers 1 population-patches.
> It includes 10 projected steps per replicate and 1 replicates.
> The number of replicates with population size above the threshold size of 1 is as in
> the following matrix, with pop-patches given by column and milepost times given by row:
> $milepost_sums
        11
1 1
```

```
> 3 1
> 6 1
> 8 1
> 11 1
>
> $extinction_times
> [1] NA
summary(cypproj3)
>
> The input lefkoProj object covers 1 population-patches.
> It includes 10 projected steps per replicate and 1 replicates.
> The number of replicates with population size above the threshold size of 1 is as in
> the following matrix, with pop-patches given by column and milepost times given by row:
> $milepost_sums
> 11
> 1 1
> 3 1
> 6 1
> 8 1
> 11 1
>
> $extinction_times
> [1] NA
```

The summary () calls give us a reasonable amount of information, showing the number of populationpatches, projected time steps and number of replicates. We also see how many replicates still have population sizes above 1.0 at $0 \%, 25 \%, 50 \%, 75 \%$, and $100 \%$ of the way through the projection. A number of options exist for summary() calls with projection3() and f_projection3(), and we encourage the user to explore upon reading the help pages for function summary.lefkoProj.

Let's also plot the projections for comparison (figure 10.2).

```
par(mfrow=c(1,2))
plot(cypproj2)
title("a)", adj = 0)
plot(cypproj3, ylab = "")
title("b)", adj = 0)
```

These are fascinating results suggesting that accounting for history reduces the population growth rate in this example. Indeed, the ahistorical population projection shows growth, while the historical projection suggests a steeper initial decline followed by more gradual population growth.

A close look at the population sizes projected at each time shows that decimal values are allowed. Naturally, partial individuals cannot exist in a population. Although the default in lefko3 is to project fractional individuals, we may allow only integers, which results in the projected number of individuals to be rounded down to the nearest non-negative integer. We can set this with the integeronly option, as below (figure 10.3).

```
cypproj3i <- projection3(cypmean3, times = 10, integeronly = TRUE)
> Warning: This function takes annual matrices as input. This lefkoMat object appears to be a set of
plot(cypproj3i)
```



Figure 10.2: Ahistorical (a) vs. historical (b) deterministic projection


Figure 10.3: Historical projection allowing integer population size only

We will not set integeronly = TRUE in the rest of this chapter, mostly because we are interested predominantly in the overall patterns of population dynamics, and forcing the projection of whole individuals makes extinction more and more likely with smaller population sizes. However, users interested in absolute population numbers, in the impacts of individual fate, or in the impacts of demographic stochasticity should use this setting.

Before moving on to ordered / cyclical projections, let's look at another setting of function projection3(): the start vector. Function projection3() assumes an initial starting vector of one individual in each stage, age (if Leslie), stage pair (if historical), or age-stage (if age-by-stage). We can see this by taking a look at the first column of the core ahistorical projection.

```
cypproj2$projection[[1]][[1]][,1]
> [1][11
```

We see that we are starting with one individual of each stage. But what if we wished to use a different start vector? In that circumstance, we have two options. The first is to use the start_vec option to load a vector of starting numbers of each stage, stage pair, or age-stage. Here is an example in which we start with 1000 seeds and 100 first-year protocorms, and no individuals of any other stage.

```
cypproj2_1000 <- projection3(cypmean2, times = 10,
    start_vec = c(1000, 100, 0, 0, 0, 0, 0, 0, 0, 0, 0))
> Warning: This function takes annual matrices as input. This lefkoMat object appears to be a set of
summary(cypproj2_1000)
>
> The input lefkoProj object covers 1 population-patches.
> It includes 10 projected steps per replicate and 1 replicates.
> The number of replicates with population size above the threshold size of 1 is as in
> the following matrix, with pop-patches given by column and milepost times given by row:
> $milepost_sums
        11
> 1 1
> 3 1
> 6 1
> 8 1
> 11 1
>
> $extinction_times
> [1] NA
```

The start_vec approach is very useful, but it requires a vector as long as the number of rows in the projection matrix. This can make it very difficult to create the right start vector for a historical matrix, an age-by-stage matrix, or an IPM. For the latter situations, we can use the start_input() function together with the start_frame option. We use start_input() to create a lefkoStart object, which is a data frame that contains everything we need to know for the start vector. Let's see this approach in action, starting the population with $10003^{\text {rd }}$ year protocorms and 100 dormant adults.

```
c2m_sv <- start_input(cypmean2, stage2 = c("P3", "D"), value = c(1000, 100))
c2m_sv
> stage2 stage_id_2 stage1 stage_id_1 age2 row_num value
> 1 P3 - 4 NA NA NA - N N N 1000
> 2 D % % NA 
```

In the code above, we told R that we wished to start with $10003^{\text {rd }}$ year protocorms and 100 dormant adults. By not stating any further information, we implied zero individuals for all other stages. The result is this start frame, which is a data frame showing the stages with non-zero starting individuals.

Now that we have our start frame, let's use it in a projection.

```
cypproj2_s1000 <- projection3(cypmean2, times = 10, start_frame = c2m_sv)
> Warning: This function takes annual matrices as input. This lefkoMat object appears to be a set of
summary(cypproj2_s1000)
>
> The input lefkoProj object covers 1 population-patches.
> It includes 10 projected steps per replicate and 1 replicates.
> The number of replicates with population size above the threshold size of 1 is as in
> the following matrix, with pop-patches given by column and milepost times given by row:
> $milepost_sums
> 1 1
> 1 1
> 3 1
> 6 1
> 8 1
> 11 1
>
> $extinction_times
> [1] NA
```

We can compare these projections via plots (figure 10.4).

```
par(mfrow = c(1, 3))
plot(cypproj2)
title("a)", adj = 0)
plot(cypproj2_1000, ylab = "")
title("b)", adj = 0)
plot(cypproj2_s1000, ylab = "")
title("c)", adj = 0)
```

The impact of a different start vector can be quite profound. The leftmost plot shows a population with a single individual of each stage, and so the population is able to grow right away. In contrast, the center and rightmost plots show populations that initially contain no reproductive individuals. In the center plot, it takes a few years for the dormant seeds and $1^{\text {st }}$ year protocorms to mature and the population to start growing. In the rightmost plot, the population starts growing once some of the dormant individuals become reproductive, which takes one or two years.

### 10.2.2 Ordered / cyclical projections of a specific time length

Situations may arise in which we wish to project an MPM with a specific order of matrices in mind. Typically these situations arise when a specific order of annual matrices is meant to correspond to the assumptions of predicted conditions. For example, we might have several years worth of matrices with some under high rain conditions and other under drought. We may believe due to climate forecasts that the next ten years should be relatively rainy while the ten years after that should be drier, and may wish to project our matrices forward in an order that matches these expectation. Or we may have


Figure 10.4: Using different start frames in projection. Projections started with (a) 1 of each stage, (b) 1000 dormant seeds and 100 seedlings set via start_vec, and (c) 10003 rd year protocorms and 100 dormant adults set via start_input
a situation in which a hurricane has affected our population, and we wish to project the population forward avoiding matrices in the immediate aftermath of the hurricane, since those matrices likely reflect dramatic changes as the population adjusts to the mass mortality and other issues that might have destabilized population dynamics during the hurricane itself.

Let's project the population forward, but this time let's force the projection to cycle through the matrices for 2007 and 2008 for ten years, and then to cycle between 2004 and 2005 for another ten years (2004 and 2005 were relatively dry years, while 2007 and 2008 were relatively wet years). Our projection will be 20 years in total.

```
year_order <- c(2007, 2008, 2007, 2008, 2007, 2008, 2007, 2008, 2007, 2008,
    2004, 2005, 2004, 2005, 2004, 2005, 2004, 2005, 2004, 2005)
cypproj2_ord1 <- projection3(cypmatrix2r, year = year_order, times = 20,
    start_frame = c2m_sv)
summary(cypproj2_ord1)
>
> The input lefkoProj object covers 1 population-patches.
> It includes 20 projected steps per replicate and 1 replicates.
> The number of replicates with population size above the threshold size of 1 is as in
> the following matrix, with pop-patches given by column and milepost times given by row:
> $milepost_sums
> 1 1
> 1 1
> 6 1
> 11 1
> 16 1
> 21 1
>
> $extinction_times
> [1] NA
```

If we have an MPM with a set of annual matrices, we might wish to project the population forward cyclically through those matrices. If the number of matrices in the lefkoMat object is shorter than the number of time steps set, then the default behavior of lefko3's projection functions is to cycle
through the MPMs. For example, if we leave the times option out from the above projection, then lefko3 will assume the default of 10,000 time steps, and cycle through our 20 defined years over and over, as below.

```
cypproj2_ord2 <- projection3(cypmatrix2r, year = year_order,
    start_frame = c2m_sv)
summary(cypproj2_ord2)
>
> The input lefkoProj object covers 1 population-patches.
> It includes }10000\mathrm{ projected steps per replicate and 1 replicates.
> The number of replicates with population size above the threshold size of 1 is as in
> the following matrix, with pop-patches given by column and milepost times given by row:
> $milepost_sums
> 1 1
> 1 1
> 2501 1
> 5001 1
> 7501 1
> 10001 1
>
> $extinction_times
> [1] NA
```

Let's see plots of these projections (figure 10.5).

```
par(mfrow=c(1,2))
plot(cypproj2_ord1)
title("a)", adj = 0)
plot(cypproj2_ord2, ylab = "")
title("b)", adj = 0)
```

Of course, the scale of the y axis differs dramatically, because the time scale also differs dramatically. The population is generally increasing, and so reaches much greater numbers in 10,000 time steps than in 20 .

If the year option is not used and stochastic = FALSE, then the default behavior of this function is to cycle through all matrices in a lefkoMat object in order. Here, we will cycle through all of the matrices in the ahistorical and historical population-level MPMs for 100 time steps each.

```
cypproj2_c100 <- projection3(cypmatrix2r, times = 100)
cypproj3_c100 <- projection3(cypmatrix3r, times = 100)
summary(cypproj2_c100)
>
> The input lefkoProj object covers 1 population-patches.
> It includes }100\mathrm{ projected steps per replicate and 1 replicates.
> The number of replicates with population size above the threshold size of 1 is as in
> the following matrix, with pop-patches given by column and milepost times given by row:
> $milepost_sums
> 11
```



Figure 10.5: Ordered projections can be cyclical. (a) A 20 year ordered projection. (b) A 10,000 year projection cycling through 20 ordered years

```
> 1 1
> 26 1
> 51 1
> 76 1
> 101 1
>
> $extinction_times
> [1] NA
summary(cypproj3_c100)
>
> The input lefkoProj object covers 1 population-patches.
> It includes }100\mathrm{ projected steps per replicate and 1 replicates.
> The number of replicates with population size above the threshold size of 1 is as in
> the following matrix, with pop-patches given by column and milepost times given by row:
> $milepost_sums
> 1 1
> 1 1
> 26 1
> 51 1
> 76 1
> 101 1
>
> $extinction_times
> [1] NA
```

Now let's plot the projections for comparison (figure 10.6).

```
par(mfrow=c(1,2))
plot(cypproj2_c100)
```

```
title("a)", adj = 0)
plot(cypproj3_c100, ylab = "")
title("b)", adj = 0)
```



Figure 10.6: Cyclical ahistorical (a) and historical (b) projections
Here we see through comparison of the scales of the y axes that individual history appears to reduce the growth rate relative to ahistorical projection, although clearly there are years in which the population growth rate appears quite high in both cases. The cyclical nature of the projection is obvious in both plots.

We might also wish to see what happens among different patches. Let's try a projection with patch structure, here using cypmatrix3rp. This MPM contains four matrices each for three patches.

```
cypproj3p_c100 <- projection3(cypmatrix3rp, times = 100)
summary(cypproj3p_c100)
>
> The input lefkoProj object covers 4 population-patches.
> It includes }100\mathrm{ projected steps per replicate and 1 replicates.
> The number of replicates with population size above the threshold size of 1 is as in
> the following matrix, with pop-patches given by column and milepost times given by row:
> $milepost_sums
> 1 A 1 B 1 C 1 0
> 1 
> 26 1 1 1 1 1
> 51 1 1 1 1 1
> 76 1 1 1 1 1 
>101 0 1 1 1
> $extinction_times
> [1] NA
```

Function projection3() has projected the matrix sets from each patch and the patch-weighted population in an ordered cycle for 100 time steps each, with each patch assumed to be independent of every other patch. The summary shows us that patch A may go extinct some time between time step 76 and time step 100. The population-level projection is summarized in the the rightmost column of the milepost_sums, and shows no extinction. Let's plot these results.

```
par(mfrow=c(2,2))
plot(cypproj3p_c100, patch = "all")
```



Figure 10.7: Patch-level projections for patch A (top left), patch B (top right), patch C (bottom left), and the patch-weighted population (bottom right)

The plots above are produced in the order of the labels element in the lefkoProj object, which we see below. The plots are propagated by row, meaning that patch A is in the top-left, patch B is in the top-right, patch C is in the bottom-left, and the patch-weighted population mean is in the bottomright, as shown in the labels element below. These results differ from the population-level projection because the population-level results from a patch-stratified MPM involve averaging the patches equally, whereas a population-level projection weights individual transitions as equal regardless of source patch.

```
cypproj3p_c100$labels
> pop patch
> 1 1 A
> 2 1 B
> 3 1 C
>4 1 0
```


### 10.2.3 Replicated simulations assuming temporal stochasticity

Let's now turn our attention to stochastic simulations. Package lefko3 currently implements temporal environmental stochasticity through its two projection functions. In projection3(), this means that the matrices in a MPM can be shuffled. Let's conduct a simple population-level stochastic simulation with 100 replicates and 10,000 time steps (note that 10,000 time steps is the default). We will then summarize the results.

```
set.seed(42)
cypstoch2 <- projection3(cypmatrix2r, nreps = 100, stochastic = TRUE)
summary(cypstoch2)
>
> The input lefkoProj object covers 1 population-patches.
> It includes }10000\mathrm{ projected steps per replicate and }100\mathrm{ replicates.
> The number of replicates with population size above the threshold size of 1 is as in
> the following matrix, with pop-patches given by column and milepost times given by row:
> $milepost_sums
> 1 1
> 1 100
> 2501 100
> 5001 100
> 7501 100
> 10001 0
>
> $extinction_times
> [1] NA
```

The summary shows us that we have one population-patch covered in this projection, and that it includes 10,000 time steps and 100 replicates. The final portion of the summary shows the number of replicates yielding a population size above a threshold value at particular milepost times in the projection. The default extinction threshold population value is 1.0 , and the default milepost times are the times once $0 \%, 25 \%, 50 \%, 75 \%$, and $100 \%$ of the projection has been run. We see that extinction may have occurred in every replicate some time after time step 7,501 . However, this is unlikely, as we know from the cyclical analysis that the ahistorical MPM suggests a growing population in the long-term. Let's take a look at what might be going on by examining the pop_size elements in the first ten replicates right after time step 7,500.

| cypstoch2\$pop_size[[1] $][c(1: 10), c(7501: 7503)]$ |  |  |  |  |
| :--- | ---: | ---: | ---: | ---: |
| $>$ |  | $[, 1]$ | $[, 2]$ | $[, 3]$ |
| $>$ | $[1]$, | $8.321329 \mathrm{e}+252$ | $3.753757 \mathrm{e}+252$ | $1.269240 \mathrm{e}+253$ |
| $>$ | $[2]$, | $1.612221 \mathrm{e}+253$ | $2.310869 \mathrm{e}+253$ | $1.539012 \mathrm{e}+253$ |
| $>$ | $[3]$, | $3.266403 \mathrm{e}+257$ | $3.040670 \mathrm{e}+257$ | $3.963822 \mathrm{e}+257$ |
| $>$ | $[4]$, | $4.595511 \mathrm{e}+257$ | $3.245229 \mathrm{e}+257$ | $4.734050 \mathrm{e}+257$ |
| $>$ | $[5]$, | $1.276522 \mathrm{e}+256$ | $1.461003 \mathrm{e}+256$ | $6.718161 \mathrm{e}+255$ |
| $>$ | $[6]$, | $7.869332 \mathrm{e}+253$ | $3.224921 \mathrm{e}+253$ | $2.112759 \mathrm{e}+254$ |
| $>$ | $[7]$, | $3.699333 \mathrm{e}+256$ | $5.988414 \mathrm{e}+256$ | $1.572965 \mathrm{e}+257$ |
| $>$ | $[8]$, | $1.049118 \mathrm{e}+254$ | $1.465183 \mathrm{e}+254$ | $1.135257 \mathrm{e}+254$ |
| $>$ | $[9]$, | $1.265153 \mathrm{e}+258$ | $5.958920 \mathrm{e}+257$ | $6.857992 \mathrm{e}+257$ |
| $>$ | $[10]$, | $1.443963 \mathrm{e}+259$ | $1.299965 \mathrm{e}+259$ | $3.667100 \mathrm{e}+258$ |

We see that the population has exploded in size, and so the zeros in the milepost section of the summary occur because the population sizes projected became too high for our computers to handle
(to see the highest number your computer can handle, type .Machine\$double.xmax at the prompt). To see this, let's view a plot (note the ylim values - exceeding the maximum allowed number in our computers means that we need to designate appropriate limits to the $y$-axis or else the plot () function will produce an error).

```
plot(cypstoch2, ylim = c(0, 10000000000000000000000000000000000000000000000000))
```



Figure 10.8: Stochastic projection with exploding population size
How does this compare to a historical projection? Let's limit the number of time steps to 1000 to keep this quick and then take a look at a summary.

```
cypstoch3 <- projection3(cypmatrix3r, nreps = 100, times = 1000,
    stochastic = TRUE)
summary(cypstoch3)
>
> The input lefkoProj object covers 1 population-patches.
> It includes }1000\mathrm{ projected steps per replicate and }100\mathrm{ replicates.
> The number of replicates with population size above the threshold size of 1 is as in
> the following matrix, with pop-patches given by column and milepost times given by row:
> $milepost_sums
> 1 1
> 1 100
> 251 100
> 501 100
> 751 100
> 1001 100
>
> $extinction_times
> [1] NA
```

We see neither extinction, nor computationally problematic population growth. Let's plot this projection against the preceding projection, but standardizing the scales of the x and y axes to compare properly (figure 10.9).

```
par(mfrow=c(1, 2))
plot(cypstoch2, xlim = c(0, 1001), ylim = c(0, 35000000000))
title("a)", adj = 0)
plot(cypstoch3, ylim = c(0, 35000000000), ylab = "")
title("b)", adj = 0)
```



Figure 10.9: Stochastic ahistorical (a) vs. historical (b) projection
All of our replicates survived and increased in the historical case, but more moderately than in the ahistorical case.

Let's try another projection using the patch-level historical MPM. We will also set growthonly = FALSE to allow the stage structure and reproductive values to be tracked.

```
cypstoch2p <- projection3(cypmatrix2rp, stochastic = TRUE, nreps = 100,
    times = 200, growthonly = FALSE)
summary(cypstoch2p)
>
> The input lefkoProj object covers 4 population-patches.
> It includes }200\mathrm{ projected steps per replicate and }100\mathrm{ replicates.
> The number of replicates with population size above the threshold size of 1 is as in
> the following matrix, with pop-patches given by column and milepost times given by row:
> $milepost_sums
> 1 A 1 B 1 C 1 0
> 1 100 100 100 100
> 51 96 100 100 100
> 101 82 100 100 100
> 151
> 201 68 100 100 100
>
> $extinction_times
```

> [1] NA
Our summary shows us that we have three patches and the full (patch-weighted) population projected. There are 100 replicates of each. We can call each patch using the first set of double brackets for the projection, stage_dist, rep_value, and pop_size elements of the resulting lefkoProj object, and each replicate within each patch using the second set of brackets. Here, we will take a look at the overall stage distribution at the population level in the first replicate at the 100th time by focusing on the first element within the fourth list, and the 100 th column of that element. If we wanted to see the equivalent stage distribution for the 100th replicate, we would adjust the second set of brackets accordingly.

```
writeLines("Replicate 1:")
> Replicate 1:
cypstoch2p$stage_dist[[4]][[1]][,100]
> [1] 4.669882e-01 4.774991e-01 5.292660e-02 1.939510e-03 2.011212e-04
> [6] 2.086144e-05 1.942072e-04 1.782106e-04 5.180013e-05 3.739271e-07
> [11] 0.000000e+00
writeLines("\nReplicate 100:")
>
> Replicate 100:
cypstoch2p$stage_dist[[4]][[100]][,100]
> [1] 4.778352e-01 4.847863e-01 3.508813e-02 1.678213e-03 5.298717e-05
> [6] 2.896109e-05 2.657260e-04 2.070924e-04 5.473344e-05 2.624975e-06
> [11] 0.000000e+00
```

Note that although these are different distributions, they are nonetheless extremely similar. This reflects the strong stochastic ergodic theorem, which predicts that that the stage structure should converge to a stationary distribution in a stochastic environment (Caswell, 2001).

Let's conduct one more projection, this time as a simulation study of the historical patch-level MPM with 100 replicates and 1000 time steps each. Note that this might take a little while to run, and will result in an object a little over 1 gigabyte in size.

```
cypstoch3p <- projection3(cypmatrix3rp, nreps = 100, times = 1000,
    stochastic = TRUE, growthonly = FALSE)
```

Let's see a summary.

```
summary(cypstoch3p)
>
> The input lefkoProj object covers 4 population-patches.
> It includes 1000 projected steps per replicate and 100 replicates.
> The number of replicates with population size above the threshold size of 1 is as in
> the following matrix, with pop-patches given by column and milepost times given by row:
> $milepost_sums
> 1 A 1 B 1 C 1 0
> 1 100 100 100 100
> 251 0
>501 0 0 0 0
>751 0 0 0 0
>1001 0 0 0 0
>
> $extinction_times
> [1] NA
```

The new output shows us how different predictions can be between historical vs. ahistorical MPMs, as well as between population-level and patch-level MPMs. The ahistorical population-level MPMs showed explosively growing population size, while the historical version showed moderate, fluctuating growth, often still dropping quite low. When run with patch-level MPMs, extinction was predicted either way. To see how quickly extinction happens, let's change the milepost settings in the summary.lefkoProj() function, as below.

```
summary(cypstoch3p, milepost= c(1, 10, 20, 25, 30, 50, 100, 150, 200, 250, 300))
>
> The input lefkoProj object covers 4 population-patches.
> It includes }1000\mathrm{ projected steps per replicate and 100 replicates.
> The number of replicates with population size above the threshold size of 1 is as in
> the following matrix, with pop-patches given by column and milepost times given by row:
> $milepost_sums
> 1 A 1 B 1 C 1 0
> 1 100}10010010010
> 10 95 100 100 100
>20 70 100 100 100
> 25 55 100 100 100
> 30 55 100 100 100
> 50 16 100 99 98
> 100
> 150}00024 48 0
> 200 0 6 6 28 0
> 250 0 1 8 8 0
> 300 0 0 4 0
>
> $extinction_times
> [1] NA
```

Let's zoom in on the historical population-level's 100th replicate, and take a peek at the first 100 population sizes.

```
cypstoch3p$pop_size[[4]][100,c(1:100)]
> [1] 121.000000 46179.502222 29023.695555 22837.294912
> [6] 9081.579148 11978.167039 3320.958945 3808.661359 2153.897008
> [11] 10310.177333 74497.822071 5409.348722 
> [16] 7203.782417 2318.704951 1498.582135 5040.397381 1737.632999
> [[21] 2319.786969 2434.495892 1008.608147 454.399196 
> [[26] 3583.433577 850.056523 680.232056 697.419729 581.051819
> [31] 2399.858548 1797.896250 1292.585451 
> [36] 423.705934 483.055480 822.205618 588.699106 207.467436
> [41] 170.084749 152.944574 219.276019 1011.397505 361.317706
> [46] 967.865891 801.997131 237.946495 92.049747 358.774818
> [51] 472.099252 154.537674 66.314238 
> [56] 108.581301 [llllll
> [61] 55.878980 [llllll
> [66] [llllllll
> [l71] 9.064451 
> [ll6] 3.822600 [ 5.375812 
> [81] [llllll
> [llllllll
```

| $>$ | $[91]$ | 1.422278 | 6.759237 | 5.958552 | 2.370643 | 1.542012 |
| :--- | :--- | :--- | :--- | :--- | :--- | :--- |
| $>$ | $[96]$ | 1.378010 | 3.707283 | 5.180411 | 3.811777 | 1.373851 |

Each patch or population's pop_size element is a matrix with rows equal to the number of replicates and columns equal to the number of times projected. In contrast, the projection, stage_dist, and rep_value elements are lists nested within lists, where the top level is the population-patch, and the lower level is the replicate. Within the replicate, there is a matrix with rows corresponding to the stages or stage-pairs, and the columns corresponding to the number of times projected. Let's take a look at the first five time steps in the 100th replicate of the population level, for each of these three elements.

| cypstoch3p\$projection[[4]][[100]][,c(1:5)] |  |  |  |  |  |  |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| > |  | [,1] | [,2] | [,3] | [,4] | [,5] |
| > | [1,] | 1 | 0.5800000 | $2.308247 \mathrm{e}+03$ | $1.289094 \mathrm{e}+03$ | $1.035021 \mathrm{e}+03$ |
| > | [2,] | 1 | 0.6000000 | $2.308259 \mathrm{e}+03$ | $1.335259 \mathrm{e}+03$ | $1.060803 \mathrm{e}+03$ |
| > | [3,] | 1 | 0.0000000 | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |
| > | [4, ] | 1 | 0.0000000 | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |
| > | [5, ] | 1 | 0.0000000 | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |
| $>$ | [6, ] | 1 | 0.0000000 | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |
| > | [7,] | 1 | 0.0000000 | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |
| > | [8,] | 1 | 0.0000000 | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |
| $>$ | [9,] | 1 | 0.0000000 | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |
| > | [10,] | 1 | 0.0000000 | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |
| > | [11,] | 1 | 0.0000000 | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |
| > | [12,] | 1 | 0.0000000 | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |
| > | [13,] | 1 | 0.0000000 | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |
| > | [14,] | 1 | 0.6000000 | $2.308261 \mathrm{e}+03$ | $1.335260 \mathrm{e}+03$ | $1.065419 \mathrm{e}+03$ |
| > | [15,] | 1 | 0.0000000 | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |
| > | [16,] | 1 | 0.0000000 | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |
| > | [17,] | 1 | 0.0000000 | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |
| > | [18,] | 1 | 0.0000000 | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |
| > | [19,] | 1 | 0.0000000 | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |
| > | [20,] | 1 | 0.0000000 | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |
| > | [21,] | 1 | 0.0000000 | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |
| > | [22,] | 1 | 0.0000000 | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |
| > | [23,] | 1 | 0.0000000 | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |
| > | [24,] | 1 | 0.0000000 | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |
| > | [25,] | 1 | 0.0000000 | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |
| > | [26,] | 1 | 0.1000000 | 6.000000e-02 | $2.308261 \mathrm{e}+02$ | $1.335260 \mathrm{e}+02$ |
| > | [27,] | 1 | 0.0000000 | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |
| $>$ | [28,] | 1 | 0.0000000 | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |
| > | [29,] | 1 | 0.0000000 | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |
| $>$ | [30,] | 1 | 0.0000000 | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |
| $>$ | [31,] | 1 | 0.0000000 | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |
| > | [32,] | 1 | 0.0000000 | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |
| > | [33,] | 1 | 0.0000000 | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |
| > | [34,] | 1 | 0.0000000 | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |
| > | [35,] | 1 | 0.0000000 | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |
| > | [36,] | 1 | 0.0000000 | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |
| > | [37,] | 1 | 0.0000000 | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |
| > | [38,] | 1 | 0.0500000 | $5.000000 \mathrm{e}-03$ | $3.000000 \mathrm{e}-03$ | $1.154131 \mathrm{e}+01$ |
|  | [39,] | 1 | 0.0000000 | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |


|  |  |  |  |  |  |  |
| :--- | :--- | :--- | ---: | :--- | :--- | :--- | :--- |
| $>$ | $[40]$, | 1 | 0.0000000 | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |
| $>$ | $[41]$, | 1 | 0.0000000 | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |
| $>$ | $[42]$, | 1 | 0.0000000 | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |
| $>$ | $[43]$, | 1 | 0.0000000 | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |
| $>$ | $[44]$, | 1 | 0.0000000 | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |
| $>$ | $[45]$, | 1 | 0.0000000 | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |
| $>$ | $[46]$, | 1 | 0.0000000 | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |
| $>$ | $[47]$, | 1 | 0.0000000 | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |
| $>$ | $[48]$, | 1 | 0.0000000 | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |
| $>$ | $[49]$, | 1 | 0.1000000 | $7.500000 \mathrm{e}-03$ | $6.250000 \mathrm{e}-04$ | $1.812500 \mathrm{e}-04$ |
| $>$ | $[50]$, | 1 | 0.1444444 | $1.083333 \mathrm{e}-02$ | $9.027778 \mathrm{e}-04$ | $0.000000 \mathrm{e}+00$ |
| $>$ | $[51]$, | 1 | 1.0333333 | $7.750000 \mathrm{e}-02$ | $6.458333 \mathrm{e}-03$ | $1.127778 \mathrm{e}-03$ |
| $>$ | $[52]$, | 1 | 0.1555556 | $1.166667 \mathrm{e}-02$ | $9.722222 \mathrm{e}-04$ | $1.409722 \mathrm{e}-03$ |
| $>$ | $[53]$, | 1 | 0.0000000 | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |
| $>$ | $[54]$, | 1 | 0.0000000 | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |
| $>$ | $[55]$, | 1 | 0.0000000 | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |
| $>$ | $[56]$, | 1 | 0.0000000 | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |
| $>$ | $[57]$, | 1 | 0.0000000 | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |
| $>$ | $[58]$, | 1 | 0.0000000 | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |
| $>$ | $[59]$, | 1 | 0.0000000 | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |
| $>$ | $[60]$, | 1 | 0.0000000 | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |
| $>$ | $[61]$, | 1 | 0.0000000 | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |
| $>$ | $[62]$, | 1 | 0.0000000 | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $3.316353 \mathrm{e}-01$ |
| $>$ | $[63]$, | 1 | 0.6666667 | $1.169312 \mathrm{e}-01$ | $1.157111 \mathrm{e}-01$ | $1.187400 \mathrm{e}-01$ |
| $>$ | $[64]$, | 1 | 0.0000000 | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |
| $>$ | $[65]$, | 1 | 0.0000000 | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |
| $>$ | $>[89]$, | 1 | 7083.3333333 | $7.152778 \mathrm{e}+03$ | $6.631944 \mathrm{e}+03$ | $1.859568 \mathrm{e}+03$ |
| $>$ | $[90]$, | 1 | 7083.3333333 | $7.152778 \mathrm{e}+03$ | $6.631944 \mathrm{e}+03$ | $1.859568 \mathrm{e}+03$ |


| $>$ [91,] | 1 | 0.0000000 | $0.000000+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |
| :---: | :---: | :---: | :---: | :---: | :---: |
| $>$ [92,] | 1 | 0.0000000 | $0.000000+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |
| $>$ [93,] | 1 | 0.0000000 | $0.000000+00$ | $0.000000 \mathrm{e}+00$ | $0.000000+00$ |
| $>$ [94,] | 1 | 0.0000000 | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |
| $>$ [95,] | 1 | 0.1666667 | 8.333333e-02 | $4.629630 \mathrm{e}-02$ | $0.000000 \mathrm{e}+00$ |
| $>$ [96,] | 1 | 1.0000000 | 6.666667e-01 | $3.657407 \mathrm{e}-01$ | $0.000000 \mathrm{e}+00$ |
| $>$ [97,] | 1 | 1.1666667 | $1.138889 \mathrm{e}+00$ | $1.041667 \mathrm{e}+00$ | $7.932099 \mathrm{e}-01$ |
| $>$ [98,] | 1 | 0.0000000 | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $3.472222 \mathrm{e}-01$ |
| > [99,] | 1 | 0.0000000 | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |
| > [100,] | 1 | 5833.3333333 | $5.555556 \mathrm{e}+02$ | $0.000000 \mathrm{e}+00$ | $0.000000+00$ |
| > [101,] | 1 | 5833.3333333 | $5.555556 \mathrm{e}+02$ | $0.000000 \mathrm{e}+00$ | 0.000000e+00 |
| $>$ [102,] | 1 | 0.0000000 | $0.000000+00$ | $0.000000 \mathrm{e}+00$ | $0.000000+00$ |
| > [103,] | 1 | 0.0000000 | $0.000000+00$ | $0.000000 \mathrm{e}+00$ | $0.000000+00$ |
| > [104,] | 1 | 0.0000000 | $0.000000+00$ | $0.000000 \mathrm{e}+00$ | $0.000000+00$ |
| > [105,] | 1 | 0.0000000 | $0.000000+00$ | $0.000000 \mathrm{e}+00$ | $0.000000+00$ |
| > [106, ] | 1 | 0.0000000 | $0.000000+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |
| > [107,] | 1 | 0.3333333 | $1.111111 \mathrm{e}-01$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |
| > [108,] | 1 | 0.6666667 | $1.111111 \mathrm{e}-01$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |
| > [109,] | 1 | 0.3333333 | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |
| > [110,] | 1 | 0.0000000 | $0.000000+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |
| > [111, ] | 1 | 7500.0000000 | $5.555556 \mathrm{e}+02$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |
| $>$ [112,] | 1 | 7500.0000000 | $5.555556 \mathrm{e}+02$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |
| $>$ [113,] | 1 | 0.0000000 | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |
| $>$ [114,] | 1 | 0.0000000 | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |
| > [115,] | 1 | 0.0000000 | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |
| > [116,] | 1 | 0.0000000 | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |
| > [117,] | 1 | 0.0000000 | $0.000000+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |
| > [118,] | 1 | 0.0000000 | $0.000000+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |
| > [119,] | 1 | 0.0000000 | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |
| > [120,] | 1 | 0.3333333 | $1.111111 \mathrm{e}-01$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |
| > [121,] | 1 | 0.3333333 | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |
| cypstoch3p\$stage_dist[[4]][[100]][,c(1:5)] |  |  |  |  |  |

$>$ [19,] $0.0082644630 .000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>[20] 0.0082644630 .000000 \mathrm{e}+,000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>[21] 0.0082644630 .000000 \mathrm{e}+,000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [22,] $0.0082644630 .000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [23,] $0.0082644630 .000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [24,] 0.008264463 0.000000e+00 0.000000 e+00 0.000000e+00 0.000000e+00
$>$ [25,] $0.0082644630 .000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
> [26,] 0.008264463 2.165463e-06 2.067276e-06 1.010742e-02 1.746992e-02
$>$ [27,] $0.0082644630 .000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [28,] 0.008264463 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00
$>$ [29,] $0.0082644630 .000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [30,] $0.0082644630 .000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [31,] $0.0082644630 .000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>[32] 0.0082644630 .000000 \mathrm{e}+,000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>[33] 0.0082644630 .000000 \mathrm{e}+,000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [34,] 0.008264463 0.000000e+00 0.000000 $+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [35,] 0.008264463 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00
$>$ [36,] 0.008264463 $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [37,] $0.0082644630 .000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [38,] $0.0082644631 .082731 \mathrm{e}-061.722730 \mathrm{e}-071.313641 \mathrm{e}-071.510010 \mathrm{e}-03$
$>$ [39,] $0.0082644630 .000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [40,] $0.0082644630 .000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [41,] $0.0082644630 .000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [42,] 0.008264463 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00
$>$ [43,] $0.0082644630 .000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>[44] 0.0082644630 .000000 \mathrm{e}+,000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [45,] $0.0082644630 .000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [46,] 0.008264463 0.000000e+00 0.000000 $+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [47,] 0.008264463 0.000000e+00 0.000000 $+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [48,] $0.0082644630 .000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [49,] 0.008264463 2.165463e-06 2.584095e-07 2.736751e-08 2.371390e-08
$>[50] 0.0082644633 .127891 \mathrm{e}-,063.732582 \mathrm{e}-073.953085 \mathrm{e}-080.000000 \mathrm{e}+00$
$>$ [51,] 0.008264463 2.237645e-05 2.670232e-06 2.827976e-07 1.475531e-07
$>\quad[52] 0.0082644633 .368498 \mathrm{e}-,064.019704 \mathrm{e}-074.257169 \mathrm{e}-081.844414 \mathrm{e}-07$
$>$ [53,] $0.0082644630 .000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [54,] 0.008264463 $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [55,] $0.0082644630 .000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>[56] 0.0082644630 .000000 \mathrm{e}+,000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [57,] 0.008264463 $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [58,] 0.008264463 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00
$>$ [59,] 0.008264463 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00
$>$ [60,] 0.008264463 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00
$>[61] 0.0082644630 .000000 \mathrm{e}+,000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [62,] $0.0082644630 .000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+004.338960 \mathrm{e}-05$
$>[63] 0.0082644631 .443642 \mathrm{e}-,054.028819 \mathrm{e}-06 \quad 5.066762 \mathrm{e}-061.553538 \mathrm{e}-05$
$>$ [64,] 0.008264463 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00
$>$ [65,] $0.0082644630 .000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>[66] 0.0082644630 .000000 \mathrm{e}+,000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [67,] 0.008264463 2.577932e-03 9.131252e-03 1.630614e-02 0.000000e+00
$>[68] 0.0082644632 .577932 \mathrm{e}-,039.131252 \mathrm{e}-031.630614 \mathrm{e}-02 \quad 0.000000 \mathrm{e}+00$
$>$ [69,] $0.0082644630 .000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>[70] 0.0082644630 .000000 \mathrm{e}+,000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [71,] $0.0082644630 .000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [72,] $0.0082644634 .468416 \mathrm{e}-06 \quad 1.158709 \mathrm{e}-05 \quad 1.448212 \mathrm{e}-05 \quad 0.000000 \mathrm{e}+00$
$>$ [73,] 0.008264463 4.820733e-05 1.077761e-04 1.380859e-04 2.632668e-04
$>$ [74,] 0.008264463 8.421245e-06 1.800815e-05 2.325904e-05 2.759903e-04
$>$ [75,] 0.008264463 $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [76,] 0.008264463 $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [77,] 0.008264463 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00
$>$ [78,] $0.0082644635 .513910 \mathrm{e}-028.666822 \mathrm{e}-021.013519 \mathrm{e}-013.987650 \mathrm{e}-02$
$>$ [79,] $0.0082644635 .513910 \mathrm{e}-028.666822 \mathrm{e}-021.013519 \mathrm{e}-013.987650 \mathrm{e}-02$
$>$ [80,] 0.008264463 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00
$>$ [81,] $0.0082644630 .000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [82,] $0.0082644630 .000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [83,] $0.0082644631 .443642 \mathrm{e}-056.252873 \mathrm{e}-06 \quad 7.799082 \mathrm{e}-06 \quad 0.000000 \mathrm{e}+00$
$>$ [84,] 0.008264463 2.085261e-05 3.617734e-05 4.446395e-05 4.641563e-06
$>$ [85,] 0.008264463 4.691836e-05 7.692735e-05 8.960614e-05 2.416799e-04
$>$ [86,] 0.008264463 1.082731e-05 9.570724e-06 6.487115e-06 6.057333e-05
$>$ [87,] $0.0082644630 .000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+001.595060 \mathrm{e}-05$
$>$ [88,] 0.008264463 $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [89,] $0.0082644631 .533870 \mathrm{e}-012.464461 \mathrm{e}-012.903997 \mathrm{e}-012.432971 \mathrm{e}-01$
$>[90] 0.0082644631 .533870 \mathrm{e}-,012.464461 \mathrm{e}-012.903997 \mathrm{e}-012.432971 \mathrm{e}-01$
$>$ [91,] $0.0082644630 .000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [92,] $0.0082644630 .000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [93,] $0.0082644630 .000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [94,] $0.0082644630 .000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [95,] $0.0082644633 .609105 \mathrm{e}-062.871217 \mathrm{e}-06 \quad 2.027223 \mathrm{e}-06 \quad 0.000000 \mathrm{e}+00$
$>$ [96,] $0.0082644632 .165463 \mathrm{e}-052.296974 \mathrm{e}-051.601506 \mathrm{e}-050.000000 \mathrm{e}+00$
$>$ [97,] 0.008264463 2.526373e-05 3.923997e-05 4.561252e-05 1.037799e-04
$>$ [98,] $0.0082644630 .000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+004.542892 \mathrm{e}-05$
$>$ [99,] $0.0082644630 .000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
[100,] $0.0082644631 .263187 \mathrm{e}-011.914145 \mathrm{e}-020.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
[101,] $0.0082644631 .263187 \mathrm{e}-011.914145 \mathrm{e}-020.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
[102,] $0.0082644630 .000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
[103,] $0.0082644630 .000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [104,] $0.0082644630 .000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
> [105,] 0.008264463 $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>[106] 0.0082644630 .000000 \mathrm{e}+,000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [107,] $0.0082644637 .218210 \mathrm{e}-063.828290 \mathrm{e}-06 \quad 0.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>[108] 0.0082644631 .443642 \mathrm{e}-,053.828290 \mathrm{e}-06 \quad 0.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [109,] $0.0082644637 .218210 \mathrm{e}-060.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
> [110,] $0.0082644630 .000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
> [111,] $0.0082644631 .624097 \mathrm{e}-011.914145 \mathrm{e}-020.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
> [112,] 0.008264463 1.624097e-01 1.914145e-02 0.000000e+00 0.000000e+00
> [113,] $0.0082644630 .000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
> [114,] $0.0082644630 .000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>[115] 0.0082644630 .000000 \mathrm{e}+,000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
> [116,] $0.0082644630 .000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
> [117,] $0.0082644630 .000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>[118] 0.0082644630 .000000 \mathrm{e}+,000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>[119] 0.0082644630 .000000 \mathrm{e}+,000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>[120] 0.0082644637 .218210 \mathrm{e}-,063.828290 \mathrm{e}-06 \quad 0.000000 \mathrm{e}+000.000000 \mathrm{e}+00$

```
> [121,] 0.008264463 7.218210e-06 0.000000e+00 0.000000e+00 0.000000e+00
cypstoch3p$rep_value[[4]][[100]][,c(1:5)]
\(>\) [,1] [,2] [,3] [,4] 5]
```

$>\quad[1] 9.883159 \mathrm{e}-,079.885868 \mathrm{e}-071.106955 \mathrm{e}-06 \quad 2.096488 \mathrm{e}-062.389431 \mathrm{e}-06$
$>\quad[2] 7.341332 \mathrm{e}-,069.215603 \mathrm{e}-069.155864 \mathrm{e}-069.523130 \mathrm{e}-061.897771 \mathrm{e}-05$
$>\quad[3] 0.000000 \mathrm{e}+,000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>\quad[4] 0.000000 \mathrm{e}+,000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>\quad[5] 0.000000 \mathrm{e}+,000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>\quad[6] 0.000000 \mathrm{e}+,000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>\quad[7] 0.000000 \mathrm{e}+,000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>\quad[8] 0.000000 \mathrm{e}+,000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>\quad[9] 0.000000 \mathrm{e}+,000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [10,] $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>[11] 0.000000 \mathrm{e}+,000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [12,] $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [13,] $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [14,] 6.515367e-05 7.432931e-05 9.360616e-05 9.264027e-05 9.488778e-05
$>$ [15,] $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [16,] $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [17,] $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [18,] $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>[19] 0.000000 \mathrm{e}+,000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [20,] $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [21,] $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>[22] 0.000000 \mathrm{e}+,000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [23,] $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [24,] $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [25,] $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [26,] 6.644859e-04 6.596660e-04 7.549893e-04 9.471198e-04 9.230610e-04
$>$ [27,] $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [28,] $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [29,] $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [30,] $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [31,] $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>[32] 0.000000 \mathrm{e}+,000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [33,] $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [34,] $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [35,] $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [36,] $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [37,] $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [38,] 1.359207e-02 1.345554e-02 1.340092e-02 1.527817e-02 1.887407e-02
$>$ [39,] $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [40,] $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [41,] $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>[42] 0.000000 \mathrm{e}+,000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>[43] 0.000000 \mathrm{e}+,000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [44,] $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [45,] $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [46,] $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [47,] $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [48,] $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [49,] 1.359207e-02 1.345554e-02 1.340092e-02 1.527817e-02 1.887407e-02
$>\quad[50] 0.000000 \mathrm{e}+,000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [51,] 1.996688e-02 1.997815e-02 1.806562e-02 2.047292e-02 2.813370e-02
$>$ [52,] 3.024501e-02 3.557370e-02 4.709978e-02 2.851265e-02 1.421137e-02
$>$ [53,] $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [54,] 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00 $0.000000 \mathrm{e}+00$
$>\quad[55] 0.000000 \mathrm{e}+,000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [56,] $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [57,] $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [58,] $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [59,] $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [60,] $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [61,] $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [62,] $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+009.411851 \mathrm{e}-030.000000 \mathrm{e}+00$
$>\quad[63] 0.000000 \mathrm{e}+,000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+009.604091 \mathrm{e}-03$
$>$ [64,] $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [65,] $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+001.589258 \mathrm{e}-020.000000 \mathrm{e}+00$
$>$ [66,] $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [67,] 1.007844e-06 1.010383e-06 1.148395e-06 2.144449e-06 2.414708e-06
$>$ [68,] 7.341332e-06 9.215603e-06 9.155864e-06 9.523130e-06 1.897771e-05
$>$ [69,] $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>[70] 0.000000 \mathrm{e}+,000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [71,] $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [72,] 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00
$>$ [73,] 2.116926e-02 2.118584e-02 1.943840e-02 2.047292e-02 2.813370e-02
$>$ [74,] 3.585611e-02 4.120960e-02 5.350609e-02 2.851265e-02 1.421137e-02
$>$ [75,] $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+009.630101 \mathrm{e}-020.000000 \mathrm{e}+00$
$>$ [76,] 1.117750e-02 4.104774e-03 0.000000e+00 0.000000e+00 $0.000000 \mathrm{e}+00$
$>$ [77,] $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [78,] 1.007844e-06 1.010383e-06 1.148395e-06 2.144449e-06 2.414708e-06
$>$ [79,] 7.341332e-06 9.215603e-06 9.155864e-06 9.523130e-06 1.897771e-05
$>$ [80,] $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [81,] $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [82,] $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [83,] $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+006.425907 \mathrm{e}-030.000000 \mathrm{e}+00$
$>$ [84,] 2.247720e-02 2.313248e-02 1.987204e-02 2.122501e-02 3.513143e-02
$>$ [85,] 5.012610e-02 5.733413e-02 7.136429e-02 9.179451e-02 9.775391e-02
$>\quad[86] \quad 1.460718 \mathrm{e}-,011.521127 \mathrm{e}-01 \quad 1.155540 \mathrm{e}-01 \quad 2.104936 \mathrm{e}-01 \quad 6.001083 \mathrm{e}-02$
$>\quad[87] 0.000000 \mathrm{e}+,000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [88,] $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [89,] 1.007844e-06 1.010383e-06 1.148395e-06 2.144449e-06 2.414708e-06
$>$ [90,] 7.341332e-06 9.215603e-06 9.155864e-06 9.523130e-06 1.897771e-05
$>$ [91,] $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [92,] $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [93,] $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [94,] $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>[95] 0.000000 \mathrm{e}+,000.000000 \mathrm{e}+000.000000 \mathrm{e}+001.416613 \mathrm{e}-024.052185 \mathrm{e}-02$
$>\quad[96] 7.080763 \mathrm{e}-,027.328743 \mathrm{e}-021.034554 \mathrm{e}-013.796758 \mathrm{e}-028.167958 \mathrm{e}-02$
$>$ [97,] 2.993882e-01 2.826221e-01 $2.568144 \mathrm{e}-012.283851 \mathrm{e}-012.878609 \mathrm{e}-01$
$>$ [98,] $5.753146 \mathrm{e}-025.211191 \mathrm{e}-025.917720 \mathrm{e}-020.000000 \mathrm{e}+000.000000 \mathrm{e}+00$
$>$ [99,] $0.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+000.000000 \mathrm{e}+00$

```
> [100,] 1.007844e-06 1.010383e-06 1.148395e-06 2.144449e-06 2.414708e-06
> [101,] 7.341332e-06 9.215603e-06 9.155864e-06 9.523130e-06 1.897771e-05
> [102,] 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00
> [103,] 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00
> [104,] 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00
> [105,] 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00
> [106,] 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00
> [107,] 5.849602e-02 4.637516e-02 7.895477e-02 3.270260e-02 0.000000e+00
> [108,] 2.412809e-02 3.395090e-02 1.250809e-02 0.000000e+00 1.083988e-01
> [109,] 4.327867e-02 4.692303e-02 2.999251e-02 3.378413e-02 4.750575e-02
> [110,] 6.448293e-02 6.548406e-02 6.726622e-02 7.181279e-02 0.000000e+00
> [111,] 1.007844e-06 1.010383e-06 1.148395e-06 2.144449e-06 2.414708e-06
> [112,] 7.341332e-06 9.215603e-06 9.155864e-06 9.523130e-06 1.897771e-05
> [113,] 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00
> [114,] 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00
> [115,] 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00
> [116,] 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00
> [117,] 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00
> [118,] 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00
> [119,] 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00
> [120,] 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00 2.735823e-02
> [121,] 1.683328e-02 1.690771e-02 1.921892e-02 0.000000e+00 8.059011e-02
```

This is a huge deal of data, and could be of great use to ecologists studying changes in population structure, natural selection, etc.

The output for our historical MPM suggests that the population will decline quite quickly. Let's see how quickly by plotting the first 100 projected time steps of each of the 100 replicates, as below (figure 10.10).

```
par(mfrow = c(2,2))
plot(cypstoch3p, xlim = c(1, 100), patch = "all")
```

We see very quick declines in patch A and the population as a whole, though patches B and C appear capable of surviving some time longer in some replicates.

Users running stochastic projections may wish to weight the probability of sampling each matrix differently. This can be done using the tweights option, which takes as its input a vector giving the respective weight of each matrix. Users may fill this vector with probabilities, or may use integers or other numbers to provide a relative weighting. In the latter case, function projection3() will calculate the probability as the vector element divided by the vector sum. Let's project our historical population-level MPM assuming that the last matrix is chosen $91 \%$ of the time while the other three matrices are chosen $3 \%$ of the time each.

```
cypstoch3_weights <- projection3(cypmatrix3r, nreps = 100, times = 200,
    stochastic = TRUE, tweights = c(0.03, 0.03, 0.03, 0.91))
```

Now let's plot the results, comparing them to the original projection with equal weighting (figure 10.11).

```
par(mfrow = c(1,2))
plot(cypstoch3)
```



Figure 10.10: Stochastic patch-level projections

```
title("a)", adj = 0)
plot(cypstoch3_weights, ylab = "")
title("b)", adj = 0)
```

Choosing the first matrix $91 \%$ of the time may lead to greater population growth, as we reach higher numbers more quickly under that assumption (note the difference in scale of the x axes).

### 10.2.4 Density dependent deterministic projections

Let's now move on to the simplest density dependent analysis: a single matrix projected forward with some elements altered at each time step according to some function of population size, $N$. We might conduct such an analysis if we have an inkling that density has some role in population dynamics. For example, we might suspect that natural selection is at work in the population, and natural selection often works under the assumption of conspecific competition. Alternatively, we might believe that the stability of the population dynamics that we see might be due to a limit on the number of individuals that a community can support. In other cases, we may see actual population trends that suggest a role for density, such as cycles of different periods.

Let's start our analysis by projecting the mean ahistorical matrix forward using four different parameterizations of the Ricker function modifying germination. Our first step will be to create four density frames (lefkoDens objects), which will describe the relationships between the matrices and density.

```
c2d_1 <- density_input(cypmean2, stage3 = c("P1", "P1"), stage2= c("SD", "rep"),
    style = 1, time_delay = 1, alpha = 0.05, beta = 0, type = c(2, 2))
```



Figure 10.11: Stochastic projections using equal (a) or different (b) matrix weights

```
c2d_2 <- density_input(cypmean2, stage3 = c("P1", "P1"), stage2= c("SD", "rep"),
    style = 1, time_delay = 1, alpha = 1, beta = 0, type = c(2, 2))
c2d_3 <- density_input(cypmean2, stage3 = c("P1", "P1"), stage2= c("SD", "rep"),
    style = 1, time_delay = 1, alpha = 0.05, beta = 0.0005, type = c(2, 2))
c2d_4 <- density_input(cypmean2, stage3 = c("P1", "P1"), stage2= c("SD", "rep"),
    style = 1, time_delay = 1, alpha = 1, beta = 0.0005, type = c(2, 2))
c2d_1
> stage3 stage2 stage1 age2 style alpha beta time_delay type type_t12
>1 P1 SD NA NA 
>2 P1 XSm NA NA 
> 3 P1 Sm NA NA 
>4 
c2d_2
> stage3 stage2 stage1 age2 style alpha beta time_delay type type_t12
\begin{tabular}{llrrrrrrrrr}
\(>1\) & P 1 & SD & NA & NA & 1 & 1 & 0 & 1 & 2 & 1 \\
\(>\) & P 1 & xSm & NA & NA & 1 & 1 & 0 & 1 & 2 & 1
\end{tabular}
\begin{tabular}{llllllllllll}
\(>2\) & \(P 1\) & \(X S m\) & \(N A\) & \(N A\) & 1 & 1 & 0 & 1 & 2 & 1
\end{tabular}
\begin{tabular}{lllllllllll}
\(>3\) & P 1 & Sm & NA & NA & 1 & 1 & 0 & 1 & 2 & 1
\end{tabular}
\begin{tabular}{llllllllllll}
\(>4\) & \(P 1\) & \(M d\) & \(N A\) & \(N A\) & 1 & 1 & 0 & 1 & 2 & 1
\end{tabular}
\begin{tabular}{rrrrrrrrrrr}
\(>5\) & \(P 1\) & Lg & NA & NA & 1 & 1 & 0 & 1 & 2 & 1 \\
\(>6\) & \(P 1\) & \(X L g\) & \(N A\) & NA & 1 & 1 & 0 & 1 & 2 & 1
\end{tabular}
c2d_3
> stage3 stage2 stage1 age2 style alpha beta time_delay type type_t12
\begin{tabular}{lllllllllll}
\(>1\) & \(P 1\) & \(S D\) & \(N A\) & \(N A\) & 1 & 0.05 & \(5 e-04\) & 1 & 2 & 1
\end{tabular}
\begin{tabular}{llllllllll}
\(>2\) & \(P 1\) & \(X S m\) & \(N A\) & \(N A\) & 1 & 0.05 & \(5 e-04\) & 1 & 2
\end{tabular}
\begin{tabular}{lllllllllll}
\(>3\) & \(P 1\) & \(S m\) & \(N A\) & \(N A\) & 1 & 0.05 & \(5 \mathrm{e}-04\) & 1 & 2 & 1 \\
\(>4\) & P 1 & \(M d\) & NA & NA & 1 & 0.05 & \(5 \mathrm{e}-04\) & 1 & 2 & 1
\end{tabular}
\begin{tabular}{lllllllllll}
\(>5\) & \(P 1\) & \(L g\) & \(N A\) & \(N A\) & 1 & 0.05 & \(5 e-04\) & 1 & 2 & 1
\end{tabular}
```



Determining the values of $\alpha$ and $\beta$ to use for the alpha and beta options generally takes a great deal of extra work. We have done this work separately and present only four interesting combinations that produce contrasting results in this case. Note also that the style, alpha, and beta terms can be vectors if necessary, giving different kinds of density dependence and different values for $\alpha$ and $\beta$ when needed.

Now let's conduct some short simulations. We will only project forward 100 time steps, and we will enforce full substochasticity.

```
cypproj2_d1 <- projection3(cypmean2, times = 100, substoch = 2, density = c2d_1)
> Warning: This function takes annual matrices as input. This lefkoMat object appears to be a set of
cypproj2_d2 <- projection3(cypmean2, times = 100, substoch = 2, density = c2d_2)
> Warning: This function takes annual matrices as input. This lefkoMat object appears to be a set of
cypproj2_d3 <- projection3(cypmean2, times = 100, substoch = 2, density = c2d_3)
> Warning: This function takes annual matrices as input. This lefkoMat object appears to be a set of
cypproj2_d4 <- projection3(cypmean2, times = 100, substoch = 2, density = c2d_4)
> Warning: This function takes annual matrices as input. This lefkoMat object appears to be a set of
summary(cypproj2_d1)
>
> The input lefkoProj object covers 1 population-patches.
> It includes }100\mathrm{ projected steps per replicate and 1 replicates.
> The number of replicates with population size above the threshold size of 1 is as in
> the following matrix, with pop-patches given by column and milepost times given by row:
> $milepost_sums
    1 1
> 1 1
> 26 1
> 51 1
> 76 1
> 101 1
>
> $extinction_times
> [1] NA
summary(cypproj2_d2)
>
> The input lefkoProj object covers 1 population-patches.
> It includes }100\mathrm{ projected steps per replicate and 1 replicates.
> The number of replicates with population size above the threshold size of 1 is as in
> the following matrix, with pop-patches given by column and milepost times given by row:
> $milepost_sums
> 1 1
> 1 1
```

```
> 26 1
> 51 1
> 76 1
> 101 1
>
> $extinction_times
> [1] NA
summary(cypproj2_d3)
>
> The input lefkoProj object covers 1 population-patches.
> It includes }100\mathrm{ projected steps per replicate and 1 replicates.
> The number of replicates with population size above the threshold size of 1 is as in
> the following matrix, with pop-patches given by column and milepost times given by row:
> $milepost_sums
> 11
> 1 1
> 26 1
> 51 1
> 76 1
> 101 1
>
> $extinction_times
> [1] NA
summary(cypproj2_d4)
>
> The input lefkoProj object covers 1 population-patches.
> It includes }100\mathrm{ projected steps per replicate and 1 replicates.
> The number of replicates with population size above the threshold size of 1 is as in
> the following matrix, with pop-patches given by column and milepost times given by row:
> $milepost_sums
> 1 1
> 1 1
> 26 1
> 51 1
> 76 1
> 101 1
>
> $extinction_times
> [1] NA
```

Let's plot these projections together and take a look (figure 10.12).

```
par(mfrow = c(2, 2))
plot(cypproj2_d1)
title("a)", adj = 0)
plot(cypproj2_d2, ylab = "")
title("b)", adj = 0)
plot(cypproj2_d3)
title("c)", adj = 0)
```

```
plot(cypproj2_d4, ylab = "")
title("d)", adj = 0)
```



Figure 10.12: Density dependent projections using the two-parameter Ricker function. (a) alpha $=$ 0.05 , beta $=0 ;(\mathrm{b})$ alpha $=1$, beta $=0 ;(\mathrm{c})$ alpha $=0.05$, beta $=0.0005$; and (d) alpha $=1$, beta $=$ 0.0005 .

Each projection behaves a little differently. Notably, raising $\alpha$ from 0 to 1 appears to keep the population from declining to extinction (b, d). Changing $\beta$ from 0 to 0.0005 leads to a seemingly stabilized population size (d) rather than what appears to be extinction (c). The patterns are contingent on this particular MPM and to some extent on the start vector, meaning that other MPMs will yield different patterns with these $\alpha$ and $\beta$ values.

Let's also try four parameterizations of the two-parameter Beverton-Holt function.

```
c2d_5 <- density_input(cypmean2, stage3 = c("P1", "P1"), stage2= c("SD", "rep"),
    style = 2, time_delay = 1, alpha = 0.05, beta = 0, type = c(2, 2))
c2d_6 <- density_input(cypmean2, stage3 = c("P1", "P1"), stage2= c("SD", "rep"),
    style = 2, time_delay = 1, alpha = 1, beta = 0, type = c(2, 2))
c2d_7 <- density_input(cypmean2, stage3 = c("P1", "P1"), stage2= c("SD", "rep"),
    style = 2, time_delay = 1, alpha = 0.05, beta = 1, type = c(2, 2))
c2d_8 <- density_input(cypmean2, stage3 = c("P1", "P1"), stage2= c("SD", "rep"),
    style = 2, time_delay = 1, alpha = 1, beta = 1, type = c(2, 2))
cypproj2_d5 <- projection3(cypmean2, times = 100, substoch = 2, density = c2d_5)
> Warning: This function takes annual matrices as input. This lefkoMat object appears to be a set of
```

```
cypproj2_d6 <- projection3(cypmean2, times = 100, substoch = 2, density = c2d_6)
> Warning: This function takes annual matrices as input. This lefkoMat object appears to be a set of
cypproj2_d7 <- projection3(cypmean2, times = 100, substoch = 2, density = c2d_7)
> Warning: This function takes annual matrices as input. This lefkoMat object appears to be a set of
cypproj2_d8 <- projection3(cypmean2, times = 100, substoch = 2, density = c2d_8)
> Warning: This function takes annual matrices as input. This lefkoMat object appears to be a set of
```

Now let's plot them (figure 10.13).

```
par(mfrow = c(2, 2))
plot(cypproj2_d5)
title("a)", adj = 0)
plot(cypproj2_d6, ylab = "")
title("b)", adj = 0)
plot(cypproj2_d7)
title("c)", adj = 0)
plot(cypproj2_d8, ylab = "")
title("d)", adj = 0)
```

The Beverton-Holt function is less likely to produce unusual shapes to population projections than the Ricker function. For example, it is generally not capable of producing chaos. This is both a strength and a weakness - its simplicity makes it attractive in many situations, but prevents it from matching reality in others. Users will need to explore the different density functions under different parameterizations to decide on an appropriate parameterization of density dependence for their research.

### 10.2.5 Ordered or cyclical density dependent simulations

We have already seen that we can set our projections to run through a specific sequence of years. This sequence can be combined with density dependence in any way that we wish. Here, we conduct a density dependent version of the previous projection in which we set the first ten years to repeatedly switch between two wet years (2007 and 2008), followed by ten years of switching between two drier years (2004 and 2005). To make things interesting in our 20 year projection, we utilize an Usher function with $\alpha=1$ and $\beta=0$, and our previous start frame with $10003^{\text {rd }}$ year protocorms and 100 dormant adults.

```
year_order <- c(2007, 2008, 2007, 2008, 2007, 2008, 2007, 2008, 2007, 2008,
    2004, 2005, 2004, 2005, 2004, 2005, 2004, 2005, 2004, 2005)
c2d_9 <- density_input(cypmean2, stage3 = c("P1", "P1"), stage2= c("SD", "rep"),
    style = 3, time_delay = 1, alpha = 1, beta = 0, type = c(2, 2))
cypproj2_ord1d <- projection3(cypmatrix2r, year = year_order, times = 20,
    start_frame = c2m_sv, density = c2d_9)
summary(cypproj2_ord1d)
>
> The input lefkoProj object covers 1 population-patches.
```



Figure 10.13: Density dependent projections using the two-parameter Beverton-Holt function. (a) alpha $=0.05$, beta $=0 ;(b)$ alpha $=1$, beta $=0 ;(c)$ alpha $=0.05$, beta $=1$; and $(d)$ alpha $=1$, beta $=1$.

```
> It includes 20 projected steps per replicate and 1 replicates.
> The number of replicates with population size above the threshold size of 1 is as in
> the following matrix, with pop-patches given by column and milepost times given by row:
> $milepost_sums
> 11
> 1 1
> 6 1
> 11 1
> 16 1
> 21 1
>
> $extinction_times
> [1] NA
```

As before, if the sequence of years noted in the year option is shorter than the time steps designated in the times option, then lefko3 will cycle through the sequences of years until the projection is done. For example, if we leave the times option out from the above projection, then lefko3 will assume the default of 10,000 time steps, and cycle through our 20 defined years over and over, as below.

```
cypproj2_ord2d <- projection3(cypmatrix2r, year = year_order,
    start_frame = c2m_sv, density = c2d_9)
summary(cypproj2_ord2d)
>
> The input lefkoProj object covers 1 population-patches.
> It includes 10000 projected steps per replicate and 1 replicates.
> The number of replicates with population size above the threshold size of 1 is as in
> the following matrix, with pop-patches given by column and milepost times given by row:
> $milepost_sums
> 1 1
> 1 1
> 2501 0
> 5001 0
> 7501 0
> 10001 0
>
> $extinction_times
> [1] NA
```

Let's see plots of these projections (figure 10.14).

```
par(mfrow=c(1,2))
plot(cypproj2_ord1d)
title("a)", adj = 0)
plot(cypproj2_ord2d, ylab = "")
title("b)", adj = 0)
```

It appears that this particular form of density dependence leads to extinction fairly early on, though only after some strong fluctuations.

Projecting a full MPM with multiple annual matrices defaults to cycling through the matrices in order, even when density dependence is set. Let's try the first four Ricker projections again, but this


Figure 10.14: Ordered, density dependent projections can be cyclical. (a) A 20 year ordered projection. (b) A 10,000 year projection cycling through 20 ordered years
time using the full ahistorical MPM rather than the arithmetic mean matrix to allow cycling of year sequences.

```
cypproj2r_d1 <- projection3(cypmatrix2r, times = 100, substoch = 2,
    density = c2d_1)
cypproj2r_d2 <- projection3(cypmatrix2r, times = 100, substoch = 2,
    density = c2d_2)
cypproj2r_d3 <- projection3(cypmatrix2r, times = 100, substoch = 2,
    density = c2d_3)
cypproj2r_d4 <- projection3(cypmatrix2r, times = 100, substoch = 2,
    density = c2d_4)
summary(cypproj2r_d1)
>
> The input lefkoProj object covers 1 population-patches.
> It includes }100\mathrm{ projected steps per replicate and 1 replicates.
> The number of replicates with population size above the threshold size of 1 is as in
> the following matrix, with pop-patches given by column and milepost times given by row:
> $milepost_sums
> 11
> 1 1
> 26 1
> 51 1
> 76 1
> 101 1
>
> $extinction_times
> [1] NA
summary(cypproj2r_d2)
>
```

```
> The input lefkoProj object covers 1 population-patches.
> It includes }100\mathrm{ projected steps per replicate and 1 replicates.
> The number of replicates with population size above the threshold size of 1 is as in
> the following matrix, with pop-patches given by column and milepost times given by row:
> $milepost_sums
> 1 1
> 1 1
> 26 1
> 51 1
> 76 1
> 101 1
>
> $extinction_times
> [1] NA
summary(cypproj2r_d3)
>
> The input lefkoProj object covers 1 population-patches.
> It includes }100\mathrm{ projected steps per replicate and 1 replicates.
> The number of replicates with population size above the threshold size of 1 is as in
> the following matrix, with pop-patches given by column and milepost times given by row:
> $milepost_sums
> 1 1
> 1 1
> 26 1
> 51 1
> 76 1
> 101 1
>
> $extinction_times
> [1] NA
summary(cypproj2r_d4)
>
> The input lefkoProj object covers 1 population-patches.
> It includes }100\mathrm{ projected steps per replicate and 1 replicates.
> The number of replicates with population size above the threshold size of 1 is as in
> the following matrix, with pop-patches given by column and milepost times given by row:
> $milepost_sums
> 11
> 1 1
> 26 1
> 51 1
> 76 1
> 101 1
>
> $extinction_times
> [1] NA
```

Let's see what these projections look like (10.15).

```
par(mfrow=c(2, 2))
plot(cypproj2r_d1)
```

```
title("a)", adj = 0)
plot(cypproj2r_d2, ylab = "")
title("b)", adj = 0)
plot(cypproj2r_d3)
title("c)", adj = 0)
plot(cypproj2r_d4, ylab = "")
title("d)", adj = 0)
```



Figure 10.15: Cyclical density dependent projections using the Ricker function

We can see the obvious cycles in all projections, but the population seems to be on the road to extinction in a) and c). Projection b) appears to show a growing population, while d) seems to be entering a stable oscillation.

### 10.2.6 Replicated simulations assuming temporal stochasticity and density dependence

Finally, we may wish to conduct projections assuming both temporal stochasticity and density dependence. Here, we will set up four projections, each assuming one of four density frames with different parameterizations of the Ricker function working on germination (we have defined these density frames previously).

```
set.seed(42)
cypproj2r_sd1 <- projection3(cypmatrix2r, stochastic = TRUE, times = 100,
    substoch = 2, density = c2d_1)
cypproj2r_sd2 <- projection3(cypmatrix2r, stochastic = TRUE, times = 100,
    substoch = 2, density = c2d_2)
cypproj2r_sd3 <- projection3(cypmatrix2r, stochastic = TRUE, times = 100,
    substoch = 2, density = c2d_3)
cypproj2r_sd4 <- projection3(cypmatrix2r, stochastic = TRUE, times = 100,
    substoch = 2, density = c2d_4)
summary(cypproj2r_sd1)
>
> The input lefkoProj object covers 1 population-patches.
> It includes }100\mathrm{ projected steps per replicate and 1 replicates.
> The number of replicates with population size above the threshold size of 1 is as in
> the following matrix, with pop-patches given by column and milepost times given by row:
> $milepost_sums
> 11
> 1 1
> 26 1
> 51 1
> 76 1
> 101 1
>
> $extinction_times
> [1] NA
summary(cypproj2r_sd2)
>
> The input lefkoProj object covers 1 population-patches.
> It includes }100\mathrm{ projected steps per replicate and 1 replicates.
> The number of replicates with population size above the threshold size of 1 is as in
> the following matrix, with pop-patches given by column and milepost times given by row:
> $milepost_sums
> 1 1
> 1 1
> 26 1
> 51 1
> 76 1
> 101 1
>
> $extinction_times
> [1] NA
summary(cypproj2r_sd3)
>
> The input lefkoProj object covers 1 population-patches.
> It includes }100\mathrm{ projected steps per replicate and 1 replicates.
> The number of replicates with population size above the threshold size of 1 is as in
> the following matrix, with pop-patches given by column and milepost times given by row:
> $milepost_sums
> 11
> 1 1
```

```
> 26 1
> 51 1
> 76 1
> 101 1
>
> $extinction_times
> [1] NA
summary(cypproj2r_sd4)
>
> The input lefkoProj object covers 1 population-patches.
> It includes }100\mathrm{ projected steps per replicate and 1 replicates.
> The number of replicates with population size above the threshold size of 1 is as in
> the following matrix, with pop-patches given by column and milepost times given by row:
> $milepost_sums
> 1 1
> 1 1
> 26 1
> 51 1
> 76 1
> 101 1
>
> $extinction_times
> [1] NA
```

Once again, let's plot the projections for comparison.

```
par(mfrow=c(2, 2))
plot(cypproj2r_sd1)
title("a)", adj = 0)
plot(cypproj2r_sd2, ylab = "")
title("b)", adj = 0)
plot(cypproj2r_sd3)
title("c)", adj = 0)
plot(cypproj2r_sd4, ylab = "")
title("d)", adj = 0)
```

We can clearly see the impact of temporal environmental stochasticity in each plot, and different density dependence relationships influence population trajectories strongly across the plots. Projections a) and c) appear to be moving to extinction, while projection b) is growing and projection d) appears to be fluctuating around some stable number. We can alter other settings, such as changing the weights of the matrices and using different density dependence relationships, as well as replicating our stochastic runs. Here, we re-run the first two projections with 100 replicates each and plot the results.

```
set.seed(42)
cypproj2r_sd1_100 <- projection3(cypmatrix2r, stochastic = TRUE, times = 100,
    nreps = 100, substoch = 2, density = c2d_1)
cypproj2r_sd2_100 <- projection3(cypmatrix2r, stochastic = TRUE, times = 100,
    nreps = 100, substoch = 2, density = c2d_2)
```



Figure 10.16: Stochastic density dependent projections assuming the two-parameter Ricker functon. (a) alpha $=0.05$, beta $=0 ;(\mathrm{b})$ alpha $=1$, beta $=0 ;(\mathrm{c})$ alpha $=0.05$, beta $=0.0005$; and (d) alpha $=1$, beta $=0.0005$.

```
par(mfrow = c(1, 2))
plot(cypproj2r_sd1_100)
title("a)", adj = 0)
plot(cypproj2r_sd2_100, ylab = "")
title("b)", adj = 0)
```



Figure 10.17: Stochastic density dependent projections with replicates assuming the two-parameter Ricker function. (a) alpha $=0.05$, beta $=0$; and $(b)$ alpha $=1$, beta $=0$.

Note that each plot includes all replicates, each as a separate line. The general trends are the same as before, but we see the degree of variability in these trends is large.

### 10.3 Projecting function-based MPMs

Function projection3() allows users to project matrices forward in time, but is limited to the case in which we have already developed our MPM and wish to project using only those matrices. What if we wished to create a new matrix at each time step, perhaps because of some changing conditions that should alter basic vital rates? Function-based projection through function f_projection3() provides us with the flexibility to construct custom matrices at each time step given some set of conditions that we are interested in. This approach is quite useful because of the flexibility offered. The price paid is that it takes extra time to develop matrices at each time step in the projection, making standard MPM projection through function projection3() preferable except in cases where custom matrices are genuinely needed at each time step.

What sorts of situations might require custom matrices at each time step? One obvious situation is when matrices are to be built using set values of an individual or environmental covariate. For example, we may have developed a set of vital rate models that have relationships with climatic variables that are likely to change going into the future. We might then wish to project the population forward under different climate change scenarios, with vectors of specific values of these climatic variables derived from climate prediction models (e.g., Shefferson et al., 2017). In this circumstance, using already made
matrices in some sequence will not likely be of much help to us because matrices should change at each time step as a function of climate. Instead, we should create new matrices at each projected time, assuming the values of the climatic variables that are relevant.

Let's explore this with the Cypripedium candidum dataset. We introduced the model and code for for our stageframe and supplements in Chapter 2, and showed how to build function-based MPMs with all of this material in Chapter 5. Let's look again at the life history model that we will use (note that this is the same as figure 2.3).


Figure 10.18: Life history model of Cypripedium candidum for use in function-based MPMs

This is a fairly large life history model with two adult life stages for each number of sprouts - a reproductive stage that flowers and a non-reproductive stage that does not flower. Let's load the raw dataset and set up the stageframe first.

```
data(cypdata)
stagevector <- c("SD", "P1", "P2", "P3", "SL", "D", "V1", "V2", "V3", "V4",
    "v5", "V6", "v7", "v8", "v9", "v10", "V11", "V12", "V13", "V14", "V15", "V16",
    "V17", "V18", "V19", "V20", "V21", "V22", "V23", "V24", "F1", "F2", "F3",
    "F4", "F5", "F6", "F7", "F8", "F9", "F10", "F11", "F12", "F13", "F14", "F15",
    "F16", "F17", "F18", "F19", "F20", "F21", "F22", "F23", "F24")
indataset <- c(0, 0, 0, 0, 0, rep(1, 49))
sizevector <- c(0, 0, 0, 0, 0, seq(from = 0, t = 24), seq(from = 1, to = 24))
repvector <- c(0, 0, 0, 0, 0, rep(0, 25), rep(1, 24))
obsvector <- c(0, 0, 0, 0, 0, 0, rep(1, 48))
matvector <- c(0, 0, 0, 0, 0, rep(1, 49))
immvector <- c(0, 1, 1, 1, 1, rep(0, 49))
propvector <- c(1, rep(0, 53))
comments <- c("Dormant seed", "Yr1 protocorm", "Yr2 protocorm", "Yr3 protocorm",
```

```
"Seedling", "Veg dorm", "Veg adult 1 stem", "Veg adult 2 stems",
"Veg adult 3 stems", "Veg adult 4 stems", "Veg adult 5 stems",
"Veg adult 6 stems", "Veg adult 7 stems", "Veg adult 8 stems",
"Veg adult 9 stems", "Veg adult 10 stems", "Veg adult 11 stems",
"Veg adult 12 stems", "Veg adult 13 stems", "Veg adult 14 stems",
"Veg adult 15 stems", "Veg adult 16 stems", "Veg adult 17 stems",
"Veg adult 18 stems", "Veg adult 19 stems", "Veg adult 20 stems",
"Veg adult 21 stems", "Veg adult 22 stems", "Veg adult 23 stems",
"Veg adult 24 stems", "Flo adult 1 stem", "Flo adult 2 stems",
"Flo adult 3 stems", "Flo adult 4 stems", "Flo adult 5 stems",
"Flo adult 6 stems", "Flo adult 7 stems", "Flo adult 8 stems",
"Flo adult 9 stems", "Flo adult 10 stems", "Flo adult 11 stems",
"Flo adult 12 stems", "Flo adult 13 stems", "Flo adult 14 stems",
"Flo adult 15 stems", "Flo adult 16 stems", "Flo adult 17 stems",
"Flo adult 18 stems", "Flo adult 19 stems", "Flo adult 20 stems",
"Flo adult 21 stems", "Flo adult 22 stems", "Flo adult 23 stems",
"Flo adult 24 stems")
cypframe_fb <- sf_create(sizes = sizevector, stagenames = stagevector,
repstatus = repvector, obsstatus = obsvector, matstatus = matvector,
propstatus = propvector, immstatus = immvector, indataset = indataset,
comments = comments)
```

Now let's standardize the dataset into a historically formatted vertical (hfv) data frame. We will also need to introduce our individual covariate, total annual precipitation. These data were derived from the National Centers for Environmental Information and the U.S. National Oceanic and Atmospheric Administration. We will insert these climatic data as constants into the data frame holding the dataset.

```
cypdata_env <- cypdata
cypdata_env$prec.04 <- 92.2
cypdata_env$prec.05 <- 57.6
cypdata_env$prec.06 <- 96.0
cypdata_env$prec.07 <- 109.8
cypdata_env$prec.08 <- 111.9
cypdata_env$prec.09 <- 106.8
cypfb_env <- verticalize3(data = cypdata_env, noyears = 6, firstyear = 2004,
    patchidcol = "patch", individcol = "plantid", blocksize = 4,
    sizeacol = "Inf2.04", sizebcol = "Inf.04", sizeccol = "Veg.04",
    repstracol = "Inf.04", repstrbcol = "Inf2.04", fecacol = "Pod.04",
    indcovacol = c("prec.04", "prec.05", "prec.06", "prec.07", "prec.08",
        "prec.09"), stageassign = cypframe_fb, stagesize = "sizeadded",
    NAas0 = TRUE, age_offset = 4)
summary_hfv(cypfb_env)
>
> This hfv dataset contains 320 rows, 60 variables, 1 population,
> 3 patches, }74\mathrm{ individuals, and 5 time steps.
\(>\) rowid popid patchid individ year2
> Min. : 1.00 Length:320 A: 93 Min. : 164.0 Min. :2004
> 1st Qu.:21.00 Class :character B:154 1st Qu.: 391.0 1st Qu.:2005
> Median :37.50 Mode :character C: 73 Median : 453.0 Median :2006
```




Now we will load the ahistorical supplement table that we will use for the ahistorical projections.

We will not work with the historical version in this chapter, just due to the increased amount of processing time required for the historical modelsearch() run.

```
seeds_per_fruit <- 5000 # Mean number of seeds per fruit
sl_mult <- 0.7 # Product of assumed germination probability and survival to 1st year
cypsupp2_fb <- supplemental(stage3 = c("SD", "P1", "P2", "P3", "SL", "SL", "D",
    "V1", "V2", "V3", "SD", "P1"),
    stage2 = c("SD", "SD", "P1", "P2", "P3", "SL", "SL", "SL", "SL", "SL", "rep",
        "rep"),
    eststage3 = c(NA, NA, NA, NA, NA, NA, "D", "V1", "V2", "V3", NA, NA),
    eststage2 = c(NA, NA, NA, NA, NA, NA, "D", "D", "D", "D", NA, NA),
    givenrate = c(0.08, 0.1, 0.1, 0.1, 0.05, 0.05, NA, NA, NA, NA, NA, NA),
    multiplier = c(NA, NA, NA, NA, NA, NA, sl_mult, sl_mult, sl_mult, sl_mult,
        0.5 * seeds_per_fruit, 0.5 * seeds_per_fruit),
    type = c("S", "S", "S", "S", "S", "S", "S", "S", "S", "S", "R", "R"),
    stageframe = cypframe_fb, historical = FALSE)
```

| cypsupp2_fb |  |  |  |  |  |  |  |  |  |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| > | stage3 | stage 2 | stage1 | age2 | eststage3 | eststage2 | eststage1 | estage2 | givenrate |
| $>1$ | SD | SD | <NA> | NA | <NA> | <NA> | <NA> | NA | 0.08 |
| $>2$ | P1 | SD | <NA> | NA | <NA> | <NA> | <NA> | NA | 0.10 |
| $>3$ | P2 | P1 | <NA> | NA | <NA> | <NA> | <NA> | NA | 0.10 |
| $>4$ | P3 | P2 | <NA> | NA | <NA> | <NA> | <NA> | NA | 0.10 |
| $>5$ | SL | P3 | <NA> | NA | <NA> | <NA> | <NA> | NA | 0.05 |
| $>6$ | SL | SL | <NA> | NA | <NA> | <NA> | <NA> | NA | 0.05 |
| $>7$ | D | SL | <NA> | NA | D | D | <NA> | NA | NA |
| $>8$ | V1 | SL | <NA> | NA | V1 | D | <NA> | NA | NA |
| $>9$ | V2 | SL | <NA> | NA | V2 | D | <NA> | NA | NA |
| > 10 | V3 | SL | <NA> | NA | V3 | D | <NA> | NA | NA |
| $>11$ | SD | rep | <NA> | NA | <NA> | <NA> | <NA> | NA | NA |
| > 12 | P1 | rep | <NA> | NA | <NA> | <NA> | <NA> | NA | NA |
| > multiplier convtype convtype_t12 |  |  |  |  |  |  |  |  |  |

$>1$ NA 1

| $>2$ | NA | 1 | 1 |
| :--- | :--- | :--- | :--- |

$\begin{array}{llll}> & \text { NA } & 1 & 1\end{array}$
$\begin{array}{llll}>4 & \text { NA } & 1 & 1\end{array}$
$\begin{array}{llll}>5 & \text { NA } & 1 & 1\end{array}$
$\begin{array}{llll}>6 & \text { NA } & 1 & 1\end{array}$
$\begin{array}{llll}>7 & 0.7 & 1 & 1\end{array}$
$\begin{array}{llll}>8 & 0.7 & 1 & 1\end{array}$
$\begin{array}{llll}>9 & 0.7 & 1 & 1\end{array}$
$\begin{array}{llll}> & 10 & 0.7 & 1\end{array}$
$\begin{array}{llll}>11 & 2500.0 & 3 & 1\end{array}$
$\begin{array}{llll}>12 & 2500.0 & 3 & 1\end{array}$

Let's now determine the proper distributions for size and fecundity. As a reminder, size in the Cypripedium dataset is given as the numbers of sprouts, and our proxy for fecundity can be either the number of flowers or the number of fruits produced by an individual. So, size and fecundity in this dataset are all non-negative integers. Count variables generally break the assumptions of a Gaussian distribution because the mean and variance are strongly related, possible values are discrete, and distributions are often dramatically skewed. Additionally, because we have absorbed individuals with
a size of zero into the dormant stage, there are no zeros left for the size model and so we will need a zero-truncated distribution. Fecundity, in contrast, is a count variable that includes zeros and so we should test whether we need a zero-inflated model. As for which specific distributions to use, we will use function sf_distrib() to make this determination.

```
hfv_qc(data = cypfb_env, vitalrates = c("surv", "obs", "size", "repst", "fec"),
    size = c("size3added", "size2added", "size1added"),
    indcova = c("indcova3", "indcova2", "indcova1"))
    Survival:
>
> Data subset has 61 variables and 320 transitions.
>
> Variable alive3 has 0 missing values.
> Variable alive3 is a binomial variable.
>
>
> Observation status:
>
> Data subset has 61 variables and 303 transitions.
>
> Variable obsstatus3 has O missing values.
> Variable obsstatus3 is a binomial variable.
>
>
> Primary size:
>
> Data subset has 61 variables and 288 transitions.
>
> Variable size3added has 0 missing values.
> Variable size3added appears to be an integer variable.
>
> Variable size3added is fully positive, lacking even Os.
>
> Overdispersion test:
> Mean size3added is 3.653
> The variance in size3added is 13.41
> The probability of this dispersion level by chance assuming that
> the true mean size3added = variance in size3added,
> and an alternative hypothesis of overdispersion, is 3.721e-138
> Variable size3added is significantly overdispersed.
>
> Zero-inflation and truncation tests:
> Mean lambda in size3added is 0.02592
> The actual number of 0s in size3added is 0
> The expected number of Os in size3added under the null hypothesis is 7.465
> The probability of this deviation in Os from expectation by chance is 0.9964
> Variable size3added is not significantly zero-inflated.
>
> Variable size3added does not include 0s, suggesting that a zero-truncated distribution may be w
>
>
Reproductive status:
```

```
>
> Data subset has 61 variables and 288 transitions.
>
> Variable repstatus3 has 0 missing values.
> Variable repstatus3 is a binomial variable.
>
>
> Fecundity:
>
> Data subset has 61 variables and 118 transitions.
>
> Variable feca2 has 0 missing values.
Variable feca2 appears to be an integer variable.
>
> Variable feca2 is fully non-negative.
>
> Overdispersion test:
> Mean feca2 is 0.7881
> The variance in feca2 is 1.536
> The probability of this dispersion level by chance assuming that
> the true mean feca2 = variance in feca2,
> and an alternative hypothesis of overdispersion, is 0.1193
> Dispersion level in feca2 matches expectation.
>
> Zero-inflation and truncation tests:
> Mean lambda in feca2 is 0.4547
> The actual number of 0s in feca2 is 68
> The expected number of 0s in feca2 under the null hypothesis is 53.65
> The probability of this deviation in Os from expectation by chance is 5.904e-06
> Variable feca2 is significantly zero-inflated.
```

The results show that size is significantly overdispersed. This means that we cannot use the Poisson distribution, and should instead use the negative binomial distribution. The lack of zeroes in size when zeroes are clearly expected also implies the use of a zero-truncated distribution. Fecundity is not significantly overdispersed, but it is significantly zero-inflated, so we will require a zero-inflated Poisson distribution for that vital rate. All other response variables appear to match the expectations of the binomial distribution, and so are fine.

Now we will run function modelsearch() and find the right parameterizations for our vital rates. Note that we need to include indcova = c("indcova3", "indcova2", "indcova1") and test.indcova $=$ TRUE in order to make sure that modelsearch() includes our individual covariate, which reflects total annual precipitation. If these covariates were categorical and random, then we would also need to stipulate random.indcova $=$ TRUE (our individual covariate is actually quantitative and numerical, so we will use the default setting of FALSE, instead). To prevent this function from taking too long to run, we will drop the patch term, set suite = "main" to prevent the inclusion of interaction terms, and run only an ahistorical set.

```
cypmodels2_env1 <- modelsearch(cypfb_env, historical = FALSE, approach = "mixed",
    vitalrates = c("surv", "obs", "size", "repst", "fec"), sizedist = "negbin",
    size.trunc = TRUE, fecdist = "poisson", fec.zero = TRUE, suite = "main",
    size = c("size3added", "size2added", "size1added"), test.indcova = TRUE,
    indcova = c("indcova3", "indcova2", "indcova1"), quiet = "partial")
```

Let's see the model summaries.

```
summary(cypmodels2_env1)
> This LefkoMod object includes 5 linear models.
> Best-fit model criterion used: aicc&k
>
>
>
Survival model:
Generalized linear mixed model fit by maximum likelihood (Laplace
        Approximation) [glmerMod]
    Family: binomial ( logit )
Formula: alive3 ~ size2added + (1 | year2) + (1 | individ)
            Data: subdata
            AIC BIC logLik deviance df.resid
128.1324 143.2057-60.0662 120.1324 316
Random effects:
    Groups Name Std.Dev.
    individ (Intercept) 1.198381
year2 (Intercept) 0.008826
Number of obs: 320, groups: individ, 74; year2, 5
Fixed Effects:
(Intercept) size2added
2.0352 0.6344
optimizer (Nelder_Mead) convergence code: O (OK) ; 0 optimizer warnings; 1 lme4 warnings
>
>
>
Observation model:
Generalized linear mixed model fit by maximum likelihood (Laplace
            Approximation) [glmerMod]
Family: binomial ( logit )
Formula: obsstatus3 ~ size2added + (1 | year2) + (1 | individ)
            Data: subdata
            AIC BIC logLik deviance df.resid
118.2567 133.1117 -55.1284 110.2567 299
Random effects:
    Groups Name Std.Dev.
    individ (Intercept) 1.078e-05
    year2 (Intercept) 8.776e-01
Number of obs: 303, groups: individ, 70; year2, 5
Fixed Effects:
(Intercept) size2added
            2.4904 0.3134
optimizer (Nelder_Mead) convergence code: 0 (OK) ; 0 optimizer warnings; 1 lme4 warnings
>
>
>
Size model:
Formula: size3added ~ (1 | year2) + (1 | individ)
Data: subdata
AIC BIC logLik df.resid
```

```
1008.2763 1022.9282 -500.1382 284
Random-effects (co)variances:
>
Conditional model:
Groups Name Std.Dev.
    year2 (Intercept) 0.1109
    individ (Intercept) 1.0561
Number of obs: 288 / Conditional model: year2, 5; individ, 70
Dispersion parameter for truncated_nbinom2 family (): 2.59e+07
Fixed Effects:
Conditional model:
(Intercept)
            0.5761
Secondary size model:
[1] 1
```



```
>
>
Tertiary size model:
[1] 1
Reproductive status model:
Generalized linear mixed model fit by maximum likelihood (Laplace
    Approximation) [glmerMod]
    Family: binomial ( logit )
Formula: repstatus3 ~ indcova2 + repstatus2 + size2added + (1 | year2) +
            (1 | individ)
        Data: subdata
            AIC BIC logLik deviance df.resid
    330.0418 352.0196-159.0209 318.0418 282
Random effects:
    Groups Name Std.Dev.
    individ (Intercept) 0.0002695
    year2 (Intercept) 0.2479120
Number of obs: 288, groups: individ, 70; year2, 5
Fixed Effects:
(Intercept) indcova2 repstatus2 size2added
    -4.23766 0.02954 1.71091 0.17187
Fecundity model:
```

```
> Formula: feca2 ~ indcova2 + size2added + (1 | year2) + (1 | individ)
> Zero inflation: ~size2added + (1 | year2) + (1 | individ)
> Data: subdata
> AIC BIC logLik df.resid
245.4867 270.4229 -113.7434 109
Random-effects (co)variances:
Conditional model:
\ Groups Name Std.Dev.
year2 (Intercept) 0.2517
individ (Intercept) 0.1927
Zero-inflation model:
Groups Name Std.Dev.
year2 (Intercept) 2.072e-04
    individ (Intercept) 3.951e-08
>
Number of obs: 118 / Conditional model: year2, 5; individ, 51 / Zero-inflation model: year2, 5; ind
Fixed Effects:
>
> Conditional model:
> (Intercept) indcova2 size2added
> 1.68997 -0.02397 0.06174
>
> Zero-inflation model:
> (Intercept) size2added
> 3.927 -1.618
>
>
> Juvenile survival model:
[1] 1
>
>
>
> Juvenile observation model:
[1] 1
>
>
>
> Juvenile size model:
[1] 1
>
>
>
> Juvenile secondary size model:
[1] 1
>
>
>
> Juvenile tertiary size model:
```

```
[1] 1
>
>
> Juvenile reproduction model:
[1] 1
Juvenile maturity model:
[1] 1
>
>
>
>
> Number of models in survival table: 7
Number of models in observation table: 8
Number of models in size table: 8
Number of models in secondary size table: 1
Number of models in tertiary size table: 1
Number of models in reproduction status table: 8
Number of models in fecundity table: 60
Number of models in juvenile survival table: 1
Number of models in juvenile observation table: 1
Number of models in juvenile size table: 1
Number of models in juvenile secondary size table: 1
Number of models in juvenile tertiary size table: 1
Number of models in juvenile reproduction table: 1
Number of models in juvenile maturity table: 1
>
>
>
>
> General model parameter names (column 1), and
specific names used in these models (column 2):
    parameter_names mainparams
```

$>$
$>$
$>$

```
> 1
> 2
> 3
> 4
> 5
> 6
> 7
> 8
> 9
> 10
> 11
> 12
> 13
> 14
> 15
> 16
> 17 sizec in time t-1 sizec1
> 18 reproductive status in time t repst2
> 19
> 20
    > 21
    >22
22 age in time t age
> 23 density in time t density
> 24 individual covariate a in time t indcova2
> 25 individual covariate a in time t-1 indcova1
> 26 individual covariate b in time t indcovb2
> 27 individual covariate b in time t-1 indcovb1
> 28 individual covariate c in time t indcovc2
> 29 individual covariate c in time t-1 indcovc1
> 30 stage group in time t group2
> 31 stage group in time t-1 group1
>
>
>
>
>
> Quality control:
>
> Survival model estimated with 74 individuals and 320 individual transitions.
> Survival model accuracy is 0.947.
> Observation status model estimated with 70 individuals and 303 individual transitions.
> Observation status model accuracy is 0.95.
> Primary size model estimated with 70 individuals and 288 individual transitions.
> Primary size model R-squared is 0.822.
> Secondary size model not estimated.
> Tertiary size model not estimated.
> Reproductive status model estimated with 70 individuals and 288 individual transitions.
> Reproductive status model accuracy is 0.719.
> Fecundity model estimated with 51 individuals and }118\mathrm{ individual transitions.
> Fecundity model R-squared is 0.524.
> Juvenile survival model not estimated.
```

```
> Juvenile observation status model not estimated.
> Juvenile primary size model not estimated.
> Juvenile secondary size model not estimated.
> Juvenile tertiary size model not estimated.
> Juvenile reproductive status model not estimated.
> Juvenile maturity status model not estimated.
```

Total annual precipitation is a factor in the best-fit models of reproductive status and the conditional portion of fecundity, suggesting that it is quite important to the demography of the population. The accuracy of survival and observation status is really excellent, while the other models have poor to reasonable accuracy or $\mathrm{R}^{2}$.

Now that we have developed our vital rate models, let's also build vectors showing our expectation for shifts in the precipitation over the next 20 years from the year that data collection stopped. This 20 year period will be our main time window of interest for projection. Our climate values will be real predictions from Shefferson et al. (2017), which were extracted from a real climatological forecast model maintained by Japan's National Meteorological Research Institute. We will use the values predicted for 2010 to 2030. Just out of interest, we will also create a vector of predicted values for 2070 to 2090.

```
pred_vals_2010to30 <- c(110.8152136, 98.87159542, 127.0623005, 110.0501797,
    82.03005742, 103.6893125, 104.2549713, 97.86989854, 111.25054, 116.4481035,
    125.9652504, 98.89823738, 105.7075192, 135.7032496, 149.6705465, 149.0110188,
    115.3965537, 119.7017105, 99.5415225, 111.7898123, 90.51828146)
pred_vals_2070to90 <- c(101.7828506, 80.70954521, 106.760383, 142.3355673,
    113.6519693, 122.3257958, 100.8745268, 138.814026, 111.1349475, 92.00440938,
    120.2391638, 115.6253448, 102.6373673, 125.3819469, 98.25464504, 102.4367633,
    118.6401779, 125.6021119, 101.9117747, 127.2149739, 109.244138)
```

Let's see what these values look like graphically (figure 10.19).

```
plot(x = c(1:21), y = pred_vals_2010to30, type = "l", lty = 1, lwd = 3,
    ylab = "Annual precip (cm)", xlab = "Projected year", ylim = c(80, 170),
    bty = "n")
lines(x = c(1:21), y = pred_vals_2070to90, lty = 2, lwd = 3)
legend("topright", c("2010-2030", "2070-2090"), lty = c(1, 2), lwd = 3,
    bty = "n")
```

Now let's run our projections. Note that many of the options in f_projection3() work similarly to those of projection3(). For example, projection can be limited to whole individuals if integeronly $=$ TRUE is set (we will not do so here in order to see the overall population patterns over time, and limit the influence of population size on predicted extinction risk). Users should experiment with the various options to see the impacts.

### 10.3.1 Deterministic projections of a specific time length

The first style of projection that we will try is a standard deterministic projection using our individual covariate values. We have two choices - we may either choose a single year to run (our vital rates all include random year terms) with our projected precipitation values, or we may try a specific sequence of years. There are different merits to both approaches. For instructional purposes, let's start by choosing a single year in the middle of the monitoring - 2007. We also need to format the individual covariate into a three-column data frame with as many rows as we wish to project, and set zero as the default value for unused covariates. Here, we develop our first projection using the 2010 to 2030 predictions.


Figure 10.19: Predicted annual precipitation for Cypripedium candidum

Note that f_projection3() does not calculate mean matrices, unlike function projection3(), so we cannot use that approach.

```
ind_frame_1 <- cbind.data.frame(inda = pred_vals_2010to30, indb = 0, indc = 0)
trial2f_1 <- f_projection3(data = cypfb_env, format = 3, stageframe = cypframe_fb,
    supplement = cypsupp2_fb, modelsuite = cypmodels2_env1, times = 21,
    year = 2007, ind_terms = ind_frame_1)
> Warning: Option patch not set, so will set to first patch/population.
```

The resulting object is of class lefkoProj, just as in the case of function projection3() above. The object includes the following elements:

1. projection - A list with a single element representing the one population or patch projected. This element is itself a list with number of elements equal to the number of replicates (option nreps, which defaults to 1 ). Each element in this lower-level list is a matrix with number of rows equal to the number of stages (if ahistorical; paired stages if historical; ages if Leslie; and age-stages if age-by-stage), and number of columns equal to the projected time steps plus 1 (the start time is time 0). The elements of the matrix show the projected number of individuals in that stage in that time.
2. stage_dist - A list showing the stage distribution projected per time, with the same order and number of elements as in element projection. Only estimated if growthonly = FALSE.
3. rep__value - A list showing the reproductive value of each stage projected per time, with the same order and number of elements as in element projection. Only estimated if growthonly = FALSE and repvalue = TRUE.
4. pop__size - A list with number of elements equal to the number of projected population-patches (in f_projection3(), this can only be one population or patch, so the length of the list will always be one for this function). This element is a matrix showing the total population size at each time (column) per replicate (row).
5. ahstages - The stageframe used in analysis, though modified and edited per rules set for functionbased MPM estimators.
6. hstages - A data frame matrix showing the pairing of ahistorical stages used to create historical stage pairs (only if historical).
7. agestages - A data frame showing the order of age-stage pairs (only if age-by-stage).
8. labels - A data frame showing the order of population and patch identities used.
9. control - An integer vector indicating the number of replicates and the number of time steps.
10. density - The data frame input under the density option. This data frame shows density dependence per matrix element, if such information was provided.
11. density__vr - The data frame input under the density_vr option. This data frame shows density dependence per vital rate model, if such information was provided. Note that this element is only output from function f_projection().

We can view a summary of this object, as follows.

```
summary(trial2f_1)
>
> The input lefkoProj object covers 1 population-patches.
> It includes 21 projected steps per replicate and 1 replicates.
> The number of replicates with population size above the threshold size of 1 is as in
> the following matrix, with pop-patches given by column and milepost times given by row:
> $milepost_sums
> 11
> 1 1
> 6 1
> 11 1
> 16 1
> 22 1
>
> $extinction_times
> [1] NA
```

Let's now also build a projection covering climatic values projected for 2070 to 2090.

```
ind_frame_2 <- cbind.data.frame(inda = pred_vals_2070to90, indb = 0, indc = 0)
trial2f_2 <- f_projection3(data = cypfb_env, format = 3, stageframe = cypframe_fb,
    supplement = cypsupp2_fb, modelsuite = cypmodels2_env1, times = 21,
    year = 2007, ind_terms = ind_frame_2)
> Warning: Option patch not set, so will set to first patch/population.
summary(trial2f_2)
>
> The input lefkoProj object covers 1 population-patches.
> It includes 21 projected steps per replicate and 1 replicates.
> The number of replicates with population size above the threshold size of 1 is as in
> the following matrix, with pop-patches given by column and milepost times given by row:
> $milepost_sums
        1 1
> 1 1
> 6 1
```

```
> 11 1
> 16 1
> 22 1
>
> $extinction_times
> [1] NA
```

Now let's plot our projections side-by-side (figure 10.20).

```
par(mfrow=c(1,2))
plot(trial2f_1, xlab = "Time from 2010")
title("a)", adj = 0)
plot(trial2f_2, xlab = "Time from 2070", ylab = "")
title("b)", adj = 0)
```



Figure 10.20: Deterministic function-based projections for (a) 2010-2030 and (b) 2070-2090
Clearly our different climate projections have different impacts on population dynamics. In particular, after an initial small lage, the population seems to grow in response to climate predicted for 2070 to 2090 than to that for 2010 to 2030.

As before, we can reset the starting numbers of individuals in different stages. Function f_projection3() assumes one individual per stage at the start by default, similarly to projection3(). As in projection3(), we can set the starting vector with the start_vec and start_frame options. Here we show an example of the latter, which is the more powerful approach. We will start the population with 1000 dormant seeds and $1001^{\text {st }}$ year protocorms, using the 2010 to 2030 climate projection (figure 10.21).Note that prior to using this option, we need to create a new MPM in the same format as our projection, as the start_input() function will need to look at the structure of the matrices and row / column indices to determine the proper vector.

```
trial_mpm <- flefko2(data = cypfb_env, stageframe = cypframe_fb,
    supplement = cypsupp2_fb, modelsuite = cypmodels2_env1, year = 2007)
c2m_sv <- start_input(trial_mpm, stage2 = c("SD", "P1"), value = c(1000, 100))
```

```
trial2f_3 <- f_projection3(data = cypfb_env, format = 3, stageframe = cypframe_fb,
    supplement = cypsupp2_fb, modelsuite = cypmodels2_env1, times = 21,
    year = 2007, ind_terms = ind_frame_1, start_frame = c2m_sv)
> Warning: Option patch not set, so will set to first patch/population.
par(mfrow = c(1, 2))
plot(trial2f_1)
title("a)", adj = 0)
plot(trial2f_3, ylab = "")
title("b)", adj = 0)
```



Figure 10.21: Deterministic function-based projections assuming different start frames. (a) Start with one individual of each stage. (b) Start with 1000 dormant seeds and 100 1st year protocorms

In the new projection, we start the population with dormant seeds and $1^{\text {st }}$ year protocorms only, which means that there is no reproduction in the population for the first few years. This causes an immediate drop in population size until some seeds and protocorms start reproducing. Once reproductive individuals enter the population, population size begins increasing.

### 10.3.2 Ordered / cyclical projections of a specific time length

The vital rate models developed with modelsearch() generally include year2 terms denoting time $t$. These are terms within a categorical variable, and so each year has its own value. If we do not specify a specific year to project, then f_projection3() will cycle through the years in the dataset in order for the number of time steps set, using the appropriate year2 term in each vital rate model used to estimate each matrix at each time step. Alternatively, if the year option is set to a vector of years, then these particular years will cycle in order for the number of time steps set. Note, however, that this function only sets the values for the categorical year2 term in the vital rate models - any related terms, such as individual covariates, are unaffected by this.

Let's take an example using three years in particular: 2006, 2007, and 2008. By choosing these years and loading this choice as a vector in the year option, we will have lefko3 cycle through them in order. However, because our vital rate models also include precipitation, we need to incorporate values for precipitation, and our choices will make a big difference. As in the year case, f_projection3() will cycle though whatever values we provide until the projection is finished. These cycles are independent of the year option cycles, and so care needs to be taken to load only values genuinely needed. Here, we will run three projections using our year vector of three years. Each one will assume a different precipitation input: the first will be a vector of six precipitation values, the second will be just the first of those values, and the third will be the mean of those six values.

```
ind_real <- data.frame(inda = c(92.2, 57.6, 96.0, 109.8, 111.9, 106.8),
    indb = 0, indc = 0)
short_real <- as.data.frame(ind_real[1,])
mean_real <- data.frame(inda = mean(ind_real$inda), indb = 0, indc = 0)
trial2f_4a <- f_projection3(data = cypfb_env, format = 3, stageframe = cypframe_fb,
    supplement = cypsupp2_fb, modelsuite = cypmodels2_env1, times = 101,
    year = c(2006, 2007, 2008), ind_terms = ind_real)
> Warning: Option patch not set, so will set to first patch/population.
> Warning: Fewer values of individual covariates have been supplied than required, so some will be cy
trial2f_4b <- f_projection3(data = cypfb_env, format = 3, stageframe = cypframe_fb,
    supplement = cypsupp2_fb, modelsuite = cypmodels2_env1, times = 101,
    year = c(2006, 2007, 2008), ind_terms = short_real)
> Warning: Option patch not set, so will set to first patch/population.
> Warning: Fewer values of individual covariates have been supplied than required, so some will be cy
trial2f_4c <- f_projection3(data = cypfb_env, format = 3, stageframe = cypframe_fb,
    supplement = cypsupp2_fb, modelsuite = cypmodels2_env1, times = 101,
    year = c(2006, 2007, 2008), ind_terms = mean_real)
> Warning: Option patch not set, so will set to first patch/population.
> Warning: Fewer values of individual covariates have been supplied than required, so some will be cy
summary(trial2f_4a)
>
> The input lefkoProj object covers 1 population-patches.
> It includes }101\mathrm{ projected steps per replicate and 1 replicates.
> The number of replicates with population size above the threshold size of 1 is as in
> the following matrix, with pop-patches given by column and milepost times given by row:
> $milepost_sums
> 1 1
> 1 1
> 26 1
> 51 1
> 76 1
> 102 1
>
> $extinction_times
> [1] NA
summary(trial2f_4b)
>
> The input lefkoProj object covers 1 population-patches.
```

```
> It includes }101\mathrm{ projected steps per replicate and 1 replicates.
> The number of replicates with population size above the threshold size of 1 is as in
> the following matrix, with pop-patches given by column and milepost times given by row:
> $milepost_sums
> 1 1
> 1 1
> 26 1
> 51 1
> 76 1
> 102 1
>
> $extinction_times
> [1] NA
summary(trial2f_4c)
>
> The input lefkoProj object covers 1 population-patches.
> It includes }101\mathrm{ projected steps per replicate and 1 replicates.
> The number of replicates with population size above the threshold size of 1 is as in
> the following matrix, with pop-patches given by column and milepost times given by row:
> $milepost_sums
> 1 1
> 1 1
> 26 1
> 51 1
> 76 1
> 102 1
>
> $extinction_times
> [1] NA
```

Here is another example using the full set of years instead of a subset, though assuming the same three precipitation vectors as the last runs.

```
trial2f_5a <- f_projection3(data = cypfb_env, format = 3, stageframe = cypframe_fb,
    supplement = cypsupp2_fb, modelsuite = cypmodels2_env1, times = 101,
    ind_terms = ind_real)
> Warning: Option year not set, so will cycle through existing years.
> Warning: Option patch not set, so will set to first patch/population.
> Warning: Fewer values of individual covariates have been supplied than required, so some will be cy
trial2f_5b <- f_projection3(data = cypfb_env, format = 3, stageframe = cypframe_fb,
    supplement = cypsupp2_fb, modelsuite = cypmodels2_env1, times = 101,
    ind_terms = short_real)
> Warning: Option year not set, so will cycle through existing years.
> Warning: Option patch not set, so will set to first patch/population.
> Warning: Fewer values of individual covariates have been supplied than required, so some will be cy
trial2f_5c <- f_projection3(data = cypfb_env, format = 3, stageframe = cypframe_fb,
    supplement = cypsupp2_fb, modelsuite = cypmodels2_env1, times = 101,
    ind_terms = mean_real)
> Warning: Option year not set, so will cycle through existing years.
> Warning: Option patch not set, so will set to first patch/population.
> Warning: Fewer values of individual covariates have been supplied than required, so some will be cy
```

```
summary(trial2f_5a)
>
> The input lefkoProj object covers 1 population-patches.
> It includes }101\mathrm{ projected steps per replicate and 1 replicates.
> The number of replicates with population size above the threshold size of 1 is as in
> the following matrix, with pop-patches given by column and milepost times given by row:
> $milepost_sums
> 1 1
> 1 1
> 26 1
> 51 1
> 76 1
> 102 1
>
> $extinction_times
> [1] NA
summary(trial2f_5b)
>
> The input lefkoProj object covers 1 population-patches.
> It includes }101\mathrm{ projected steps per replicate and 1 replicates.
> The number of replicates with population size above the threshold size of 1 is as in
> the following matrix, with pop-patches given by column and milepost times given by row:
> $milepost_sums
> 1 1
> 1 1
> 26 1
> 51 1
> 76 1
> 102 1
>
> $extinction_times
> [1] NA
summary(trial2f_5c)
>
> The input lefkoProj object covers 1 population-patches.
> It includes }101\mathrm{ projected steps per replicate and 1 replicates.
> The number of replicates with population size above the threshold size of 1 is as in
> the following matrix, with pop-patches given by column and milepost times given by row:
> $milepost_sums
> 1 1
> 1 1
> 26 1
> 51 1
> 76 1
> 102 1
>
> $extinction_times
> [1] NA
```

Let's plot the results of all six runs together (figure 10.22).

```
par(mfrow=c (3,2))
plot(trial2f_4a)
title("a)", adj = 0)
plot(trial2f_4b, ylab = "")
title("b)", adj = 0)
plot(trial2f_4c)
title("c)", adj = 0)
plot(trial2f_5a, ylab = "")
title("d)", adj = 0)
plot(trial2f_5b)
title("e)", adj = 0)
plot(trial2f_5c, ylab = "")
title("f)", adj = 0)
```



Figure 10.22: Cyclical function-based projections using 2007-2009 only (a, b, c), or all years (d, e, f, ), assuming all six real precip values (a, d), precip for 2004 only (b, e), or mean precip over 2004-2009 (d, f).

The choice of which years and individual covariate values led to similar projections, although plot (d)
suggests that the choice of using all years led to stronger population fluctuations in that case.
The approach above can be used to program any particular order of years. The year option should simply be paired with a vector of whatever length showing the order of year terms desired. Here, we use a sequence of years initially switching between years 2007 and 2008, then settling into 2008 for nine years, and ending on year 2005. Let's assume constant mean precipitation.

```
year_order <- c(2007, 2008, 2007, 2008, 2007, 2008, 2007, 2008, 2007, 2008,
    2008, 2008, 2008, 2008, 2008, 2008, 2008, 2008, 2008, 2005)
trial2f_6 <- f_projection3(data = cypfb_env, format = 3, stageframe = cypframe_fb,
    supplement = cypsupp2_fb, modelsuite = cypmodels2_env1, year = year_order,
    times = 20, ind_terms = mean_real)
> Warning: Option patch not set, so will set to first patch/population.
> Warning: Fewer values of individual covariates have been supplied than required, so some will be cy
summary(trial2f_6)
>
> The input lefkoProj object covers 1 population-patches.
> It includes 20 projected steps per replicate and 1 replicates.
> The number of replicates with population size above the threshold size of 1 is as in
> the following matrix, with pop-patches given by column and milepost times given by row:
> $milepost_sums
        11
    1 1
    6 1
    11 1
    16 1
    21 1
>
$extinction_times
[1] NA
```

As before, if we set a larger number for times than we will cycle through the sequence.

```
trial2f_7 <- f_projection3(data = cypfb_env, format = 3, stageframe = cypframe_fb,
    supplement = cypsupp2_fb, modelsuite = cypmodels2_env1, year = year_order,
    times = 100, ind_terms = mean_real)
> Warning: Option patch not set, so will set to first patch/population.
> Warning: Fewer values of individual covariates have been supplied than required, so some will be cy
summary(trial2f_7)
>
> The input lefkoProj object covers 1 population-patches.
> It includes }100\mathrm{ projected steps per replicate and 1 replicates.
> The number of replicates with population size above the threshold size of 1 is as in
> the following matrix, with pop-patches given by column and milepost times given by row:
> $milepost_sums
> 1 1
> 1 1
> 26 1
> 51 1
> 76 1
```

```
> 101 1
>
> $extinction_times
> [1] NA
```

No extinction happened in either case. Let's take a look at plots of these projections for further details (figure 10.23).

```
par(mfrow=c(1, 2))
plot(trial2f_6)
title("a)", adj= 0)
plot(trial2f_7, ylab = "")
title("b)", adj= 0)
```



Figure 10.23: Ordered progression of year terms used in projection. (a) Specific sequence of 20 years projected for 20 years. (b) Specific sequence of 20 years projected cyclically for 100 years.

The population appears to grow. When cycled, the year values cause small dips at roughly 20 year intervals, with large upswings as time goes on.

### 10.3.3 Replicated simulations assuming temporal stochasticity

Function f_projection3() can also run temporally stochastic projections and replicated simulations. In this case, randomness is included through random draws of year2 terms in the vital rate models, meaning that all other terms are unaffected and can be run in whatever order the user deems fit. Here we conduct a simple stochastic projection for 101 time steps in a single replicate, using the mean precipitation over the study period.

```
set.seed(42)
trial2f_8 <- f_projection3(data = cypfb_env, format = 3, stochastic = TRUE,
```

```
    stageframe = cypframe_fb, supplement = cypsupp2_fb,
    modelsuite = cypmodels2_env1, times = 101, ind_terms = mean_real)
> Warning: Option patch not set, so will set to first patch/population.
> Warning: Fewer values of individual covariates have been supplied than required, so some will be cy
summary(trial2f_8)
>
> The input lefkoProj object covers 1 population-patches.
> It includes }101\mathrm{ projected steps per replicate and 1 replicates.
> The number of replicates with population size above the threshold size of 1 is as in
> the following matrix, with pop-patches given by column and milepost times given by row:
> $milepost_sums
> 1 1
> 1 1
> 26 1
> 51 1
> 76 1
> 102 1
> $extinction_times
> [1] NA
```

Let's say that we wished to compare this particular scenario with one in which we double fecundity through artificial pollination. For this purpose, we will define a new supplement table in which we double our two routes of fecundity. Then we will run a projection with this new supplement table.

```
cypsupp2_fb_alt <- supplemental(stage3 = c("SD", "P1", "P2", "P3", "SL", "SL",
    "D", "V1", "V2", "V3", "SD", "P1"),
    stage2 = c("SD", "SD", "P1", "P2", "P3", "SL", "SL", "SL", "SL", "SL", "rep",
        "rep"),
    eststage3 = c(NA, NA, NA, NA, NA, NA, "D", "V1", "V2", "V3", NA, NA),
    eststage2 = c(NA, NA, NA, NA, NA, NA, "D", "D", "D", "D", NA, NA),
    givenrate =c(0.08, 0.1, 0.1, 0.1, 0.05, 0.05, NA, NA, NA, NA, NA, NA),
    multiplier = c(NA, NA, NA, NA, NA, NA, sl_mult, sl_mult, sl_mult, sl_mult,
        0.5 * seeds_per_fruit * 2, 0.5 * seeds_per_fruit * 2),
    type = c("S", "S", "S", "S", "S", "S", "S", "S", "S", "S", "R", "R"),
    stageframe = cypframe_fb, historical = FALSE)
set.seed(42)
trial2f_9 <- f_projection3(data = cypfb_env, format = 3, stochastic = TRUE,
    stageframe = cypframe_fb, supplement = cypsupp2_fb_alt,
    modelsuite = cypmodels2_env1, times = 101, ind_terms = mean_real)
> Warning: Option patch not set, so will set to first patch/population.
> Warning: Fewer values of individual covariates have been supplied than required, so some will be cy
summary(trial2f_9)
>
> The input lefkoProj object covers 1 population-patches.
> It includes }101\mathrm{ projected steps per replicate and 1 replicates.
> The number of replicates with population size above the threshold size of 1 is as in
> the following matrix, with pop-patches given by column and milepost times given by row:
> $milepost_sums
```

```
> 1 1
> 1 1
> 26 1
> 51 1
> 76 1
> 102 1
>
> $extinction_times
> [1] NA
```

Let's look at plots to compare the results (figure 10.24).

```
par(mfrow=c(1,2))
plot(trial2f_8)
title("a)", adj = 0)
plot(trial2f_9, ylab = "")
title("b)", adj = 0)
```



Figure 10.24: Stochastic function-based projections testing the impacts of natural fecundity values (a) vs. doubled fecundity (b)

It certainly appears that doubling fecundity has a positive impact, with higher spikes in numbers over time. Graphically, it appears that the population also goes extinct at a later time when fecundity is doubled. However, these are single stochastic replicates, and we have no idea what the level of uncertainty is. So, let's repeat the projections, but using 100 replicates each.

```
set.seed(42)
trial2f_10 <- f_projection3(data = cypfb_env, format = 3, stochastic = TRUE,
    stageframe = cypframe_fb, supplement = cypsupp2_fb,
    modelsuite = cypmodels2_env1, nreps = 100, times = 101,
```

```
    ind_terms = mean_real)
set.seed(42)
trial2f_11 <- f_projection3(data = cypfb_env, format = 3, stochastic = TRUE,
    stageframe = cypframe_fb, supplement = cypsupp2_fb_alt,
    modelsuite = cypmodels2_env1, nreps = 100, times = 101,
    ind_terms = mean_real)
```

Let's see some summaries.

```
summary(trial2f_10)
>
> The input lefkoProj object covers 1 population-patches.
> It includes }101\mathrm{ projected steps per replicate and 100 replicates.
> The number of replicates with population size above the threshold size of 1 is as in
> the following matrix, with pop-patches given by column and milepost times given by row:
> $milepost_sums
> 11
> 1 100
> 26 100
> 51 100
> 76 100
> 102 100
>
> $extinction_times
> [1] NA
summary(trial2f_11)
>
> The input lefkoProj object covers 1 population-patches.
> It includes }101\mathrm{ projected steps per replicate and }100\mathrm{ replicates.
> The number of replicates with population size above the threshold size of 1 is as in
> the following matrix, with pop-patches given by column and milepost times given by row:
> $milepost_sums
> 11
> 1 100
> 26 100
> 51 100
> 76 100
> 102 100
>
> $extinction_times
> [1] NA
```

Our summaries now include information on the numbers of replicates still alive at each milepost time. They did before as well, but we only ran single replicates before. Let's compare these results graphically (figure 10.25).

```
par(mfrow=c(1,2))
plot(trial2f_10)
plot(trial2f_11, ylab = "")
```



Figure 10.25: Stochastic function-based projections testing doubled fecundity with replicates

We see consistent population growth in both cases, but much greater when fecundity is doubled.
Let's try one more stochastic projection. This time, we will weight the matrices differently, with the 2007 matrix being chosen $88 \%$ of the time while the other matrices are chosen $3 \%$ of the time each.

```
set.seed(42)
trial2f_12 <- f_projection3(data = cypfb_env, format = 3, stochastic = TRUE,
    stageframe = cypframe_fb, supplement = cypsupp2_fb,
    modelsuite = cypmodels2_env1, nreps = 100, times = 101,
    ind_terms = mean_real, tweights = c(0.03, 0.03, 0.03, 0.88, 0.03))
```

And the summary....

```
summary(trial2f_12)
>
> The input lefkoProj object covers 1 population-patches.
> It includes }101\mathrm{ projected steps per replicate and 100 replicates.
> The number of replicates with population size above the threshold size of 1 is as in
> the following matrix, with pop-patches given by column and milepost times given by row:
> $milepost_sums
> 1 1
> 1 100
> 26 100
> 51 100
> 76 100
> 102 100
>
> $extinction_times
> [1] NA
```

Let's compare the new projection with the equivalent projection with equal year weights (figure 10.26).

```
par(mfrow = c(1, 2))
plot(trial2f_10)
title("a)", adj = 0)
plot(trial2f_12, ylab = "")
title("b)", adj = 0)
```



Figure 10.26: Stochastic function-based projections assuming (a) equal weights for all years, and (b) user-defined weights for different years

Assessing the difference in the y axus shows that the set of user-defined weights dropped the level of population growth to some extent.

### 10.3.4 Density dependent deterministic projections

Density dependent projections can run in two ways using function f_projection3(). The first involves setting density dependence relationships in matrix elements, just as is done with function projection3(). Users may use the two-parameter Ricker function, two-parameter Beverton-Holt function, the Usher function, and the logistic function, and need to specify which transitions to apply these to. Likewise, function f_projection3() can also enforce substochasticity, with both a mild setting forcing survival-transition elements to be in the range $[0,1]$, and a strict setting forcing survival-transition column sums to this range. Let's see a simple example. Remember that, as in start_input(), we need to provide density_input() with a MPM in with the same characteristics as the projection that we will run, so that the function can infer the proper stage distribution and produce the proper output vector.

```
cyp_ex <- flefko2(year = 2004, stageframe = cypframe_fb,
    supplement = cypsupp2_fb, modelsuite = cypmodels2_env1, data = cypfb_env)
c2d <- density_input(cyp_ex, stage3 = c("P1", "P1"),
    stage2 = c("SD", "rep"), style = 1, time_delay = 1, alpha = 1, beta = 0.0005,
```

```
    type = c(2, 2))
trial2f_13 <- f_projection3(data = cypfb_env, format = 3, stageframe = cypframe_fb,
    supplement = cypsupp2_fb, modelsuite = cypmodels2_env1, times = 101,
    year = 2007, ind_terms = mean_real, density = c2d)
> Warning: Option patch not set, so will set to first patch/population.
> Warning: Fewer values of individual covariates have been supplied than required, so some will be cy
summary(trial2f_13)
>
> The input lefkoProj object covers 1 population-patches.
> It includes }101\mathrm{ projected steps per replicate and 1 replicates.
> The number of replicates with population size above the threshold size of 1 is as in
> the following matrix, with pop-patches given by column and milepost times given by row:
> $milepost_sums
> 1 1
> 1 1
> 26 1
> 51 1
> 76 1
> 102 1
>
> $extinction_times
> [1] NA
```

We can compare the results to the previous projection assuming the year2 value for 2007, but without density dependence (figure 10.27). Note the extreme impact that density dependence has on the population trajectory, forcing a major drop to near 0 .

```
par(mfrow=c(1,2))
plot(trial2f_2)
title("a)", adj = 0)
plot(trial2f_13, ylab = "")
title("b)", adj = 0)
```

Function f_projection3() can also run density dependent simulations assuming that vital rates are density dependent. This situation allows us to modify the vital rates used to estimate the matrix elements. As before, it is possible to use the two-parameter Ricker function, the two-parameter Beverton-Holt function, the Usher function, or the logistic function. Let's try a simulation in which we make survival and fecundity density dependent, using the Ricker function for both. We will use the density_vr() function for this purpose.

```
vr1 <- density_vr(density_yn = c(T, F, F, F, F, F, T, F, F, F, F, F, F, F),
    style = c(1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0),
    alpha =c(1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0),
    beta = c(-0.0000000005, 0, 0, 0, 0, 0, 0.0000000003, 0, 0, 0, 0, 0, 0, 0))
trial2f_14 <- f_projection3(data = cypfb_env, format = 3, stageframe = cypframe_fb,
    supplement = cypsupp2_fb, modelsuite = cypmodels2_env1, times = 101,
```



Figure 10.27: Density independent (a) vs. dependent (b) function-based projections

```
year = 2007, ind_terms = mean_real, density_vr = vr1, substoch = 2)
> Warning: Option patch not set, so will set to first patch/population.
> Warning: Fewer values of individual covariates have been supplied than required, so some will be cy
summary(trial2f_14)
>
> The input lefkoProj object covers 1 population-patches.
> It includes }101\mathrm{ projected steps per replicate and 1 replicates.
> The number of replicates with population size above the threshold size of 1 is as in
> the following matrix, with pop-patches given by column and milepost times given by row:
> $milepost_sums
> 1 1
> 1 1
> 26 1
> 51 1
> 76 1
> 102 1
> $extinction_times
> [1] NA
```

In our density_vr() call, we need to set the density dependence relationships for 14 different vital rates that are used by the function-based matrix estimators in lefko3. The 14 models, in order, are: survival, observation, primary size, secondary size, tertiary size, reproduction status, and fecundity, followed by juvenile survival, juvenile observation, juvenile primary size, juvenile secondary size, juvenile tertiary size, juvenile reproduction status, and juvenile maturity status. The density_yn option stipulates with TRUE and FALSE statements which of these will be modified, and the style, alpha, beta, and time_delay option give the settings to use (last option not shown). Let's plot the results.

```
par(mfrow=c(1,2))
plot(trial2f_2)
title("a)", adj = 0)
plot(trial2f_14, ylab = "")
title("b)", adj = 0)
```



Figure 10.28: Density independent (a) vs. dependent (b) function-based projections with density dependent vital rates

We see a dramatic difference in projected population dynamics, resulting from two density dependent vital rates. Note also the contrast with our previous case of density dependence, which caused a population drop after an initial spike. Here, in contrast, the population is growing explosively.

### 10.3.5 Ordered / cyclical density dependent simulations

We can also use f_projection() to run density dependent projections cycling through a specific set of year2 terms. Here is an example cycling through all terms, which is the default, while using the mean precipitation value for all.

```
trial2f_15 <- f_projection3(data = cypfb_env, format = 3, stageframe = cypframe_fb,
    supplement = cypsupp2_fb, modelsuite = cypmodels2_env1, times = 101,
    ind_terms = mean_real, density = c2d)
> Warning: Option year not set, so will cycle through existing years.
> Warning: Option patch not set, so will set to first patch/population.
> Warning: Fewer values of individual covariates have been supplied than required, so some will be cy
summary(trial2f_15)
>
> The input lefkoProj object covers 1 population-patches.
> It includes }101\mathrm{ projected steps per replicate and 1 replicates.
```

```
> The number of replicates with population size above the threshold size of 1 is as in
> the following matrix, with pop-patches given by column and milepost times given by row:
> $milepost_sums
> 1 1
> 1 1
> 26 1
> 51 1
> 76 1
> 102 1
>
> $extinction_times
> [1] NA
```

Here is a further set cycling through just three particular years: 2005, 2006, and 2008.

```
trial2f_16 <- f_projection3(data = cypfb_env, format = 3, stageframe = cypframe_fb,
    supplement = cypsupp2_fb, modelsuite = cypmodels2_env1, times = 101,
    year = c(2005, 2006, 2008), ind_terms = mean_real, density = c2d)
> Warning: Option patch not set, so will set to first patch/population.
> Warning: Fewer values of individual covariates have been supplied than required, so some will be cy
summary(trial2f_16)
>
> The input lefkoProj object covers 1 population-patches.
> It includes }101\mathrm{ projected steps per replicate and 1 replicates.
> The number of replicates with population size above the threshold size of 1 is as in
> the following matrix, with pop-patches given by column and milepost times given by row:
> $milepost_sums
> 1 1
> 1 1
> 26 1
> 51 1
> 76 1
> 102 1
>
> $extinction_times
> [1] NA
```

As before, we can also run a specific ordered sequence of years. Here we use the previously defined 20 year vector, initially switching for ten years between 2007 and 2008 before heading on to nine years of 2008 and ending on 2005.

```
trial2f_17 <- f_projection3(data = cypfb_env, format = 3, stageframe = cypframe_fb,
    supplement = cypsupp2_fb, modelsuite = cypmodels2_env1, times = 20,
    year = year_order, ind_terms = mean_real, density = c2d)
> Warning: Option patch not set, so will set to first patch/population.
> Warning: Fewer values of individual covariates have been supplied than required, so some will be cy
summary(trial2f_17)
>
> The input lefkoProj object covers 1 population-patches.
> It includes 20 projected steps per replicate and 1 replicates.
```

```
> The number of replicates with population size above the threshold size of 1 is as in
> the following matrix, with pop-patches given by column and milepost times given by row:
> $milepost_sums
> 11
> 1 1
> 6 1
> 11 1
> 16 1
> 21 1
>
> $extinction_times
> [1] NA
```

Now let's compare the results (figure 10.29).

```
par(mfrow = c(2, 2))
plot(trial2f_15)
title("a)", adj = 0)
plot(trial2f_16, ylab = "")
title("b)", adj = 0)
plot(trial2f_17)
title("c)", adj = 0)
```

Extinction happens quickly in our two cyclical projections, though we seem to get there after a much lower high population size when cycling through year terms for all years in order. In contrast, our ordered projection seems more stable over the 20 projected years.

As before, we can use density dependence in vital rates even with a cyclical or ordered simulation. In that circumstance, the density option simply needs to be replaced with an appropriate density_vr input.

```
trial2f_18 <- f_projection3(data = cypfb_env, format = 3, stageframe = cypframe_fb,
    supplement = cypsupp2_fb, modelsuite = cypmodels2_env1, times = 20,
    year = year_order, ind_terms = mean_real, density_vr = vr1)
> Warning: Option patch not set, so will set to first patch/population.
> Warning: Fewer values of individual covariates have been supplied than required, so some will be cy
plot(trial2f_18)
```


### 10.3.6 Replicated simulations assuming temporal stochasticity and density dependence

Finally, we might conduct a density dependent stochastic projection. In this case, we will probably wish to run a simulation with replicates. Here is one such projection using the previous density frame.

```
set.seed(42)
trial2f_19 <- f_projection3(data = cypfb_env, format = 3, stochastic = TRUE,
    substoch = 2, stageframe = cypframe_fb, supplement = cypsupp2_fb,
```



Figure 10.29: Cyclical and ordered density dependent function-based projections. (a) Cycle of all 6 year values for 101 years; (b) cycle of year values for 2005, 2006, and 2008 for 101 years; and (c) ordered progression for 20 years.


Figure 10.30: Ordered function-based projection with vital rate density dependence.

```
modelsuite = cypmodels2_env1, times = 101, nreps = 100,
ind_terms = mean_real, density = c2d)
```

The summary....

```
summary(trial2f_19)
>
> The input lefkoProj object covers 1 population-patches.
> It includes }101\mathrm{ projected steps per replicate and }100\mathrm{ replicates.
> The number of replicates with population size above the threshold size of 1 is as in
> the following matrix, with pop-patches given by column and milepost times given by row:
> $milepost_sums
> 11
> 1 100
> 26 100
> 51 100
> 76 100
> 102 100
>
> $extinction_times
> [1] NA
```

Here is one further such projection, using a different density dependence relationship.

```
c2d1 <- density_input(cyp_ex, stage3 = c("P1", "P1"),
    stage2 = c("SD", "rep"), style = 1, time_delay = 1, alpha = 2, beta = 0.5,
    type = c(2, 2))
trial2f_20 <- f_projection3(data = cypfb_env, format = 3, stochastic = TRUE,
    substoch = 2, stageframe = cypframe_fb, supplement = cypsupp2_fb,
    modelsuite = cypmodels2_env1, times = 101, nreps = 100,
    ind_terms = mean_real, density = c2d1)
```

```
summary(trial2f_20)
>
> The input lefkoProj object covers 1 population-patches.
> It includes }101\mathrm{ projected steps per replicate and }100\mathrm{ replicates.
> The number of replicates with population size above the threshold size of 1 is as in
> the following matrix, with pop-patches given by column and milepost times given by row:
> $milepost_sums
> 1 1
> 1 100
> 26 100
> 51 100
> 76 100
> 102 100
>
> $extinction_times
> [1] NA
```

Let's take a peek at what the results look like (figure 10.30).

```
par(mfrow = c(1, 2))
plot(trial2f_19)
plot(trial2f_20, ylab = "")
```



Time


Time

Figure 10.31: Stochastic density dependent function-based projections, assuming two different sets of values for the Ricker function

The different values of the Ricker function have mot yielded dramatic differences here, although looking through the pop_size elements of each simulation will reveal large differences in predicted population size across years. We encourage the user to experiment further with different settings.

### 10.4 Points to remember

1. Although deterministic and stochastic analyses are very useful, sometimes more general tools are needed to assess population dynamics. Examples include density dependent analyses, projections under altered climatic regimes, and replicated simulations. For these situations, we use the general projection functions projection3() and f_projection3().
2. Function projection3() is the quickest simulation tool for most situations, because it uses pre-defined matrices and so does not create matrices at every step.
3. The greatest generality is available with function f_projection3(), because it creates new matrices at every time step based on input vital rate models and user-defined conditions.
4. Both projection functions include two substochasticity settings that can prevent impossible values, such as negative fecundity or survival probability above 1.0, from being incorporated into matrix projection.
5. Users may set density dependence in matrix elements or in vital rates, using the density_input () and density_vr() functions, respectively.

## Chapter 11

## Special Analyses I: Life Table Response Experiments

Karma police arrest this man He talks in maths, he buzzes like a fridge He's like a detuned radio. - Radiohead, Karma Police (1997)

A life table response experiment (LTRE) is a set of methodologies developed to analyze the dynamics of a population with respect to its vital rates (Caswell, 1989). The name refers to life tables, which were originally developed in the 17 th century by John Graunt and Edmund Halley to show the age-specific mortality probabilities, fecundity rates, and overall survivorship of a cohort over time, or a population at a slice in time (Greenwood, 1938). LTREs, as developed originally by Caswell (1989) and further developed by others, now typically use matrix projection models and discretized integral projection models to assess how population dynamics change as a function of some treatment or set of treatments, which may be observational factors or true experimental treatments.

Imagine that you work for a management agency, and your job is to develop a management plan for a rare organism. Perhaps you have attempted some experimental management on the population, splitting the population into a treatment portion and a control area. As you have monitored the population over time, you begin to note shifts in the population growth rate in the treatment area relative to the control. However, you may wish to know why the changes in population growth rate have occurred. At the coarse scale, you might imply that the treatment is responsible for the change (and, one would hope that this is indeed the case, rather than the shifts in population growth rate being due to some artefactual reason, such as biased environmental differences between the two areas). At a finer scale, you may wish to know what happened to the specific vital rates to cause population growth rate to change.

In a standard LTRE, the ultimate response term is the deterministic population growth rate, $\lambda$. The vital rates are the matrix elements in the MPM. LTREs may then be analyzed with treatments classified as fixed or random, nested or factorial, one-way or two-way (or more), or using a regression framework (Caswell, 2001). More recent developments have expanded LTREs to include stochastic analysis, where the response is the stochastic population growth rate, a (Caswell, 2010; Davison et al., 2010, 2013). In lefko3, we currently implement the exact one-way fixed deterministic LTRE (Caswell, 2001), the approximate stochastic LTRE (Davison et al., 2010), and the small-noise approximation LTRE (Davison et al., 2013, 2019) with plans to develop further methods later.

### 11.1 The one-way fixed LTRE

For this explanation of the method, we will use the notation provided by Caswell (2001). We will assume that $\mathbf{A}^{(M)}$ is a matrix corresponding to treatment $M$. We will also assume that one matrix
serves as a control or reference matrix, and this is given as $\mathbf{A}^{(R)}$. Under this scenario, we have one further matrix corresponding to the midway matrix $\mathbf{A}^{\dagger}$ (meaning that it is midway between treatment matrix $\mathbf{A}^{(M)}$ and control matrix $\mathbf{A}^{(R)}$ ). Note that below, lower-case $m$ corresponds to the number of treatment matrices when used by itself and not as a matrix symbol.

$$
\begin{equation*}
\mathbf{A}^{\dagger}=\frac{1}{2}\left(\mathbf{A}^{(M)}+\mathbf{A}^{(R)}\right) \tag{11.1}
\end{equation*}
$$

Using this notation, the asymptotic population growth rate $\lambda$ of some treatment $M$ can be approximated as the following.

$$
\begin{equation*}
\lambda^{(M)} \approx \lambda^{(R)}+\left.\sum_{i, j}\left(a_{i j}^{(M)}-a_{i j}^{(R)}\right) \frac{\partial \lambda}{\partial a_{i j}}\right|_{\mathbf{A}^{\dagger}} \tag{11.2}
\end{equation*}
$$

In equation @ref(eq:ltre_l), we see that the differences in respective matrix elements between the treatment matrix and the reference matrix are multiplied by the sensitivity of the asymptotic growth rate of the midway matrix to that element (we use the sensitivity of the midway matrix because it is more likely to give us a sense of the shape of the differences between matrices than simply using one of the two endpoint matrices). These products are referred to as the contributions of shifts in that element to shifts in $\lambda$, and are generally output in matrix format, for example as matrix $\mathbf{C}^{(\mathbf{M})}$ for treatment $M$. This matrix has the same dimensions and relates to the same respective transitions as the original treatment matrix. The contributions sum to the difference in $\lambda$ between reference and treatment matrices, and so may also be summed to assess, for example, how stasis, growth, shrinkage, and fecundity shift and contribute to the noted difference in population growth rate.

### 11.2 The approximate stochastic LTRE

The one-way fixed LTRE is quite useful, but is conducted with reference to the asymptotic growth rate and so cannot properly deal with temporal shifts in matrix elements. If the stochastic population growth rate, $a=\log \lambda_{S}$, is a more realistic measure of the population, and we have matrices developed across time for both a treatment and a control, then we may wish to perform a stochastic LTRE (sLTRE) in order to assess the impacts of changes in both the mean matrix elements on $a$ across time, and the variation in matrix elements on $a$ across time.

Since 2010, several stochastic life table response experiment (sLTREs) methodologies have been developed. Package lefko3 currently implements the Davison et al. (2010) approximate sLTRE. In contrast to the fixed one-way LTRE, the treatment and reference here are not single matrices, but sets of matrices that vary randomly across time. Here, we assume that $\log \lambda^{(M)}$ and $\log \lambda^{(R)}$ are the $\log$ stochastic growth rates of the treatment MPM and the reference MPM, respectively.

$$
\begin{equation*}
\Delta \log \lambda_{S}=\log \lambda^{(M)}-\log \lambda^{(R)} \approx \sum_{j, i}\left[\log \mu_{j i}^{(M)}-\log \mu_{j i}^{(R)}\right] E_{j i}^{\mu}+\left[\log \sigma_{j i}^{(M)}-\log \sigma_{j i}^{(R)}\right] E_{j i}^{\sigma} \tag{11.3}
\end{equation*}
$$

In the above equation, $E_{j i}^{\mu}$ and $E_{j i}^{\sigma}$ are the stochastic elasticities of the log stochastic growth rate to changes in the mean and standard deviation, respectively, of the element at row $j$ and column $i$ in the reference matrix set. These are the same stochastic elasticities as described in @ref(sec:stoch_elas). Internally, these are estimated using a simulated run in which the long-run properties of the projection are evaluated.

### 11.3 The small-noise approximation LTRE

The stochastic LTRE represents a large step forward in understanding the impacts of temporal environmental variability on demography. However, sLTREs do not assess the impacts of vital rate
correlations, which can be thought of as resulting from constraints on the variability and potentially the evolution of specific vital rates. Davison et al. (2013) developed an approach to deal with this issue that decomposes the difference in the stochastic population growth rate, $a$ between a treatment set and a reference set of matrices into contributions from shifts in the temporal means of matrix elements, the temporal variation in these elements, the elasticities of $\lambda$ to these elements, and temporal correlations between matrix elements. The procedure is quite different from the sLTRE procedure, which uses a simulation approach to estimate the stochastic growth rate, $a$. Instead, it uses Tuljapurkar's small noise approximation (Tuljapurkar, 1990), and then solves it analytically to produce the various contributions (Davison et al., 2013).

$$
\begin{equation*}
a^{(M)}=\log \lambda_{S}^{(M)} \approx r^{(M)}-\frac{1}{2} \sum_{i, j} \sum_{k, l}\left\{e_{i j} e_{k l} c_{i j} c_{k l} \rho_{i j, k l}\right\} \tag{11.4}
\end{equation*}
$$

In the above equation, $r^{(M)}=\log \lambda_{0}$, where $\lambda_{0}$ is the deterministic growth rate, or dominant eigenvalue, of the mean reference matrix. The symbols $e_{i j}$ and $c_{i j}$ refer to the elasticities of the mean treatment matrix and the scaled covariances (coefficients of variation) of the treatment matrices, respectively. Symbol $\rho_{i j, k l}$ refers to the temporal correlation matrix of the treatment matrices.

The difference in the stochastic growth rate between treatment and reference is then given as the following
$a^{(M)}-a^{(R)} \approx r^{(M)}-r^{(R)}-\frac{1}{2}\left(\sum_{k, l}\left\{e_{i j} e_{k l} c_{i j} c_{k l} \rho_{i j, k l}\right\}^{(M)}-\sum_{k, l}\left\{e_{i j} e_{k l} c_{i j} c_{k l} \rho_{i j, k l}\right\}^{(R)}\right)=\Delta r-\frac{1}{2} \Delta\left(\sum_{k, l}\left\{e_{i j} e_{k l} c_{i j} c_{k l} \rho_{i j, k l}\right\}\right)$
Using the Kitagawa decomposition, we find that the contributions to the difference in the stochastic growth rate from shifts in the means, elasticities, variances, and correlations of elements decompose to the following.

$$
\begin{gather*}
C_{i j}^{\mu}=r^{(M)}-r^{(R)}=\sum_{i j}\left(\frac{\Delta \mu_{i j}^{(M)}}{\mu_{i j}^{(R)}}\right) e_{i j}^{(R)}  \tag{11.6}\\
C_{i j}^{e}=-\frac{1}{2} \sum_{k l}\left[\overline{c_{i j} c_{k l} \rho_{i j, k l}}\right] \Delta\left\{e_{i j} e_{k l}\right\}  \tag{11.7}\\
C_{i j}^{\sigma}=-\frac{1}{2} \sum_{k l}\left[\overline{e_{i j} e_{k l}}\right]\left[\overline{\rho_{i j, k l}}\right] \Delta\left\{c_{i j} c_{k l}\right\}  \tag{11.8}\\
C_{i j, k l}^{\rho}=-\frac{1}{2}\left[\overline{e_{i j} e_{k l}}\right]\left[\overline{c_{i j} c_{k l}}\right] \Delta\left\{\overline{\rho_{i j, k l}}\right\} \tag{11.9}
\end{gather*}
$$

These contributions are additive. Thus, the full contribution of differences in means is given as $C^{\mu}=$ $\sum_{i, j} C_{i j}^{\mu}$, the full contribution of differences in elasticities is given as $C^{e}=\sum_{i, j} C_{i j}^{e}$, the full contribution of differences in variability is given as $C^{\sigma}=\sum_{i, j} C_{i j}^{\sigma}$, and the full contribution of differences in correlations is given as $C^{\rho}=\sum_{i, j} C_{i j}^{\rho}$. The latter three terms may be summed to give the contribution of stochastic effects $\left(C^{e}+C^{\sigma}+C^{\rho}\right)$, while the contribution of covariance is given as $C^{\sigma}+C^{\rho}$. As some changes will have negative effects while others have positive effects, analyses can also be performed looking at the magnitude of these differences in both directions.

### 11.4 Example analysis with Anthyllis vulneraria

Next we will illustrate the one-way fixed LTRE, the stochastic LTRE, and the small noise approximation LTRE using the example published by Davison et al. (2010). That paper focused on nine
natural populations of the perennial herb Anthyllis vulneraria, also known as kidney vetch, occurring in calcareous grasslands in the Viroin Valley of southwestern Belgium and monitored from 2003 to 2006. The published analysis included a stochastic LTRE. We provide a description of the plant and populations in 1.8.3. The plant is a short-lived, rosette-forming legume with a complex life cycle including stasis and retrogression between four stages but no seedbank (seedlings, juveniles, small adults and large adults; Figure 11.1).


Figure 11.1: Life history model of Anthyllis vulneraria. Solid arrows indicate survival transitions while dashed arrows indicate fecundity transitions.
\#\#\# Life history model development and data organization
We will first describe the life history characterizing the dataset, matching it to our analyses with a stageframe. Since we do not have the original demographic dataset that produced the published matrices, we do not need to know the exact sizes of plants and so will use proxy values. These proxy values need to be unique and non-negative, and they need non-overlapping bins usable as size classes defining each stage. However, since we are not analyzing size itself here, they do not need any further basis in reality. Other characteristics must be exact and realistic to make sure that the analyses work properly, including all other stage descriptions such as reproductive status, propagule status, and observation status.

```
sizevector <- c(1, 1, 2, 3) # These sizes are not from the original paper
stagevector <- c("Sdl", "Veg", "SmFlo", "LFlo")
repvector <- c(0, 0, 1, 1)
obsvector <- c(1, 1, 1, 1)
matvector <- c(0, 1, 1, 1)
immvector <- c(1, 0, 0, 0)
propvector <- c(0, 0, 0, 0)
indataset <- c(1, 1, 1, 1)
```

```
binvec <- c(0.5, 0.5, 0.5, 0.5)
comments <- c("Seedling", "Vegetative adult", "Small flowering",
    "Large flowering")
anthframe <- sf_create(sizes = sizevector, stagenames = stagevector,
    repstatus = repvector, obsstatus = obsvector, matstatus = matvector,
    immstatus = immvector, indataset = indataset, binhalfwidth = binvec,
    propstatus = propvector, comments = comments)
anthframe
\begin{tabular}{lrrrrrrrrr}
\(>\) & stage size & size_b & size_c min_age max_age \\
\(>\) & 1 & Sdl & 1 & \(N A\) & \(N A\) & \(N A\) & \(N A\) & 0 & 1
\end{tabular}
    sizebin_center sizebin_width binhalfwidthb_raw sizebinb_min sizebinb_max
1 1 N NA NA NA
>2 1 1 NA NA NA
> 2 2 N NA NA NA
>4 3 1 NA NA NA
> sizebinb_center sizebinb_width binhalfwidthc_raw sizebinc_min sizebinc_max
> NA NA NA NA NA
2 NA NA NA NA NA
3 NA NA NA NA NA
4 NA NA NA NA NA
sizebinc_center sizebinc_width group comments
1 NA NA 0 Seedling
2 ~ N A ~ N A ~ O ~ V e g e t a t i v e ~ a d u l t ~
3 NA NA 0 Small flowering
4 NA NA 0 Large flowering
```

Next, we will load data for this vignette, then take a look at a summary.

```
data(anthyllis)
summary(anthyllis)
>
> This ahistorical lefkoMat object contains 27 matrices.
>
> Each matrix is square with 4 rows and columns, and a total of 16 elements.
A total of 167 survival transitions were estimated, with 6.185 per matrix.
A total of 48 fecundity transitions were estimated, with 1.778 per matrix.
This lefkoMat object covers 1 population, }9\mathrm{ patches, and 3 time steps.
This lefkoMat object appears to have been imported. Number of unique individuals and transitions no
Survival probability sum check (each matrix represented by column in order):
> [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11]
```

```
> Min. 0.0667 0.500 0.0625 0.0769 0.0909 0.000 0.0000 0.333 0.171 0.000 0.0000
> 1st Qu. 0.0810 0.575 0.1708 0.1192 0.1334 0.075 0.0375 0.405 0.268 0.170 0.0000
> Median 0.1198 0.657 0.2106 0.3077 0.2276 0.100 0.1706 0.453 0.350 0.239 0.0000
> Mean 0.1599 0.637 0.2209 0.4231 0.2303 0.175 0.1895 0.469 0.320 0.203 0.0556
> 3rd Qu. 0.1987 0.720 0.2607 0.6116 0.3245 0.200 0.3225 0.517 0.402 0.271 0.0556
> Max. 0.3333 0.736 0.4000 1.0000 0.3750 0.500 0.4167 0.636 0.409 0.333 0.2222
> [,12] [,13] [,14] [,15] [,16] [,17] [,18] [,19] [,20] [,21] [,22]
> Min. 0.000 0.0000 0.000 0.0370 0.333 0.000 0.238 0.00 0.000 0.000 0.000
> 1st Qu. 0.375 0.0000 0.193 0.0408 0.394 0.381 0.310 0.00 0.500 0.000 0.125
> Median 0.500 0.0163 0.313 0.0657 0.564 0.518 0.410 0.00 0.833 0.250 0.271
> Mean 0.409 0.0510 0.281 0.0705 0.565 0.420 0.388 0.25 0.667 0.304 0.260
> 3rd Qu. 0.534 0.0673 0.401 0.0954 0.736 0.557 0.488 0.25 1.000 0.554 0.406
>Max. 0.636 0.1714 0.500 0.1136 0.800 0.645 0.493 1.00 1.000 0.714 0.500
> [,23] [,24] [,25] [,26] [,27]
> Min. 0.000 0.000 0.0000 0.000 0.0000
> 1st Qu. 0.500 0.250 0.0000 0.375 0.0833
> Median 0.762 0.333 0.0833 0.583 0.2556
> Mean 0.631 0.323 0.0833 0.542 0.3153
> 3rd Qu. 0.893 0.406 0.1667 0.750 0.4875
> Max. 1.000 0.625 0.1667 1.000 0.7500
```

We have 27 matrices, each four rows by four columns. These are raw matrices, so they are not completely full where matrix elements are possible. Let's take a look at the first such matrix as an example.

```
anthyllis$A[[1]]
> [,1] [,2] [,3] [,4]
> [1,] 0.00000000 0.00000000 1.74000000 1.74000000
> [2,] 0.20833333 0.00000000 0.00000000 0.05714286
> [3,] 0.04166667 0.07692308 0.00000000 0.00000000
> [4,] 0.08333333 0.07692308 0.06666667 0.02857143
```

This is an A matrix, meaning that it includes all survival-transitions and fecundity for population C as a whole over the interval 2003 to 2004. The corresponding $U$ and $F$ matrices were not provided in that paper, although it is most likely that the elements valued at 1.74 in the top right-hand corner are only composed of fecundity values while the rest of the matrix is only composed of survival transitions (this might not be the case if clonal reproduction were possible). The order of rows and columns corresponds to the order of stages in the stageframe anthframe.

Our anthyllis dataset is a full lefkoMat object, so it already includes a stage frame (in other words, we did not really need to make object anthframe). This is included as object ahstages within its structure. It also includes a labels element that shows the order of matrices by population, patch, and time. Let's take a look at that object here.

```
anthyllis
> $A
> $A[[1]]
> [,1] [,2] [,3] [,4]
> [1,] 0.00000000 0.00000000 1.74000000 1.74000000
> [2,] 0.20833333 0.00000000 0.00000000 0.05714286
> [3,] 0.04166667 0.07692308 0.00000000 0.00000000
> [4,] 0.08333333 0.07692308 0.06666667 0.02857143
>
```

```
> $A[[2]]
> [,1] [,2] [,3] [,4]
> [1,] 0.0000000 0.0000000 0.3 0.6
> [2,] 0.3218391}00.1428571 0.0 0.0
> [3,] 0.1609195 0.2857143 0.0 0.0
> [4,] 0.2528736 0.2857143 0.5 0.6
>
> $A[[3]]
> [,1] [,2] [,3] [,4]
> [1,] 0.0 0.00000000 0.50625 0.67500000
> [2,] 0.0 0.00000000 0.00000 0.03571429
> [3,] 0.1 0.06896552 0.06250}00.1071428
> [4,] 0.3 0.13793103 0.00000 0.07142857
>
> $A[[4]]
> [,1] [,2] [,3] [,4]
> [1,] 0.0000000 0.0 2.4400000 6.56923077
> [2,] 0.1964286 0.0 0.0000000 0.00000000
> [3,] 0.1250000 0.5 0.0000000 0.00000000
> [4,] 0.1607143 0.5 0.1333333 0.07692308
>
> $A[[5]]
> [,1] [,2] [,3] [,4]
> [1,] 0.00000000 0.0000000000.450 0.64615385
> [2,] 0.06557377 0.09090909 0.125 0.00000000
> [3,] 0.03278689 0.00000000 0.125 0.07692308
> [4,] 0.04918033 0.00000000 0.125 0.23076923
>
> $A[[6]]
> [,1] [,2] [,3] [,4]
> [1,] 0.00000000 0.0 2.85 3.99
> [2,] 0.08333333 0.0 0.00 0.00
> [3,] 0.00000000 0.0 0.00 0.00
> [4,] 0.41666667 0.1 0.00 0.10
>
> $A[[7]]
> [,1] [,2] [,3] [,4]
> [1,] 0.00000000 0 1.815 7.05833333
> [2,] 0.07594937 0 0.050 0.08333333
> [3,] 0.13924051 0 0.000 0.25000000
> [4,] 0.07594937 0 0.000 0.08333333
>
> $A[[8]]
> [,1] [,2] [,3] [,4]
> [1,] 0.0000000 0.0000000 1.23333333 7.4000000
> [2,] 0.2238806 0.0000000 0.111111111 0.1428571
> [3,] 0.1343284 0.2727273 0.16666667 0.1428571
> [4,] 0.1194030 0.3636364 0.05555556 0.1428571
>
> $A[[9]]
> [,1] [,2] [,3] [,4]
```

```
> [1,] 0.00000000 0.000 1.06000000 3.3727273
> [2,] 0.07317073 0.025 0.03333333 0.0000000
> [3,] 0.03658537 0.150 0.10000000 0.1363636
> [4,] 0.06097561 0.225 0.16666667 0.2727273
>
> $A[[10]]
    [,1] [,2] [,3] [,4]
> [1,] 0.000 0 0.24545454 2.1000000
> [2,] 0.000 0 0.04545454 0.0000000
> [3,] 0.125 0 0.09090909 0.0000000
> [4,] 0.125 0 0.09090909 0.3333333
>
> $A[[11]]
> [,1] [,2] [,3] [,4]
> [1,] 0.0000000 0 1.1 1.54
> [2,] 0.11111111 0 0.0 0.00
> [3,] 0.0000000 0 0.0 0.00
> [4,] 0.11111111 0 0.0 0.00
>
> $A[[12]]
> [,1] [,2] [,3] [,4]
> [1,] 0.00000000 0.0 0 1.5
> [2,] 0.00000000 0.0 0 0.0
> [3,] 0.09090909 0.0 0}00.
> [4,] 0.54545455 0.5 0}00.
>
> $A[[13]]
> [,1] [,2] [,3] [,4]
> [1,] 0.00000000 0 1.785366 1.85652174
> [2,] 0.12857143 0 0.000000 0.01086956
> [3,] 0.02857143 0 0.000000 0.00000000
> [4,] 0.01428571 0 0.000000 0.02173913
>
> $A[[14]]
> [,1] [,2] [,3] [,4]
> [1,] 0.00000000 0.00000000 14.25 16.625
> [2,] 0.13144330}00.05714286 0.00 0.250
> [3,] 0.14432990 0.00000000 0.00 0.000
> [4,] 0.09278351 0.20000000 0.00 0.250
>
> $A[[15]]
> [,1] [,2] [,3] [,4]
> [1,] 0.00000000 0.00000000 0.59464286 1.76590909
> [2,] 0.00000000 0.00000000 0.01785714 0.00000000
> [3,] 0.02105263 0.01851852 0.03571429 0.04545454
> [4,] 0.02105263 0.01851852 0.03571429 0.06818182
>
> $A[[16]]
    [,1] [,2] [,3] [,4]
> [1,] 0.00 0.0000000 11.5000000 2.7758621
> [2,] 0.60 0.2857143 0.3333333 0.2413793
```

```
> [3,] 0.04 0.1428571 0.0000000 0.0000000
> [4,] 0.16 0.2857143 0.0000000 0.1724138
>
> $A[[17]]
> [,1] [,2] [,3] [,4]
> [1,] 0.00000000 0.00000000 3.78 1.22500000
> [2,] 0.28358209 0.17105263 0.00 0.16666667
> [3,] 0.08457711 0.02631579 0.00 0.05555556
> [4,] 0.13930348 0.44736842 0.00 0.30555556
>
> $A[[18]]
> [,1] [,2] [,3] [,4]
> [1,] 0.00000000 0.0000000 1.54285714 1.03561644
> [2,] 0.12698413 0.1052632 0.04761905 0.05479452
> [3,] 0.09523809 0.1578947 0.19047619 0.08219178
> [4,] 0.11111111 0.2236842 0.00000000 0.35616438
>
> $A[[19]]
> [,1] [,2] [,3] [,4]
> [1,] 0 0 0.15 0.175
> [2,] 0 0 0.00 0.000
> [3,] 0 0 0.00 0.000
> [4,] 1 0 0.00 0.000
>
> $A[[20]]
> [,1] [,2] [,3] [,4]
> [1,] 0 0.0000000 0 0.25
> [2,] 0 0.0000000 00.00
> [3,] 0 0.0000000 0 0.00
> [4,] 1 0.6666667 0 1.00
>
> $A[[21]]
> [,1] [,2] [,3] [,4]
> [1,] 0.00 0 0 1.4285714
> [2,] 0.00 0 0 0.1428571
> [3,] 0.25 0 0 0.0000000
> [4,] 0.25 0 0 0.5714286
>
> $A[[22]]
> [,1] [,2] [,3] [,4]
> [1,] 0.00 0.0000000 0.7 0.6125
> [2,] 0.25 0.0000000 0.0 0.1250
> [3,] 0.00 0.0000000 0.0 0.0000
> [4,] 0.25 0.1666667 0.0 0.2500
>
> $A[[23]]
> [,1] [,2] [,3] [,4]
> [1,] 0.0000000 0.0000000 0 0.6
> [2,] 0.2857143 0.0000000 0 0.0
> [3,] 0.2857143 0.3333333 0 0.0
> [4,] 0.2857143 0.3333333 0 1.0
```

```
>
> $A[[24]]
> [,1] [,2] [,3] [,4]
> [1,] 0.0000000 0 0.7000000 0.6125
> [2,] 0.0000000 0 0.0000000 0.0000
> [3,] 0.0000000 0 0.0000000 0.0000
> [4,] 0.3333333 0 0.3333333 0.6250
>
> $A[[25]]
> [,1] [,2] [,3] [,4]
> [1,] 0.0000000 0 2.1 0.8166667
> [2,] 0.1666667 0}0.00.000000
> [3,] 0.0000000 0 0.0 0.0000000
> [4,] 0.0000000 0}00.00.166666
>
> $A[[26]]
> [,1] [,2] [,3] [,4]
> [1,] 0.0000000 0.0 0 7
> [2,] 0.3333333 0.5 0 0
> [3,] 0.0000000 0.0 0 0
> [4,] 0.3333333
>
> $A[[27]]
> [,1] [,2] [,3] [,4]
> [1,] 0.0000000 0.00 0 1.4
> [2,] 0.0000000 0.00 0}00.
> [3,] 0.0000000 0.00 0}00.
> [4,] 0.11111111 0.75 0}00.
>
>
> $U
$U[[1]]
> [,1] [,2] [,3] [,4]
> [1,] 0.0000000000.00000000 0.00000000 0.00000000
> [2,] 0.20833333 0.00000000 0.00000000 0.05714286
> [3,] 0.04166667 0.07692308 0.00000000 0.00000000
> [4,] 0.08333333 0.07692308 0.06666667 0.02857143
>
> $U[[2]]
> [,1] [,2] [,3] [,4]
> [1,] 0.0000000 0.0000000 0.0 0.0
> [2,] 0.3218391 0.1428571 0.0
> [3,] 0.1609195 0.2857143 0.0 0.0
> [4,] 0.2528736 0.2857143 0.5 0.6
>
> $U[[3]]
> [,1] [,2] [,3] [,4]
> [1,] 0.0 0.00000000 0.0000 0.00000000
> [2,] 0.0 0.00000000 0.0000 0.03571429
> [3,] 0.1 0.06896552 0.0625 0.10714286
> [4,] 0.3 0.13793103 0.0000 0.07142857
```

```
>
> $U[[4]]
> [,1] [,2] [,3] [,4]
[1,] 0.0000000 0.0 0.0000000 0.00000000
> [2,] 0.1964286 0.0 0.0000000 0.00000000
> [3,] 0.1250000 0.5 0.0000000 0.00000000
> [4,] 0.1607143 0.5 0.1333333 0.07692308
>
> $U[[5]]
> [,1] [,2] [,3] [,4]
> [1,] 0.00000000 0.00000000 0.000 0.00000000
> [2,] 0.06557377 0.09090909 0.125 0.00000000
> [3,] 0.03278689 0.00000000 0.125 0.07692308
> [4,] 0.04918033 0.00000000 0.125 0.23076923
>
> $U[[6]]
> [,1] [,2] [,3] [,4]
> [1,] 0.00000000 0.0 0 0.0
> [2,] 0.08333333 0.0 0 0.0
> [3,] 0.00000000 0.0 0 0.0
> [4,] 0.41666667
>
> $U[[7]]
> [,1] [,2] [,3] [,4]
> [1,] 0.00000000 0 0.00 0.00000000
> [2,] 0.07594937 0 0.05 0.08333333
> [3,] 0.13924051 0 0.00 0.25000000
> [4,] 0.07594937 0 0.00 0.08333333
>
> $U[[8]]
> [,1] [,2] [,3] [,4]
> [1,] 0.0000000 0.0000000 0.00000000 0.0000000
> [2,] 0.2238806 0.0000000 0.111111111 0.1428571
> [3,] 0.1343284 0.2727273 0.16666667 0.1428571
> [4,] 0.1194030 0.3636364 0.05555556 0.1428571
>
> $U[[9]]
> [,1] [,2] [,3] [,4]
> [1,] 0.00000000 0.000 0.00000000 0.0000000
> [2,] 0.07317073 0.025 0.03333333 0.0000000
> [3,] 0.03658537 0.150 0.10000000 0.1363636
> [4,] 0.06097561 0.225 0.16666667 0.2727273
>
> $U[[10]]
> [,1] [,2] [,3] [,4]
> [1,] 0.000 0 0.00000000 0.0000000
> [2,] 0.000 0 0.04545454 0.0000000
> [3,] 0.125 0 0.09090909 0.00000000
> [4,] 0.125 0 0.09090909 0.3333333
>
> $U[[11]]
```

```
> [,1] [,2] [,3] [,4]
> [1,] 0.0000000 0 0 0
> [2,] 0.1111111 
> [3,] 0.0000000 0 0 0
> [4,] 0.1111111 0}0
>
> $U[[12]]
> [,1] [,2] [,3] [,4]
> [1,] 0.00000000 0.0 0 0.0
> [2,] 0.00000000 0.0 0 0.0
> [3,] 0.09090909 0.0 0}00.
> [4,] 0.54545455 0.5 0}0.
>
> $U[[13]]
> [,1] [,2] [,3] [,4]
> [1,] 0.00000000 0 0 0.00000000
> [2,] 0.12857143 0 0 0.01086956
> [3,] 0.02857143 0 0 0.00000000
> [4,] 0.01428571 0 0 0.02173913
>
> $U[[14]]
> [,1] [,2] [,3] [,4]
> [1,] 0.00000000 0.00000000 0 0.00
> [2,] 0.13144330 0.05714286 0 0.25
> [3,] 0.14432990 0.00000000 0 0.00
> [4,] 0.09278351 0.20000000 0 0.25
>
> $U[[15]]
> [,1] [,2] [,3] [,4]
> [1,] 0.00000000 0.00000000 0.00000000 0.00000000
> [2,] 0.00000000 0.00000000 0.01785714 0.00000000
> [3,] 0.02105263 0.01851852 0.03571429 0.04545454
> [4,] 0.02105263 0.01851852 0.03571429 0.06818182
>
> $U[[16]]
> [,1] [,2] [,3] [,4]
> [1,] 0.00 0.0000000 0.0000000 0.0000000
> [2,] 0.60 0.2857143 0.3333333 0.2413793
> [3,] 0.04 0.1428571 0.0000000 0.0000000
> [4,] 0.16 0.2857143 0.0000000 0.1724138
>
> $U[[17]]
> [,1] [,2] [,3] [,4]
> [1,] 0.00000000 0.00000000 0 0.00000000
> [2,] 0.28358209 0.17105263 0 0.16666667
> [3,] 0.08457711 0.02631579 0 0.05555556
> [4,] 0.13930348 0.44736842 0 0.30555556
>
> $U[[18]]
> [,1] [,2] [,3] [,4]
> [1,] 0.00000000 0.0000000 0.00000000 0.00000000
```

```
> [2,] 0.12698413 0.1052632 0.04761905 0.05479452
> [3,] 0.09523809 0.1578947 0.19047619 0.08219178
> [4,] 0.11111111 0.2236842 0.00000000 0.35616438
>
> $U[[19]]
> [,1] [,2] [,3] [,4]
> [1,] 0 0 0 0
> [2,] 0
> [3,] 0
>[4,] 1 [ 0 0 0
>
> $U[[20]]
> [,1] [,2] [,3] [,4]
> [1,] 0 0.0000000 0 0
> [2,] 0 0.0000000 0 0
> [3,] 0 0.0000000 0 0
> [4,] 1 0.6666667 0 1
>
> $U[[21]]
> [,1] [,2] [,3] [,4]
> [1,] 0.00 0 0 0.0000000
> [2,] 0.00 0 0 0.1428571
> [3,] 0.25 0 0 0.0000000
> [4,] 0.25 0 0 0.5714286
>
> $U[[22]]
> [,1] [,2] [,3] [,4]
> [1,] 0.00 0.0000000 0 0.000
> [2,] 0.25 0.0000000 0 0.125
> [3,] 0.00 0.0000000 0 0.000
> [4,] 0.25 0.1666667 0 0.250
>
> $U[[23]]
> [,1] [,2] [,3] [,4]
> [1,] 0.0000000 0.0000000 0 0
> [2,] 0.2857143 0.0000000 0 0
> [3,] 0.2857143 0.3333333 0 0
> [4,] 0.2857143 0.3333333 0 0 1
>
> $U[[24]]
> [,1] [,2] [,3] [,4]
> [1,] 0.0000000 0 0.0000000 0.000
> [2,] 0.0000000 0 0.0000000 0.000
> [3,] 0.0000000 0 0.0000000 0.000
> [4,] 0.3333333 0 0.3333333 0.625
>
> $U[[25]]
> [,1] [,2] [,3] [,4]
> [1,] 0.0000000 0 0 0.0000000
> [2,] 0.1666667 0 0 0.0000000
> [3,] 0.0000000 0 0 0.0000000
```

```
> [4,] 0.0000000 0 0 0.1666667
>
> $U[[26]]
> [,1] [,2] [,3] [,4]
> [1,] 0.0000000 0.0 0 0
> [2,] 0.3333333 0.5 0 0
> [3,] 0.0000000 0.0 0 0
> [4,] 0.3333333 0.0 0 1
>
> $U[[27]]
> [,1] [,2] [,3] [,4]
> [1,] 0.0000000 0.00 0 0.0
> [2,] 0.0000000 0.00 0}00.
> [3,] 0.0000000 0.00 0
> [4,] 0.1111111 0.75 0
>
>
> $F
> $F[[1]]
> [,1] [,2] [,3] [,4]
> [1,] 0 0 1.74 1.74
> [2,] 0 0 0.00 0.00
> [3,] 0 0 0.00 0.00
> [4,] 0 0 0.00 0.00
>
> $F[[2]]
> [,1] [,2] [,3] [,4]
> [1,] 0 0 0.3 0.6
> [2,] 0}0
> [3,] 0}0
> [4,] 0 0}00.
>
> $F[[3]]
> [,1] [,2] [,3] [,4]
> [1,] 0 0 0.50625 0.675
> [2,] 0 0 0.00000 0.000
> [3,] 0 0 0.00000 0.000
> [4,] 0 0 0.00000 0.000
>
> $F[[4]]
> [,1] [,2] [,3] [,4]
> [1,] 0 0 2.44 6.569231
> [2,] 0 0 0.00 0.000000
> [3,] 0 0 0.00 0.000000
> [4,] 0 0 0.00 0.000000
>
> $F[[5]]
> [,1] [,2] [,3] [,4]
> [1,] 0 0 0.45 0.6461538
> [2,] 0 0 0.00 0.0000000
> [3,] 0 0 0.00 0.0000000
```

```
> [4,] 0 0 0.00 0.0000000
>
> $F[[6]]
> [,1] [,2] [,3] [,4]
> [1,] 0 0 2.85 3.99
> [2,] 0 0 0.00 0.00
> [3,] 0 0 0.00 0.00
> [4,] 0 0 0.00 0.00
>
> $F[[7]]
> [,1] [,2] [,3] [,4]
> [1,] 0 0 1.815 7.058333
> [2,] 0 0 0.000 0.000000
> [3,] 0 0 0.000 0.000000
> [4,] 0 0 0.000 0.000000
>
> $F[[8]]
> [,1] [,2] [,3] [,4]
> [1,] 0 0 1.233333 7.4
> [2,] 0 0 0.000000 0.0
> [3,] 0 0 0.000000 0.0
> [4,] 0 0 0.000000 0.0
>
> $F[[9]]
> [,1] [,2] [,3] [,4]
> [1,] 0 0 1.06 3.372727
> [2,] 0 0 0.00 0.000000
> [3,] 0 0 0.00 0.000000
> [4,] 0 0 0.00 0.000000
>
> $F[[10]]
> [,1] [,2] [,3] [,4]
> [1,] 0 0 0.2454545 2.1
> [2,] 0 0 0.0000000 0.0
> [3,] 0}000.0000000 0.
> [4,] 0 0 0.0000000 0.0
>
$F[[11]]
> [,1] [,2] [,3] [,4]
> [1,] 0 0 1.1 1.54
> [2,] 0 0 0.0 0.00
> [3,] 0 0 0.0 0.00
> [4,] 0 0 0.0 0.00
>
> $F[[12]]
> [,1] [,2] [,3] [,4]
> [1,] 0 0 0 1.5
> [2,] 0 0 0 0.0
> [3,] 0
> [4,] 0 0 0}00.
>
```

```
> $F[[13]]
[,1] [,2] [,3] [,4]
> [1,] 0 0 1.785366 1.856522
> [2,] 0 0 0.000000 0.000000
> [3,] 0 0 0.000000 0.000000
> [4,] 0 0 0.000000 0.000000
$F[[14]]
[,1] [,2] [,3] [,4]
> [1,] 0 0 14.25 16.625
> [2,] 0}0000.00 0.00
> [3,] 0}0000.00 0.00
> [4,] 0}0000.00 0.00
>
> $F[[15]]
> [,1] [,2] [,3] [,4]
> [1,] 0 0 0.5946429 1.765909
> [2,] 0 0 0.0000000 0.000000
> [3,] 0 0 0.0000000 0.000000
> [4,] 0 0 0.0000000 0.000000
>
> $F[[16]]
> [,1] [,2] [,3] [,4]
> [1,] 0 0 11.5 2.775862
> [2,] 0 0 0.0 0.000000
> [3,] 0 0 0.0 0.000000
> [4,] 0 0 0.0 0.000000
>
> $F[[17]]
> [,1] [,2] [,3] [,4]
> [1,] 0 0 3.78 1.225
> [2,] 0 0 0.00 0.000
> [3,] 0 0 0.00 0.000
> [4,] 0 0 0.00 0.000
>
> $F[[18]]
> [,1] [,2] [,3] [,4]
> [1,] 0 0 1.542857 1.035616
> [2,] 0 0 0.000000 0.000000
> [3,] 0 0 0.000000 0.000000
> [4,] 0 0 0.000000 0.000000
>
> $F[[19]]
> [,1] [,2] [,3] [,4]
> [1,] 0 0 0.15 0.175
> [2,] 0 0 0.00 0.000
> [3,] 0 0 0.00 0.000
> [4,] 0 0 0.00 0.000
>
> $F[[20]]
> [,1] [,2] [,3] [,4]
```

```
> [1,] 0}0000.2
> [2,] 0}0000.0
> [3,] 0 0 0 0.00
> [4,] 0 0 0 0.00
>
> $F[[21]]
> [,1] [,2] [,3] [,4]
> [1,] 0 0 0 1.428571
> [2,] 0
> [3,] 0 0 0 0.000000
> [4,] 0
>
> $F[[22]]
> [,1] [,2] [,3] [,4]
> [1,] 0 0}00.7 0.612
> [2,] 0 0}00.0 0.000
> [3,] 0 0}00.0 0.000
> [4,] 0 0 0.0 0.0000
>
> $F[[23]]
> [,1] [,2] [,3] [,4]
> [1,] 0 0 0 0.6
> [2,] 0
> [3,] 0
> [4,] 0}000
>
> $F[[24]]
> [,1] [,2] [,3] [,4]
> [1,] 0 0 0.7 0.6125
> [2,] 0 0}00.0 0.000
> [3,] 0 0 0.0 0.0000
> [4,] 0 0 0.0 0.0000
>
> $F[[25]]
> [,1] [,2] [,3] [,4]
> [1,] 0 0 2.1 0.8166667
> [2,] 0}0
> [3,] 0}0
> [4,] 0 0 0.0 0.0000000
>
> $F[[26]]
    [,1] [,2] [,3] [,4]
> [1,] 0 0 0 % 7
> [2,] 0
> [3,] 0
> [4,] 0 0 0 0
>
> $F[[27]]
> [,1] [,2] [,3] [,4]
> [1,] 0
> [2,] 0}00<
```

```
[3,] 0}00\quad0\quad0.
[4,] 0}000
>
>
> $hstages
[1] NA
$agestages
[1] NA
>
> $ahstages
stage_id stage_id stage original_size original_size_b original_size_c min_age
> 1 1 1 1 N Sdl llllll
>2 2 2 Veg NA NA NA
> 3 3 3 SmFlo Nalll
>4 4 4 LFlo N N NA NA
> max_age repstatus obsstatus propstatus immstatus matstatus entrystage
> 1 NA 
> 2 NA 
\begin{tabular}{llllllll}
\(>\) & 3 & NA & 1 & 1 & 0 & 0 & 1 \\
\(>\) & NA & 1 & 1 & 0 & 0 & 1 & 0
\end{tabular}
> indataset binhalfwidth_raw sizebin_min sizebin_max sizebin_center
l 1 1 0.5 0.5 0.5 1.5 l
\begin{tabular}{llllll}
\(>2\) & 1 & 0.5 & 0.5 & 1.5 & 1
\end{tabular}
\begin{tabular}{llllll}
\(>3\) & 1 & 0.5 & 1.5 & 2.5 & 2
\end{tabular}
\begin{tabular}{llllll}
\(>4\) & 1 & 0.5 & 2.5 & 3.5 & 3
\end{tabular}
> sizebin_width binhalfwidthb_raw sizebinb_min sizebinb_max sizebinb_center
> 1 1 NA NA NA
> 2 1 NA NA NA
> 3 1 NA NA NA
> 4 1 NA NA NA NA
    sizebinb_width binhalfwidthc_raw sizebinc_min sizebinc_max sizebinc_center
> 1 NA NA NA NA NA
\(>2\) NA NA NA NA
> NA NA NA NA NA
> 4 NA NA NA NA
> sizebinc_width group comments alive almostborn
> 1 NA 0 Seedling 1 0
> 2 NA 0 Vegetative adult 1 0
> 3 NA 0 Small flowering 1 0
> 4 NA 0 Large flowering 1 0
$labels
    pop patch year2
1 1 C 2003
> 2 1 C 2004
> 3 1 C 2005
>4 1 E 2003
> 5 1 E 2004
> 6 1 E 2005
>7 1 F
```

| > | 8 | 1 | F | 2004 |
| :---: | :---: | :---: | :---: | :---: |
| > | 9 | 1 | F | 2005 |
| > | 10 | 1 | G | 2003 |
| > | 11 | 1 | G | 2004 |
| > | 12 | 1 | G | 2005 |
| > | 13 | 1 | L | 2003 |
| > | 14 | 1 | L | 2004 |
| > | 15 | 1 | L | 2005 |
| > | 16 | 1 | 0 | 2003 |
| > | 17 | 1 | 0 | 2004 |
| > | 18 | 1 | 0 | 2005 |
| > | 19 | 1 | Q | 2003 |
| > | 20 | 1 | Q | 2004 |
| > | 21 | 1 | Q | 2005 |
| $>$ | 22 | 1 | R | 2003 |
| > | 23 | 1 | R | 2004 |
| > | 24 | 1 | R | 2005 |
| > | 25 | 1 | S | 2003 |
| > | 26 | 1 | S | 2004 |
| > | 27 | 1 | S | 2005 |
| > |  |  |  |  |
| > \$matrixqc |  |  |  |  |
| > | [1] | 167 | 48 | 27 |
| > |  |  |  |  |
| > \$dataqc |  |  |  |  |
| > [1] NA NA |  |  |  |  |
| > |  |  |  |  |
| > attr(,"class") |  |  |  |  |
| > [1] "lefkoMat" |  |  |  |  |

The resulting object has all of the elements of a standard lefkoMat object except for those elements related to quality control in the demographic dataset and linear modeling. The option UFdecomp was left at its default (UFdecomp = TRUE), and so create_1M () used the stageframe to infer where fecundity values were located in the matrices and created U and F matrices separating those values. The default option for historical is set to FALSE, yielding an NA in place of the hstages element, which would typically list the order of historical stage pairs.

### 11.4.1 LTRE analysis

Now, we will develop arithmetic mean matrices and assess the deterministic and stochastic population growth rates, $\lambda$ and $a$ (figure 11.2).

```
anth_lmean <- lmean(anthyllis)
lambda2 <- lambda3(anthyllis)
lambda2m <- lambda3(anth_lmean)
set.seed(42)
sl2 <- slambda3(anthyllis) #Stochastic growth rate
sl2$expa <- exp(sl2$a)
plot(lambda ~ as.integer(year2), data = lambda2,
```

```
    ylim = c(0, 2.5), xlim = c(2003, 2005), xlab = "Year",
    ylab = expression(lambda), type = "l", col = "gray", lty= 2, lwd = 2, bty = "n")
lines(lambda ~ year2, data = subset(lambda2, patch == 2), col = "gray", lty= 2, lwd = 2)
lines(lambda ~ year2, data = subset(lambda2, patch == 3), col = "gray", lty= 2, lwd = 2)
lines(lambda ~ year2, data = subset(lambda2, patch == 4), col = "gray", lty= 2, lwd = 2)
lines(lambda ~ year2, data = subset(lambda2, patch == 5), col = "gray", lty= 2, lwd = 2)
lines(lambda ~ year2, data = subset(lambda2, patch == 6), col = "gray", lty= 2, lwd = 2)
lines(lambda ~ year2, data = subset(lambda2, patch == 7), col = "gray", lty= 2, lwd = 2)
lines(lambda ~ year2, data = subset(lambda2, patch == 8), col = "gray", lty= 2, lwd = 2)
lines(lambda ~ year2, data = subset(lambda2, patch == 9), col = "gray", lty= 2, lwd = 2)
abline(a = lambda2m$lambda[1], b = 0, lty = 1, lwd = 4, col = "orangered")
abline(a = sl2$expa[1], b = 0, lty = 1, lwd = 4, col = "darkred")
legend("topleft", c("det annual", "det mean", "stochastic"), lty = c(2, 1, 1),
    col = c("gray", "orangered", "darkred"), lwd = c(2, 4, 4), bty = "n")
```



Figure 11.2: Deterministic vs. stochastic lambda
Clearly, these populations exhibit extremely variable growth across the short field study during which they were monitored. Also very clear is that these populations appear to be on the decline, as shown by $\lambda$ for the overall mean matrix, and $e$ to the power of the stochastic growth rate $a=\log \lambda$.

### 11.4.1.1 Deterministic LTREs

Let's conduct a one-way, fixed life table response experiment (LTRE) next. This will be a deterministic LTRE against the input matrices, meaning that we will assess the impacts of differences in matrix elements between the core matrices input and the overall arithmetic grand mean matrix on differences in the deterministic growth rate, $\lambda$. Since it is not appropriate to assess the differences due to time this way, we will conduct this LTRE using the patch-level mean matrices from anth_lmean against the grand mean. Let's take a look at the labels element to this object first. Note in the output below that we have 10 matrices, including 9 population matrices and one overall among-population mean.


One key problem is that the grand mean is actually included as the last matrix. We do not wish to run an LTRE in which the reference matrix is also a treatment matrix, so we will first delete the grand mean from this object, and use the resulting lefkoMat object in our LTRE. Let's delete the final matrix as below.

```
pruned_anth_lmean <- delete_lM(anth_lmean, mat_num = 10)
pruned_anth_lmean$labels
> pop patch
> 1 1 C
> 2 1 E
> 3 1 F
> 1 G
> 5 1 L
> 6 1 0
> 7 1 Q
> 8 1 R
> 9 1 S
```

Success! Now, let's run the ltre3() function. We will use the default settings, which set the reference as the grand mean matrix (which is computed during the LTRE run). We could use a different reference matrix if we wished, and this would be specified by setting the refmats option to an appropriate reference lefkoMat object.

```
trialltre_det <- ltre3(pruned_anth_lmean)
> Warning: Matrices input as mats will also be used in reference matrix
> calculation.
> Using all refmats matrices in reference matrix calculation.
trialltre_det
> $cont_mean
> $cont_mean[[1]]
> [,1] [,2] [,3] [,4]
> [1,] 0.00000000 0.0000000000 -0.0526643830 -0.278604545
> [2,] 0.01288781 -0.0002075509 -0.0011802734 -0.002247916
> [3,] 0.02099388 0.0095296944 -0.0007380232 -0.001431440
> [4,] -0.02838087 -0.0062826320 0.0199742696 -0.036219658
>
> $cont_mean[[2]]
> [,1] [,2] [,3] [,4]
> [1,] 0.000000000 0.000000000000.0010320913 0.113969244
```

```
> [2,] -0.008180535 -0.0010477967 0.0004334319 -0.005568240
> [3,] -0.016462320 0.0100496346 0.0009236077 -0.003737373
> [4,] -0.045108893 0.0008961807 0.0039524441 -0.081432730
>
> $cont_mean[[3]]
> [,1] [,2] [,3] [,4]
> [1,] 0.000000000 0.0000000000 -0.024063752 0.275500504
> [2,] -0.005037786 -0.0025268625 0.001872599 0.002568217
> [3,] 0.023776228
> [4,] -0.315031272 0.0001712855 0.004413070-0.065764758
>
> $cont_mean[[4]]
> [,1] [,2] [,3] [,4]
> [1,] 0.0000000000 0.000000000 -5.028329e-02 -0.169540748
> [2,] -0.0269687794 -0.001420158 -3.137497e-04 -0.005202208
> [3,] 0.0001492085 -0.004256203 8.430731e-05 -0.008839788
> [4,] 0.0311640114 -0.003928060 -2.974225e-03 -0.017954340
>
> $cont_mean[[5]]
> [,1] [,2] [,3] [,4]
> [1,] 0.00000000 0.00000000 0.112678253 0.300395132
> [2,] -0.01743882 -0.00131768 -0.000567669 0.002407567
> [3,] -0.01091381 -0.01241459 -0.001846088 -0.006955586
> [4,] -0.46041272 -0.03371817 -0.007443071 -0.083077750
>
> $cont_mean[[6]]
> [,1] [,2] [,3] [,4]
> [1,] 0.000000000 0.00000000 0.145611873 -0.132570128
> [2,] 0.085120191 0.01611388
> [3,] 0.002147524 0.01287140 0.005025013 0.002326089
> [4,] -0.114627718 0.03934714 -0.006712288 -0.012485180
>
> $cont_mean[[7]]
\(>\quad[, 1] \quad[, 2] \quad[, 3] \quad[, 4]\)
> [1,] 0.000000000 0.0000000000 -0.0565636772 -0.5554555750
> [2,] -0.022442169 -0.0007237743 -0.0003715165 -0.0001003139
> [3,] 0.003947167-0.0021924335 -0.0007702247 -0.0090681614
> [4,] 0.444644607 0.0019804891 -0.0038457456 0.1207791604
>
> $cont_mean[[8]]
> [,1] [,2] [,3] [,4]
> [1,] 0.000000000 0.000000000 -0.0524795118 -0.3990970676
> [2,] 0.009959864 -0.002144705 -0.0007147407 -0.0008570809
> [3,] 0.011313284 0.003107712 -0.0014907816 -0.0104568792
> [4,] 0.055374257-0.005505580 0.0063263888 0.1766385565
>
> $cont_mean[[9]]
> [,1] [,2] [,3] [,4]
> [1,] 0.000000000 0.000000000 -0.0245397016 0.034433102
> [2,] 0.01213148 0.007593275 -0.0006363292 -0.006457662
> [3,] -0.03527604 -0.006195371 -0.0008050954 0.004342704
```

```
> [4,] -0.14291846 0.014609995 -0.0051357739 0.078041234
>
>
> $ahstages
> stage_id stage_id stage original_size original_size_b original_size_c min_age
\begin{tabular}{rrrrllll}
\(>\) & 1 & 1 & 1 & Sdl & 1 & NA & NA \\
\(>\) & 2 & 2 & 2 & Veg & 1 & NA & NA \\
\(>\) & 3 & 3 & 3 & SmFlo & 2 & NA & 0 \\
\(>4\) & 4 & 4 & LFlo & 3 & NA & 0 \\
\(>\) & 4 & & & NA & 0
\end{tabular}
max_age repstatus obsstatus propstatus immstatus matstatus entrystage
l NA 
> 2 NA 
\begin{tabular}{llllllll}
\(>\) & 3 & NA & 1 & 1 & 0 & 0 & 1 \\
\(>4\) & NA & 1 & 1 & 0 & 0 & 1 & 0
\end{tabular}
> indataset binhalfwidth_raw sizebin_min sizebin_max sizebin_center
> 1 1 0.5 0.5 0.5 1.5
>2 1 0
> 3 1 0.5 0.5 1.5 2.5
\begin{tabular}{llllll}
\(>\) & 1 & 0.5 & 2.5 & 3.5 & 3
\end{tabular}
    sizebin_width binhalfwidthb_raw sizebinb_min sizebinb_max sizebinb_center
\begin{tabular}{llllll} 
& 1 & NA NA & NA
\end{tabular}
> 2 NA NA NA NA
> 3 NA NA NA
> 4 NA NA NA NA N NA N N N
> sizebinb_width binhalfwidthc_raw sizebinc_min sizebinc_max sizebinc_center
> 1 NA NA NA NA NA
> 2 NA NA NA NA NA
> NA NA NA NA N NA
> 4 NA NA NA NA
    sizebinc_width group comments alive almostborn
\begin{tabular}{lllll}
\(>\) & NA & 0 & Seedling & 1
\end{tabular}
\begin{tabular}{lllll}
\(>\) & NA & 0 Vegetative adult & 1 & 0
\end{tabular}
> 3 NA 0 Small flowering 110
\begin{tabular}{llllll}
\(>\) & NA & 0 & Large flowering & 1 & 0
\end{tabular}
>
    $agestages
> NA
    1 NA
>
$hstages
NA
1 NA
$labels
pop patch
> 1 1 C
> 2 1 E
> 3 1 F
>4 1 G
> 5 1 L
> 6 1 0
```

```
> 7 1 Q
> % 1 R
> 9 1 S
>
> attr(,"class")
> [1] "lefkoLTRE"
```

The resulting lefkoLTRE object gives the LTRE contributions for each patch-level mean matrix relative to the arithmetic grand mean matrix. These are provided within the top list, cont_mean. These differences in LTRE contributions are essentially across space, and so are of interest if we wished to analyze these patterns geographically. They do not provide us with any understanding of how time affects population dynamics, though. Let's move on to a method that will help us assess differences between the treatment group of matrices and the reference while accounting for time.

### 11.4.1.2 Stochastic LTREs (sLTRE)

To assess the contributions across space while accounting for temporal shifts, we should conduct either a stochastic LTRE (sLTRE) or a small noise approximation LTRE (SNA-LTRE) on the original lefkoMat object holding the annual matrices. Let's conduct a stochastic LTRE first, as below.

```
trialltre_sto <- ltre3(anthyllis, stochastic = TRUE, steps = 10000)
> Warning: Matrices input as mats will also be used in reference matrix
> calculation.
> Using all refmats matrices in reference matrix calculation.
trialltre_sto
> $cont_mean
> $cont_mean[[1]]
> [,1] [,2] [,3] [,4]
> [1,] 0.00000000 0.0000000000 -0.0515416500 -0.370937927
> [2,] 0.01070788 -0.0001500566 0.0027370245 -0.002244957
> [3,] 0.01794720 0.0050753795 -0.0006037437-0.001323003
> [4,] -0.03534884-0.0058807825 0.0082570892 -0.041726820
>
> $cont_mean[[2]]
\(>\quad[, 1] \quad[, 2] \quad[, 3] \quad[, 4]\)
> [1,] 0.000000000 0.0000000000 0.0010311785 0.105734697
> [2,] -0.006706474 -0.0011291073 0.0002973553 0.015093977
> [3,] -0.016296606 0.0062287141 0.0007220703-0.004735088
> [4,] -0.040182417 0.0008234334 0.0029362458-0.119265460
>
> $cont_mean[[3]]
> [,1] [,2] [,3] [,4]
> [1,] 0.000000000 0.0000000000 -0.0205998976 0.274410792
> [2,] -0.003575952 -0.0039255244 0.0006365332 0.002196989
> [3,] 0.019246411 0.0049124671 0.0021713301 0.015124290
> [4,] -0.322969218 0.0001159174 0.0019163175 -0.090300975
>
> $cont_mean[[4]]
> [,1] [,2] [,3] [,4]
> [1,] 0.0000000000 0.000000000 -0.0927991820 -0.17724231
> [2,] -0.0527720512 0.006444722 -0.0004792117 0.01509398
> [3,] 0.0001969125 0.020276759 0.0001129494 0.03298972
```

```
> [4,] 0.0297566408 -0.005922822 -0.0041380947-0.01671769
>
> $cont_mean[[5]]
> [,1] [,2] [,3] [,4]
```



```
[2,] -0.018234327 -0.002134846 -0.001196445 0.002908661
[3,] -0.005442887-0.019611843-0.001674146 -0.010152429
> [4,] -0.542378808 -0.036550729 -0.010466781 -0.145342677
>
> $cont_mean[[6]]
> [,1] [,2] [,3] [,4]
> [1,] 0.000000000 0.000000000 0.070578256 -0.184629877
> [2,] 0.036912112 0.002816890 0.001152807 0.005769269
> [3,] 0.001139878 0.002900943 0.001527744 0.001264277
> [4,] -0.174040744 0.018089665 0.019546035 -0.016579987
>
> $cont_mean[[7]]
> [,1] [,2] [,3] [,4]
> [1,] 0.000000000 0.000000000 -0.234693467 -5.475797e-01
> [2,] 0.081113729 0.006444722 0.002737024 -9.570216e-05
> [3,] 0.007907095 0.020276759 0.006800878 3.298972e-02
> [4,] 0.364304221 0.004721979 0.019546035 7.426682e-02
>
> $cont_mean[[8]]
> [,1] [,2] [,3] [,4]
> [1,] 0.00000000 0.000000000 -0.090228851 -0.5532201480
> [2,] 0.01113030 0.006444722 0.002737024 -0.0007619147
> [3,] 0.01492979 0.003049720 0.006800878 0.0329897193
> [4,] 0.06332534 -0.005922822 0.004662797 0.0996015753
>
> $cont_mean[[9]]
\(>\quad[, 1] \quad[, 2] \quad[, 3] \quad[, 4]\)
> [1,] 0.000000000 0.000000000 -0.064004117 0.034792319
> [2,] 0.008327628 0.002563569 0.002737024 0.015093977
> [3,] 0.138593529 0.020276759 0.006800878 0.005104104
> [4,] -0.148837915 0.009080183 0.019546035 0.054241194
>
>
> $cont_sd
> $cont_sd[[1]]
> [,1] [,2] [,3] [,4]
> [1,] 0.000000000 0.0000000000 -5.598272e-04 0.0098581857
> [2,] -0.003566677 -0.0004696143 -2.161410e-05 -0.0001399796
> [3,] -0.002022219 -0.0006931012 -3.992209e-06 0.0007501200
> [4,] -0.001948788 -0.0006428649 -8.445059e-04 -0.0125644917
>
> $cont_sd[[2]]
> [,1] [,2] [,3] [,4]
> [1,] 0.000000000 0.0000000000 -2.200022e-03 -0.0284951461
> [2,] -0.001046855 -0.0003129411 -9.170339e-06 -0.0010251792
> [3,] -0.002203994 -0.0010689387-6.195287e-06 0.0005818493
```

```
> [4,] -0.004396696 -0.0018280211 -4.356751e-04 0.0037252154
>
$cont_sd[[3]]
[,1] [,2] [,3] [,4]
[1,] 0.000000000 0.0000000000 1.663148e-03 -0.0213876082
[2,] -0.001635444 0.0001345571 -6.488022e-06 -0.0003673547
[3,] -0.001958015 -0.0007395638 -6.673516e-06 0.0007659381
[4,] 0.004491542 -0.0013473554 -4.761216e-04 0.0018506667
$cont_sd[[4]]
[,1] [,2] [,3] [,4]
[1,] 0.0000000000 0.000000000 4.213639e-04 0.025869390
[2,] -0.0007407232 -0.001334540 -4.381703e-06 -0.001025179
[3,] -0.0021974119 -0.001615813 -5.183123e-06 0.002163339
[4,] -0.0057118493 -0.001942571 -3.239115e-04 -0.009859634
$cont_sd[[5]]
    [,1] [,2] [,3] [,4]
[1,] 0.000000000 0.0000000000 -8.018983e-03 -0.0548652527
[2,] -0.001216515 -0.0001519973 4.105651e-08 -0.0005365545
[3,] -0.002343985 0.0003817458 -2.213544e-06 0.0003146872
[4,] 0.002739139 -0.0006809099 -2.793319e-05 -0.0008102534
$cont_sd[[6]]
[,1] [,2] [,3] [,4]
[1,] 0.0000000000 0.0000000000 -6.804802e-03 -0.0002060203
[2,] -0.0047451981 -0.0005049195 -1.350415e-05 -0.0004345241
[3,] -0.0004719449 -0.0004578419 -7.534059e-06 0.0005527141
[4,] 0.0055256824 -0.0007386864 -1.257551e-03 0.0020952799
$cont_sd[[7]]
    [,1] [,2] [,3] [,4]
    [1,] 0.000000000 0.000000000 6.646112e-03 0.0074216476
    [2,] -0.009051256-0.001334540 -2.161410e-05 -0.0004021212
    [3,] -0.003951052 -0.001615813 -1.455042e-05 0.0021633392
    [4,] -0.008447405 -0.002320584 -1.257551e-03 -0.0180788408
    $cont_sd[[8]]
    [,1] [,2] [,3] [,4]
    [1,] 0.000000000 0.0000000000 1.592000e-03 0.1215757147
    [2,] -0.003423259 -0.0013345396 -2.161410e-05 -0.0003687783
    [3,] -0.004242417 -0.0008904705 -1.455042e-05 0.0021633392
    [4,] 0.002932141 -0.0012207845 -7.355093e-04 -0.0145520935
$cont_sd[[9]]
    [,1] [,2] [,3] [,4]
    [1,] 0.000000000 0.0000000000 -2.012483e-03 -0.031970737
    [2,] -0.003629698-0.0009038638-2.161410e-05 -0.001025179
    [3,] -0.008174522 -0.0016158128 -1.455042e-05 0.001067080
    [4,] -0.003888201 -0.0024753508 -1.257551e-03 -0.017334303
>
```

```
>
> $ahstages
> stage_id stage_id stage original_size original_size_b original_size_c min_age
> 1 1 1 N Sdl NA NA
> 2 2 2 Veg NA NA NA
> 3 <rrroll
> max_age repstatus obsstatus propstatus immstatus matstatus entrystage
> 1 NA 
>2 NA 
> 3 N NA 
> indataset binhalfwidth_raw sizebin_min sizebin_max sizebin_center
> 1 1 0.0.5 0.5 1.5
> 2 1 0.5 0.5 0.5 1.5
> 3 1 0.5 0.5 1.5 2.5
\begin{tabular}{llllll}
\(>\) & 1 & 0.5 & 2.5 & 3.5 & 3
\end{tabular}
        sizebin_width binhalfwidthb_raw sizebinb_min sizebinb_max sizebinb_center
> 1 1 NA NA NA
> 2 1 NA NA NA NA
> 3 NA NA NA NA
> 4 N \(1 \quad\) NA NA NA
> sizebinb_width binhalfwidthc_raw sizebinc_min sizebinc_max sizebinc_center
> NA NA NA NA
> 2 NA NA NA NA NA
> NA NA NA NA
> 4 NA NA NA NA
sizebinc_width group comments alive almostborn
1 \begin{tabular}{lllll}
1 & NA & 0 & Seedling & 1
\end{tabular}
2 NA 0 Vegetative adult 1 0
3 NA 0 Small flowering 1 0
> 4 NA 0 Large flowering 1 0
$agestages
[1] NA
>
$hstages
[1] NA
$labels
    pop patch
1 1 C
> 4 E
> 7 1 F
> 10 1 G
>13 1 L
> 16 1 0
> 19 19 Q
> 22 1 R
> 25 1 S
>
```

```
> attr(,"class")
```

> [1] "lefkoLTRE"

The sLTRE produces output that is a bit different from the deterministic LTRE output. In the output, we see two lists of matrices prior to the MPM metadata. The first, cont_mean, is a list of matrices showing the impact of differences in mean elements between the patch-level temporal mean matrices and the reference temporal mean matrix. The second, cont_sd, is a list of matrices showing the impact of differences in the temporal standard deviation of each element between the patch-level and reference matrix sets. In other words, while a standard LTRE shows the impact of changes in matrix elements on $\lambda$, the sLTRE shows the impacts of changes in the temporal mean and variability of matrix elements on $\log \lambda$. The labels element shows the order of matrices with reference to the populations or patches (remember that here, the populations are referred to as patches), and the order is the same as in pruned_anth_lmean.

### 11.4.1.3 Small Noise Approximation LTREs (SNA-LTRE)

The stochastic LTRE is a very useful analytical approach, but it assumes that matrix elements and hence vital rates are not correlated. In truth, constraints may be operating on organisms that do not allow elements to vary independently. The small noise approximation LTRE is a form of the stochastic LTRE that allows these correlations to be assessed.

Let's conduct an SNA-LTRE analysis, as below. We will suppress the final line to prevent the output from taking too many pages of this book.

```
> Warning: Matrices input as mats will also be used in reference matrix
> calculation.
> Using all refmats matrices in reference matrix calculation.
> $cont_mean
> $cont_mean[[1]]
> [,1] [,2] [,3] [,4]
> [1,] 0.00000000 0.0000000000 -0.0476202656 -0.344040835
> [2,] 0.01032171 -0.0001486831 0.0000000000-0.002131189
> [3,] 0.01698357 0.0049341375 -0.0005569668-0.001225294
> [4,] -0.03436670 -0.0058801269 0.0078552035 -0.039845843
>
> $cont_mean[[2]]
> [,1] [,2] [,3] [,4]
> [1,] 0.000000000 0.0000000000 0.0009527245 0.098067765
> [2,] -0.006464612 -0.0011187723 0.0002812110 0.000000000
> [3,] -0.015421599 0.0060553761 0.0006661257 -0.004385384
> [4,] -0.039065983 0.0008233416 0.0027933340-0.113889168
>
> $cont_mean[[3]]
\(>\) [,1] \(\quad[, 2] \quad[, 3] \quad[, 4]\)
> [1,] 0.000000000 0.0000000000 -0.0190326192 0.254512983
> [2,] -0.003446989 -0.0038895931 0.0006019741 0.002085652
> [3,] 0.018213022 0.0047757587
> [4,] -0.313995798 0.0001159045 0.0018230473 -0.086230354
>
> $cont_mean[[4]]
> [,1] [,2] [,3] [,4]
> [1,] 0.0000000000 0.000000000 -0.0857388480 -0.16439029
> [2,] -0.0508688867 0.000000000 -0.0004531940 0.00000000
```

```
> [3,] 0.0001863397 0.000000000 0.0001041983 0.00000000
> [4,] 0.0289298783-0.005922162 -0.0039366871 -0.01596409
>
> $cont_mean[[5]]
> [,1] [,2] [,3] [,4]
> [1,] 0.000000000 0.000000000 0.064519548 0.297311615
> [2,] -0.017576727 -0.002115305 -0.001131487 0.002761258
> [3,] -0.005150644 -0.019066068 -0.001544436 -0.009402634
> [4,] -0.527309283-0.036546654 -0.009957346 -0.138790866
>
> $cont_mean[[6]]
> [,1] [,2] [,3] [,4]
> [1,] 0.000000000 0.000000000 0.065208531 -0.171242174
> [2,] 0.035580919 0.002791106 0.001090218 0.005476899
> [3,] 0.001078675 0.002820213 0.001409378 0.001170906
> [4,] -0.169205172 0.018087649 0.000000000 -0.015832588
>
> $cont_mean[[7]]
> [,1] [,2] [,3] [,4]
> [1,] 0.000000000 0.000000000 -0.2168376 -5.078741e-01
> [2,] 0.000000000 0.000000000 0.0000000 -9.085225e-05
> [3,] 0.007482543 0.000000000 0.0000000 0.000000e+00
> [4,] 0.354182344 0.004721453 0.0000000 7.091899e-02
>
> $cont_mean[[8]]
\(>\quad[, 1] \quad[, 2] \quad[, 3] \quad[, 4]\)
> [1,] 0.00000000 0.000000000 -0.083364073 -0.5131055846
> [2,] 0.01072890 0.000000000 0.000000000 -0.0007233031
> [3,] 0.01412817 0.002964850 0.000000000 0.0000000000
> [4,] 0.06156590 -0.005922162 0.004435852 0.0951116984
>
> $cont_mean[[9]]
> [,1] [,2] [,3] [,4]
> [1,] 0.000000000 0.000000000 -0.05913457 0.032269492
> [2,] 0.008027302 0.002540104 0.00000000 0.000000000
> [3,] 0.000000000 0.000000000 0.00000000 0.004727146
> [4,] -0.144702583 0.009079171 0.00000000 0.051796089
>
>
> $cont_elas
> $cont_elas[[1]]
> [,1] [,2] [,3] [,4] [,5] [,6]
> [1,] 0 0.000000e+00 0.000000e+00 0.000000e+00 0
> [2,] 0 1.076414e-03 2.352419e-04 1.434057e-06 0 1.151661e-04
> [3,] 0 2.352419e-04 4.283633e-04 1.051179e-04 0
> [4,] 0 1.434057e-06 1.051179e-04 -3.849196e-03 0 3.877057e-05
> [5,] 0 0.000000e+00 0.000000e+00 0.000000e+00 0 0.000000e+00
> [6,] 0 1.151661e-04 8.701039e-05 3.877057e-05 0 2.137169e-05
> [7,] 0 4.127369e-04 2.681131e-04 1.525169e-04 0 6.580792e-05
> [8,] 0 1.399565e-04 2.066573e-04 -1.163171e-04 0 4.847590e-05
> [9,] 0 3.374024e-04 2.420751e-05 0.000000e+00 0 2.240561e-05
> [10,] 0 -2.361078e-06 0.000000e+00 0.0000000e+00 0 -1.954024e-08
```

```
> [11,] 0 0.000000e+00 3.752392e-06 3.655536e-05 0 4.602877e-08
> [12,] 0 7.208568e-04 5.796014e-04 3.177181e-04 0 1.325681e-04
> [13,] 0 4.186480e-04 1.122424e-04 -3.171635e-04 0 6.371443e-05
> [14,] 0 2.022332e-05 3.300908e-06 -9.458058e-10 0 1.822699e-06
> [15,] 0 0.000000e+00 0.000000e+00 7.485821e-05 0 0.000000e+00
> [16,] 0 9.027842e-04 8.404479e-04 -1.267457e-03 0 2.673230e-04
> [,7] [,8] [,9] [,10] [,11]
> [1,] 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00
> [2,] 4.127369e-04 1.399565e-04 3.374024e-04 -2.361078e-06 0.000000e+00
> [3,] 2.681131e-04 2.066573e-04 2.420751e-05 0.000000e+00 3.752392e-06
> [4,] 1.525169e-04 -1.163171e-04 0.000000e+00 0.000000e+00 3.655536e-05
> [5,] 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00
> [6,] 6.580792e-05 4.847590e-05 2.240561e-05 -1.954024e-08 4.602877e-08
> [7,] 2.057798e-04 1.536078e-04 9.981754e-05 -1.878964e-07 0.000000e+00
> [8,] 1.536078e-04 9.186061e-05 0.000000e+00 0.000000e+00 4.505735e-06
> [9,] 9.981754e-05 0.0000000+00 1.126477e-03 -3.011785e-06 0.000000e+00
> [10,] -1.878964e-07 0.000000e+00 -3.011785e-06 -5.586616e-08 0.000000e+00
> [11,] 0.000000e+00 4.505735e-06 0.000000e+00 0.000000e+00 8.433637e-06
> [12,] 3.874225e-04 3.957651e-04 0.000000e+00 0.000000e+00 6.023972e-06
> [13,] 2.274047e-04 -2.786298e-06 6.848409e-04 -4.823061e-06 0.000000e+00
> [14,] 6.973409e-06 0.000000e+00 7.646942e-05 -1.681941e-07 9.742139e-07
> [15,] 0.000000e+00 5.640677e-06 0.000000e+00 0.000000e+00 3.414841e-05
> [16,] 9.684039e-04 2.290654e-04 0.000000e+00 0.000000e+00 1.352264e-05
> [,12] [,13] [,14] [,15] [,16]
> [1,] 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00 0.0000000e+00
> [2,] 7.208568e-04 4.186480e-04 2.022332e-05 0.000000e+00 1.027842e-04
> [3,] 5.796014e-04 1.122424e-04 3.300908e-06 0.000000e+00 8.404479e-04
> [4,] 3.177181e-04 -3.171635e-04 -9.458058e-10 7.485821e-05 -1.267457e-03
> [5,] 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00
> [6,] 1.325681e-04 6.371443e-05 1.822699e-06 0.000000e+00 2.673230e-04
> [7,] 3.874225e-04 2.274047e-04 6.973409e-06 0.000000e+00 9.684039e-04
> [8,] 3.957651e-04 -2.786298e-06 0.000000e+00 5.640677e-06 2.290654e-04
> [9,] 0.000000e+00 6.848409e-04 7.646942e-05 0.000000e+00 0.000000e+00
> [10,] 0.000000e+00 -4.823061e-06 -1.681941e-07 0.000000e+00 0.000000e+00
> [11,] 6.023972e-06 0.000000e+00 9.742139e-07 3.414841e-05 1.352264e-05
> [12,] 8.936679e-04 1.676949e-04 0.000000e+00 6.217921e-06 2.404203e-03
> [13,] 1.676949e-04 -5.576203e-03 4.628817e-05 0.000000e+00 -3.932006e-04
> [14,] 0.000000e+00 4.628817e-05 6.698721e-06 3.923765e-06 2.271869e-06
> [15,] 6.217921e-06 0.000000e+00 3.923765e-06 1.409858e-04 2.979157e-06
> [16,] 2.404203e-03 -3.932006e-04 2.271869e-06 2.979157e-06 -8.440603e-04
>
> $cont_elas[[2]]
> [,1] [,2] [,3] [,4] [,5] [,6]
> [1,] 0 0.000000e+00 0.000000e+00 0.000000e+00 0
> [2,] 0 2.319310e-04 8.939067e-05 3.000317e-06 0 -6.100821e-07
> [3,] 0 8.939067e-05 -1.117270e-04 3.714621e-06 0 -2.106912e-06
> [4,] 0 3.000317e-06 3.714621e-06 3.879036e-03 0 -3.107849e-06
> [5,] 0 0.000000e+00 0.000000e+00 0.000000e+00 0 0.000000e+00
> [6,] 0 -6.100821e-07 -2.106912e-06 -3.107849e-06 0 -2.237280e-06
> [7,] 0 1.837680e-04 2.145766e-04 1.258783e-05 0 7.554565e-07
> [8,] 0 8.540884e-05 -5.559268e-05 9.046159e-06 0-1.238956e-06
> [9,] 0 8.158883e-05 -7.793246e-06 1.245674e-04 0-2.810857e-06
```




```
> [8,] 0 0.000000e+00 -1.937221e-04 -3.156115e-03 0 -4.080795e-06
> [9,] 0 -1.347545e-03 -1.505000e-05 0.000000e+00 0 -8.905397e-06
> [10,] 0 -2.049088e-06 -7.686431e-06 0.000000e+00 0 -1.954024e-08
> [11,] 0 0.000000e+00 -2.029324e-05 -2.938429e-06 0 -1.172880e-08
> [12,] 0 0.000000e+00 -7.103353e-05 -7.155646e-06 0 -8.638454e-07
> [13,] 0 -5.275206e-04 -4.149709e-04 2.464273e-04 0 -6.835728e-05
> [14,] 0 -1.255931e-05 -2.393388e-06 -1.177230e-06 0 -7.255730e-07
> [15,] 0 0.000000e+00 0.000000e+00 -5.307437e-06 0 0.000000e+00
> [16,] 0 -3.586527e-05 -5.524694e-04 4.796185e-03 0 -3.332669e-05
> [,7] [,8] [,9] [,10] [,11]
> [1,] 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00
> [2,] -1.848965e-05 0.000000e+00 -1.347545e-03 -2.049088e-06 0.000000e+00
> [3,] -6.055219e-06 -1.937221e-04 -1.505000e-05 -7.686431e-06 -2.029324e-05
> [4,] -9.006694e-06 -3.156115e-03 0.000000e+00 0.000000e+00 -2.938429e-06
> [5,] 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00
> [6,] -1.356001e-06 -4.080795e-06 -8.905397e-06 -1.954024e-08 -1.172880e-08
> [7,] -2.201744e-06 -1.275882e-06 -1.968252e-05 -1.878964e-07 0.000000e+00
> [8,] -1.275882e-06 -9.860547e-04 0.000000e+00 0.000000e+00 -2.581603e-06
> [9,] -1.968252e-05 0.000000e+00 -1.537051e-03 -2.539552e-06 0.000000e +00
> [10,] -1.878964e-07 0.000000e+00 -2.539552e-06 -3.356551e-07 -7.302652e-07
> [11,] 0.000000e+00 -2.581603e-06 0.000000e+00 -7.302652e-07 -2.020732e-06
> [12,] -4.421662e-07 -7.208705e-06 0.000000e+00 -2.476487e-06 -6.541101e-06
> [13,] -9.733425e-05 -1.187843e-04 -4.892843e-04 -1.090214e-05 -2.329926e-05
> [14,] -1.376869e-06 0.000000e+00 -1.400386e-05 -1.681941e-07 0.000000e+00
> [15,] 0.000000e+00 -5.218768e-06 0.000000e+00 0.000000e+00 -9.657494e-07
> [16,] -2.436425e-05 -1.100242e-03 0.000000e+00 -2.359928e-06 -1.150927e-05
> [,12] [,13] [,14] [,15] [,16]
> [1,] 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00
> [2,] 0.000000e+00 -5.275206e-04 -1.255931e-05 0.000000e+00 -3.586527e-05
> [3,] -7.103353e-05 -4.149709e-04 -2.393388e-06 0.000000e+00 -5.524694e-04
> [4,] -7.155646e-06 2.464273e-04 -1.177230e-06 -5.307437e-06 4.796185e-03
> [5,] 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00
> [6,] -8.638454e-07 -6.835728e-05 -7.255730e-07 0.000000e+00 -3.332669e-05
> [7,] -4.421662e-07 -9.733425e-05 -1.376869e-06 0.000000e+00 -2.436425e-05
> [8,] -7.208705e-06 -1.187843e-04 0.000000e+00 -5.218768e-06 -1.100242e-03
> [9,] 0.000000e+00 -4.892843e-04 -1.400386e-05 0.000000e+00 0.000000e+00
> [10,] -2.476487e-06 -1.090214e-05 -1.681941e-07 0.000000e+00 -2.359928e-06
> [11,] -6.541101e-06 -2.329926e-05 0.000000e+00 -9.657494e-07 -1.150927e-05
> [12,] -2.167189e-05 -7.680287e-05 0.000000e+00 -6.365250e-07 -2.525980e-05
> [13,] -7.680287e-05 1.601094e-03 -5.643506e-05 0.000000e+00 1.063591e-03
> [14,] 0.000000e+00 -5.643506e-05 -9.255458e-07 0.000000e+00 -5.579944e-06
> [15,] -6.365250e-07 0.000000e+00 0.000000e+00 -5.426486e-06 -4.678731e-06
> [16,] -2.525980e-05 1.063591e-03 -5.579944e-06 -4.678731e-06 4.666480e-03
>
> $cont_elas[[5]]
> [,1] [,2] [,3] [,4] [,5] [,6]
> [1,] 0 0.000000e+00 0.000000e+00 0.000000e+00 0 0.000000e+00
> [2,] 0 -5.947035e-05 8.770015e-04 -4.707494e-04 0 -1.604684e-05
> [3,] 0 8.770015e-04 1.381029e-02 7.904315e-03 0 4.336996e-05
> [4,] 0 -4.707494e-04 7.904315e-03 -1.434901e-02 0 -2.241783e-04
> [5,] 0 0.000000e+00 0.000000e+00 0.000000e+00 0 0.000000e+00
> [6,] 0 -1.604684e-05 4.336996e-05 -2.241783e-04 0 -3.781875e-06
```

```
> [7,] 0 -1.337792e-05 1.830607e-06 -7.269903e-06 0 -1.193394e-06
> [8,] 0 -1.452435e-05 2.642577e-03 -1.585136e-03 0 -3.342284e-05
> [9,] 0 1.445643e-03 1.671538e-02 8.799133e-03 0 4.131947e-05
[10,] 0 -1.778462e-06 0.000000e+00 0.000000e+00 0 -1.744882e-08
> [11,] 0 0.000000e+00 1.072560e-05 9.245505e-07 0 -3.437915e-09
> [12,] 0 0.000000e+00 1.520093e-05 -1.327403e-05 0 -6.321923e-07
> [13,] 0 -9.163471e-04 1.249912e-02 -1.854314e-02 0 -3.334515e-04
> [14,] 0 -2.488340e-05 1.349364e-04 -3.966655e-04 0 -6.214296e-06
> [15,] 0 0.000000e+00 0.000000e+00 -4.058416e-06 0 0.000000e+00
> [16,] 0 -3.618972e-04 3.157331e-04 -1.005624e-02 0 -1.487662e-04
> [,7] [,8] [,9] [,10] [,11]
> [1,] 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00
> [2,] -1.337792e-05 -1.452435e-05 1.445643e-03 -1.778462e-06 0.000000e+00
> [3,] 1.830607e-06 2.642577e-03 1.671538e-02 0.000000e+00 1.072560e-05
> [4,] -7.269903e-06 -1.585136e-03 8.799133e-03 0.000000e+00 9.245505e-07
> [5,] 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00
> [6,] -1.193394e-06 -3.342284e-05 4.131947e-05 -1.744882e-08 -3.437915e-09
> [7,] -4.175447e-05 -8.984707e-07 1.787175e-06 -3.997998e-06 -4.864674e-06
[8,] -8.984707e-07 -3.142190e-06 3.018943e-03 0.000000e+00 2.221567e-06
[9,] 1.787175e-06 3.018943e-03 2.362048e-02 -7.953067e-08 0.000000e+00
[10,] -3.997998e-06 0.000000e+00 -7.953067e-08 -4.195874e-07 -5.167908e-07
[11,] -4.864674e-06 2.221567e-06 0.000000e+00 -5.167908e-07 5.349229e-06
[12,] -2.995773e-05 -2.927548e-06 0.000000e+00 -2.893565e-06 1.425118e-06
[13,] -7.716843e-05 -1.856017e-03 1.483318e-02 -3.931183e-06 0.000000e+00
> [14,] -1.164382e-06 -5.148269e-05 1.565294e-04 -1.450264e-07 0.000000e+00
> [15,] -4.867468e-05 -3.334794e-06 0.000000e+00 -4.695883e-06 -5.001973e-06
> [16,] -2.248042e-05 -1.707169e-03 -1.137664e-04 0.000000e+00 -7.563332e-06
> [,12] [,13] [,14] [,15] [,16]
> [1,] 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00 0.0000000e+00
> [2,] 0.000000e+00 -9.163471e-04 -2.488340e-05 0.000000e+00 -3.618972e-04
> [3,] 1.520093e-05 1.249912e-02 1.349364e-04 0.000000e+00 3.157331e-04
> [4,] -1.327403e-05 -1.854314e-02 -3.966655e-04 -4.058416e-06 -1.005624e-02
> [5,] 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00
> [6,] -6.321923e-07 -3.334515e-04 -6.214296e-06 0.000000e+00 -1.487662e-04
> [7,] -2.995773e-05 -7.716843e-05 -1.164382e-06 -4.867468e-05 -2.248042e-05
> [8,] -2.927548e-06 -1.856017e-03 -5.148269e-05 -3.334794e-06 -1.707169e-03
> [9,] 0.000000e+00 1.483318e-02 1.565294e-04 0.000000e+00 -1.137664e-04
> [10,] -2.893565e-06 -3.931183e-06 -1.450264e-07 -4.695883e-06 0.000000e+00
> [11,] 1.425118e-06 0.000000e+00 0.000000e+00 -5.001973e-06 -7.563332e-06
> [12,] -1.835906e-05 -2.198856e-05 0.000000e+00 -3.552700e-05 -4.264209e-05
> [13,] -2.198856e-05 -2.563633e-02 -5.578350e-04 0.000000e+00 -1.389359e-02
> [14,] 0.000000e+00 -5.578350e-04 -1.146805e-05 0.000000e+00 -2.407963e-04
> [15,] -3.552700e-05 0.000000e+00 0.000000e+00 -6.400654e-05 -4.237241e-06
> [16,] -4.264209e-05 -1.389359e-02 -2.407963e-04 -4.237241e-06 -6.553095e-03
>
> $cont_elas[[6]]
> [,1] [,2] [,3] [,4] [,5] [,6]
> [1,] 0 0.000000e+00 0.000000e+00 0.000000e+00 0 0.000000e+00
> [2,] 0 4.036577e-03 2.026206e-04 3.479199e-05 0 8.144725e-04
> [3,] 0 2.026206e-04 6.139678e-04 -3.393062e-05 0 2.011861e-04
> [4,] 0 3.479199e-05 -3.393062e-05 -9.957928e-04 0 9.793096e-05
> [5,] 0 0.000000e+00 0.000000e+00 0.000000e+00 0 0.000000e+00
```




```
> [4,] 0 -4.326007e-06 -9.361114e-05 -5.649676e-04 0 -1.399992e-05
> [5,] 0 0.000000e+00 0.000000e+00 0.000000e+00 0 0.000000e+00
> [6,] 0 -9.496188e-06 -6.372763e-06 -1.399992e-05 0 -1.067376e-06
> [7,] 0 -4.177398e-05 -1.981439e-04 -4.889314e-06 0 -1.356001e-06
> [8,] 0 -1.557975e-04 -5.215217e-04 -7.351234e-05 0 -4.080795e-06
[9,] 0 -1.579596e-04 -1.561445e-05 -3.983631e-05 0 -8.905397e-06
[10,] 0 -2.361078e-06 0.000000e+00 0.000000e+00 0 -1.954024e-08
> [11,] 0 0.000000e+00 -1.606376e-06 -6.762812e-06 0 -1.172880e-08
> [12,] 0 0.000000e+00 -3.307237e-06 -2.175326e-05 0 -8.638454e-07
> [13,] 0 -4.593132e-04 -2.761692e-04 -4.636473e-04 0-6.835728e-05
> [14,] 0 2.430857e-06 -8.177874e-07 -6.336888e-08 0 -7.255730e-07
> [15,] 0 0.000000e+00 0.000000e+00 -5.307437e-06 0 0.000000e+00
> [16,] 0 1.870199e-04 8.605532e-04 1.295786e-03 0 -3.332669e-05
> [,7] [,8] [,9] [,10] [,11]
> [1,] 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00
> [2,] -4.177398e-05 -1.557975e-04 -1.579596e-04 -2.361078e-06 0.000000e+00
> [3,] -1.981439e-04 -5.215217e-04 -1.561445e-05 0.000000e+00 -1.606376e-06
> [4,] -4.889314e-06 -7.351234e-05 -3.983631e-05 0.000000e+00 -6.762812e-06
> [5,] 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00 0.000000ee+00
> [6,] -1.356001e-06 -4.080795e-06 -8.905397e-06 -1.954024e-08 -1.172880e-08
> [7,] -1.635631e-05 -4.494871e-05 -1.535946e-05 -1.878964e-07 0.000000e+00
> [8,] -4.494871e-05 -1.856983e-04 0.000000e+00 0.000000e+00 -2.987178e-06
> [9,] -1.535946e-05 0.000000e+00 -8.232419e-04 -3.011785e-06 0.000000e+00
> [10,] -1.878964e-07 0.000000e+00 -3.011785e-06 -5.586616e-08 0.000000e+00
> [11,] 0.000000e+00 -2.987178e-06 0.000000e+00 0.000000e+00 -2.716427e-07
> [12,] 7.954278e-08 5.610112e-07 -4.308722e-05 0.000000e+00 -5.027954e-07
> [13,] -5.011320e-05 -9.879327e-05 -6.717918e-04 -4.823061e-06 0.000000e+00
> [14,] 4.546068e-07 2.635071e-14 -3.658151e-05 -1.681941e-07 0.000000e+00
> [15,] 0.000000e+00 -5.218768e-06 0.000000e+00 0.000000e+00 -9.657494e-07
> [16,] 5.137715e-04 1.106085e-03 0.000000e+00 0.000000e+00 -1.389764e-05
> [,12] [,13] [,14] [,15] [,16]
> [1,] 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00
> [2,] 0.000000e+00 -4.593132e-04 2.430857e-06 0.000000e+00 1.870199e-04
> [3,] -3.307237e-06 -2.761692e-04 -8.177874e-07 0.000000e+00 8.605532e-04
> [4,] -2.175326e-05 -4.636473e-04 -6.336888e-08 -5.307437e-06 1.295786e-03
> [5,] 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00
> [6,] -8.638454e-07 -6.835728e-05 -7.255730e-07 0.000000e+00 -3.332669e-05
> [7,] 7.954278e-08 -5.011320e-05 4.546068e-07 0.000000e+00 5.137715e-04
> [8,] 5.610112e-07 -9.879327e-05 2.635071e-14 -5.218768e-06 1.106085e-03
> [9,] -4.308722e-05 -6.717918e-04 -3.658151e-05 0.000000e+00 0.000000e+00
> [10,] 0.000000e+00 -4.823061e-06 -1.681941e-07 0.000000e+00 0.000000e+00
> [11,] -5.027954e-07 0.000000e+00 0.000000e+00 -9.657494e-07 -1.389764e-05
> [12,] 3.817894e-05 -5.088591e-06 0.000000e+00 -6.365250e-07 2.306589e-04
> [13,] -5.088591e-06 -2.922705e-03 2.491892e-07 0.000000e+00 2.177080e-03
> [14,] 0.000000e+00 2.491892e-07 3.111808e-05 0.000000e+00 2.885444e-05
> [15,] -6.365250e-07 0.000000e+00 0.000000e+00 -5.426486e-06 -4.678731e-06
> [16,] 2.306589e-04 2.177080e-03 2.885444e-05 -4.678731e-06 4.367528e-02
>
> $cont_elas[[9]]
> [,1] [,2] [,3] [,4] [,5] [,6]
> [1,] 0 0.000000e+00 0.000000e+00 0.000000e+00 0 0.000000e+00
> [2,] 0 1.698007e-03 -2.932472e-05 1.380871e-03 0
```

```
[3,] 0 -2.932472e-05 -4.674190e-05 -1.209830e-04 0 -6.372763e-06
> [4,] 0 1.380871e-03 -1.209830e-04 -1.138416e-02 0 1.486231e-03
> [5,] 0 0.000000e+00 0.000000e+00 0.000000e+00 0 0.000000e+00
> [6,] 0 5.384622e-04 -6.372763e-06 1.486231e-03 0 2.175704e-04
> [7,] 0 -1.848965e-05 -6.055219e-06 -9.006694e-06 0 -1.356001e-06
[8,] 0 0.000000e+00 -4.101344e-05 8.804655e-05 0
[9,] 0 -1.613425e-04 -1.751877e-05 0.000000e+00 0 -4.033470e-06
[10,] 0 -2.361078e-06 0.000000e+00 0.000000e+00 0-1.954024e-08
[11,] 0 0.000000e+00 -1.606376e-06 -6.762812e-06 0-1.172880e-08
[12,] 0 0.000000e+00 -7.948917e-06 -2.333796e-05 0-8.638454e-07
[13,] 0 3.434155e-03 -3.634290e-04 -7.897535e-03 0 2.346580e-03
[14,] 0 -1.255931e-05 -2.393388e-06 -1.177230e-06 0-7.255730e-07
[15,] 0 0.000000e+00 0.000000e+00 -3.234821e-06 0 0.000000e+00
> [16,] 0 3.345611e-03 -2.755438e-04 5.982136e-03 0 1.965919e-03
> [,7] [,8] [,9] [,10] [,11]
> [1,] 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00
> [2,] -1.848965e-05 0.000000e+00 -1.613425e-04 -2.361078e-06 0.000000e+00
> [3,] -6.055219e-06 -4.101344e-05 -1.751877e-05 0.000000e+00 -1.606376e-06
[4,] -9.006694e-06 8.804655e-05 0.000000e+00 0.000000e+00 -6.762812e-06
[5,] 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00
[6,] -1.356001e-06 5.694359e-05 -4.033470e-06 -1.954024e-08 -1.172880e-08
[7,] -2.201744e-06 -1.275882e-06 -1.968252e-05 -1.878964e-07 0.000000e+00
[8,] -1.275882e-06 3.816822e-03 0.000000e+00 0.000000e+00 -2.987178e-06
[9,] -1.968252e-05 0.000000e+00 -2.881039e-03 -3.011785e-06 0.000000e+00
[10,] -1.878964e-07 0.000000e+00 -3.011785e-06 -5.586616e-08 0.000000e+00
[11,] 0.000000e+00 -2.987178e-06 0.000000e+00 0.000000e+00 -2.716427e-07
[12,] -4.421662e-07 -8.648474e-06 0.000000e+00 0.000000e+00 -5.027954e-07
[13,] -9.733425e-05 1.804328e-04 -6.977470e-04 -4.823061e-06 0.000000e+00
[14,] -1.376869e-06 0.000000e+00 -1.400386e-05 -1.681941e-07 0.000000e+00
> [15,] 0.000000e+00 1.181955e-05 0.000000e+00 0.000000e+00 -9.657494e-07
> [16,] -2.436425e-05 8.031561e-04 0.000000e+00 0.000000e+00 -1.389764e-05
> [,12] [,13] [,14] [,15] [,16]
> [1,] 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00
> [2,] 0.000000e+00 3.434155e-03 -1.255931e-05 0.000000e+00 3.345611e-03
> [3,] -7.948917e-06 -3.634290e-04 -2.393388e-06 0.000000e+00 -2.755438e-04
> [4,] -2.333796e-05 -7.897535e-03 -1.177230e-06 -3.234821e-06 5.982136e-03
> [5,] 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00
> [6,] -8.638454e-07 2.346580e-03 -7.255730e-07 0.000000e+00 1.965919e-03
> [7,] -4.421662e-07 -9.733425e-05 -1.376869e-06 0.000000e+00 -2.436425e-05
> [8,] -8.648474e-06 1.804328e-04 0.000000e+00 1.181955e-05 8.031561e-04
> [9,] 0.000000e+00 -6.977470e-04 -1.400386e-05 0.000000e+00 0.000000e+00
> [10,] 0.000000 e+00 -4.823061e-06 -1.681941e-07 0.000000e+00 0.000000e+00
> [11,] -5.027954e-07 0.000000e+00 0.000000e+00 -9.657494e-07 -1.389764e-05
> [12,] -1.595388e-06 -4.097081e-05 0.000000e+00 -6.365250e-07 -5.155659e-05
> [13,] -4.097081e-05 -3.880369e-03 -5.643506e-05 0.000000e+00 1.123953e-02
> [14,] 0.000000e+00 -5.643506e-05 -9.255458e-07 0.000000e+00 -5.579944e-06
> [15,] -6.365250e-07 0.000000e+00 0.000000e+00 -5.601956e-05 -4.630112e-07
> [16,] -5.155659e-05 1.123953e-02 -5.579944e-06 -4.630112e-07 1.614308e-02
>
>
> $cont_cv
> $cont_cv[[1]]
```




```
>
> $cont_cv[[3]]
> [,1] [,2] [,3] [,4] [,5] [,6]
> [1,] 0 0.000000e+00 0.000000e+00 0.000000e+00 0 0.000000e+00
> [2,] 0 6.561464e-05 1.667133e-04 6.145001e-04 0 6.803815e-06
> [3,] 0 1.667133e-04 5.312652e-04 1.134999e-03 0 1.305416e-05
> [4,] 0 6.145001e-04 1.134999e-03 4.119634e-03 0 3.754492e-05
> [5,] 0 0.0000000 +00 0.000000e+00 0.000000e+00 0 0.000000e+00
> [6,] 0 6.803815e-06 1.305416e-05 3.754492e-05 0 2.651005e-06
> [7,] 0 1.342477e-04 5.216148e-05 2.180084e-04 0 4.182820e-06
> [8,] 0 2.759892e-04 3.558894e-04 1.621997e-03 0 1.475357e-05
> [9,] 0 -1.342945e-04 2.559914e-05 0.000000e+00 0 2.898283e-07
> [10,] 0 3.968117e-07 7.321574e-06 2.978809e-05 0 2.772963e-08
> [11,] 0 2.273898e-05 1.944061e-05 1.313456e-04 0 2.068190e-07
> [12,] 0 0.000000e+00 1.714937e-04 4.058902e-04 0 1.154662e-05
> [13,] 0 -5.521742e-05 9.196895e-04 2.426485e-03 0 3.018078e-05
> [14,] 0 6.051773e-05 5.019517e-05 1.066757e-04 0 1.699574e-06
> [15,] 0 0.000000e+00 1.000707e-05 7.871110e-06 0 0.000000e+00
> [16,] 0 4.966386e-08 2.313889e-04 9.457904e-04 0 3.127996e-05
> [,7] [,8] [,9] [,10] [,11]
> [1,] 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00
> [2,] 1.342477e-04 2.759892e-04 -1.342945e-04 3.968117e-07 2.273898e-05
> [3,] 5.216148e-05 3.558894e-04 2.559914e-05 7.321574e-06 1.944061e-05
> [4,] 2.180084e-04 1.621997e-03 0.000000e+00 2.978809e-05 1.313456e-04
> [5,] 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00
> [6,] 4.182820e-06 1.475357e-05 2.898283e-07 2.772963e-08 2.068190e-07
> [7,] 5.727226e-05 1.501761e-04 1.586518e-05 5.521417e-06 1.501673e-05
> [8,] 1.501761e-04 1.251886e-03 0.000000e+00 1.353176e-05 1.288851e-04
> [9,] 1.586518e-05 0.0000000e+00 -3.943452e-04 -7.891350e-06 0.000000e+00
> [10,] 5.521417e-06 1.353176e-05 -7.891350e-06 -2.365013e-07 1.169937e-06
> [11,] 1.501673e-05 1.288851e-04 0.000000e+00 1.169937e-06 1.575013e-05
> [12,] 3.831664e-05 4.195024e-04 0.000000e+00 0.000000e+00 4.308234e-05
> [13,] 1.857963e-04 3.330224e-04 -6.937241e-04 -6.861421e-06 0.000000e+00
> [14,] 1.717142e-05 1.739926e-05 9.317337e-06 3.024655e-06 1.966884e-06
> [15,] 0.000000e+00 1.128016e-05 -3.532754e-05 0.000000e+00 1.166176e-06
> [16,] 6.259061e-05 5.411792e-04 0.000000e+00 0.000000e+00 2.324818e-05
> [,12] [,13] [,14] [,15] [,16]
> [1,] 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00
> [2,] 0.000000e+00 -5.521742e-05 6.051773e-05 0.000000e+00 4.966386e-08
> [3,] 1.714937e-04 9.196895e-04 5.019517e-05 1.000707e-05 2.313889e-04
> [4,] 4.058902e-04 2.426485e-03 1.066757e-04 7.871110e-06 9.457904e-04
> [5,] 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00
> [6,] 1.154662e-05 3.018078e-05 1.699574e-06 0.000000e+00 3.127996e-05
> [7,] 3.831664e-05 1.857963e-04 1.717142e-05 0.000000e+00 6.259061e-05
> [8,] 4.195024e-04 3.330224e-04 1.739926e-05 1.128016e-05 5.411792e-04
> [9,] 0.000000e+00 -6.937241e-04 9.317337e-06 -3.532754e-05 0.000000e+00
> [10,] 0.000000e+00 -6.861421e-06 3.024655e-06 0.000000e+00 0.000000e+00
> [11,] 4.308234e-05 0.000000e+00 1.966884e-06 1.166176e-06 2.324818e-05
> [12,] 2.864409e-04 1.900416e-04 0.0000000e+00 3.950020e-06 3.016543e-04
> [13,] 1.900416e-04 -1.115104e-03 1.576644e-04 -3.341760e-05 -1.268062e-04
> [14,] 0.000000e+00 1.576644e-04 1.202314e-05 4.558738e-07 5.728788e-06
> [15,] 3.950020e-06 -3.341760e-05 4.558738e-07 -1.248132e-05 -1.296623e-06
```

```
> [16,] 3.016543e-04 -1.268062e-04 5.728788e-06 -1.296623e-06 -1.505905e-04
>
> $cont_cv[[4]]
> [,1] [,2] [,3] [,4] [,5] [,6]
> [1,] 0 0.000000e+00 0.000000e+00 0.000000e+00 0 0.000000ee+00
> [2,] 0
> [3,] 0 1.227429e-04 5.459642e-04 2.978489e-03 0 -3.186382e-06
> [4,] 0 8.649435e-05 2.978489e-03 5.061030e-02 0 -6.999959e-06
> [5,] 0 0.000000e+00 0.000000e+00 0.000000e+00 0 0.000000e+00
> [6,] 0 -4.748094e-06 -3.186382e-06 -6.999959e-06 0 -5.336882e-07
> [7,] 0 -9.244825e-06 -3.027610e-06 -4.503347e-06 0 -6.780004e-07
> [8,] 0 0.000000e+00 4.249065e-04 6.025473e-03 0 -2.040398e-06
> [9,] 0 1.178928e-03 7.461051e-05 0.000000e+00 0-4.452698e-06
> [10,] 0 7.891974e-06 5.433937e-06 0.000000e+00 0 -9.770120e-09
> [11,] 0 0.000000e+00 2.158838e-05 1.165721e-04 0 -5.864399e-09
> [12,] 0 0.000000e+00 1.196735e-04 6.657378e-04 0 -4.319227e-07
> [13,] 0 1.265050e-04 2.993865e-04 2.328283e-03 0 -3.417864e-05
> [14,] 0 -6.279654e-06 -1.196694e-06 -5.886149e-07 0 -3.627865e-07
> [15,] 0 0.000000e+00 0.000000e+00 -2.653719e-06 0 0.000000e+00
> [16,] 0 1.100022e-04 1.624664e-03 1.937680e-02 0 -1.666335e-05
> [,7] [,8] [,9] [,10] [,11]
> [1,] 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00
> [2,] -9.244825e-06 0.000000e+00 1.178928e-03 7.891974e-06 
> [3,] -3.027610e-06 4.249065e-04 7.461051e-05 5.433937e-06 2.158838e-05
> [4,] -4.503347e-06 6.025473e-03 0.000000e+00 0.000000e+00 1.165721e-04
> [5,] 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00 0.0000000e+00
> [6,] -6.780004e-07 -2.040398e-06 -4.452698e-06 -9.770120e-09 -5.864399e-09
> [7,] -1.100872e-06 -6.379408e-07 -9.841261e-06 -9.394819e-08 0.000000e+00
> [8,] -6.379408e-07 1.059419e-03 0.000000e+00 0.000000e+00 2.142994e-05
> [9,] -9.841261e-06 0.000000e+00 1.380267e-03 1.026221e-05 0.000000ee+00
> [10,] -9.394819e-08 0.000000e+00 1.026221e-05 4.263173e-07 5.192694e-07
> [11,] 0.000000e+00 2.142994e-05 0.000000e+00 5.192694e-07 2.527566e-06
> [12,] -2.210831e-07 9.857867e-05 0.000000e+00 2.077231e-06 8.853500e-06
> [13,] -4.866713e-05 1.464031e-04 1.314424e-04 4.784073e-06 1.150690e-05
> [14,] -6.884346e-07 0.000000e+00 -7.001931e-06 -8.409706e-08 0.000000e+00
> [15,] 0.000000e+00 -2.609384e-06 0.000000e+00 0.000000e+00 -4.828747e-07
> [16,] -1.218213e-05 2.053871e-03 0.000000e+00 4.720945e-06 5.078835e-05
> [,12] [,13] [,14] [,15] [,16]
> [1,] 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00
> [2,] 0.000000e+00 1.265050e-04 -6.279654e-06 0.000000e+00 1.100022e-04
> [3,] 1.196735e-04 2.993865e-04 -1.196694e-06 0.000000e+00 1.624664e-03
> [4,] 6.657378e-04 2.328283e-03 -5.886149e-07 -2.653719e-06 1.937680e-02
> [5,] 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00
> [6,] -4.319227e-07 -3.417864e-05 -3.627865e-07 0.000000e+00 -1.666335e-05
> [7,] -2.210831e-07 -4.866713e-05 -6.884346e-07 0.000000e+00 -1.218213e-05
> [8,] 9.857867e-05 1.464031e-04 0.000000e+00 -2.609384e-06 2.053871e-03
> [9,] 0.000000e+00 1.314424e-04 -7.001931e-06 0.000000e+00 0.000000e+00
> [10,] 2.077231e-06 4.784073e-06 -8.409706e-08 0.000000e+00 4.720945e-06
> [11,] 8.853500e-06 1.150690e-05 0.000000e+00 -4.828747e-07 5.078835e-05
> [12,] 3.953660e-05 1.202313e-04 0.000000e+00 -3.182625e-07 3.124844e-04
> [13,] 1.202313e-04 -8.031171e-03 -2.821753e-05 0.000000e+00 -7.027153e-04
> [14,] 0.000000e+00 -2.821753e-05 -4.627729e-07 0.000000e+00 -2.789972e-06
```

```
> [15,] -3.182625e-07 0.000000e+00 0.000000e+00 -2.713243e-06 -2.339366e-06
> [16,] 3.124844e-04 -7.027153e-04 -2.789972e-06 -2.339366e-06 6.286179e-03
>
> $cont_cv[[5]]
> [,1] [,2] [,3] [,4] [,5] [,6]
> [1,] 0 0.000000e+00 0.000000e+00 0.000000e+00 0 0.000000e+00
> [2,] 0 2.186922e-04 8.214906e-04 9.240429e-04 0 1.514625e-05
> [3,] 0 8.214906e-04 1.342007e-02 1.515950e-02 0 1.118836e-04
> [4,] 0 9.240429e-04 1.515950e-02 3.443875e-02 0 2.967172e-04
> [5,] 0 0.000000e+00 0.000000e+00 0.000000e+00 0 0.000000e+00
> [6,] 0 1.514625e-05 1.118836e-04 2.967172e-04 0 2.778427e-06
> [7,] 0 5.634711e-05 1.136717e-04 1.963181e-04 0 5.317863e-06
> [8,] 0 1.650553e-04 3.505316e-03 6.965150e-03 0 4.963554e-05
> [9,] 0 1.730332e-03 9.893722e-03 1.024567e-02 0 1.186528e-04
> [10,] 0 3.610492e-06 0.000000e+00 0.000000e+00 0 4.094508e-08
> [11,] 0 0.000000e+00 7.252853e-05 1.715682e-04 0 4.279235e-08
> [12,] 0 0.000000e+00 2.517339e-04 6.060791e-04 0 3.806287e-06
> [13,] 0 3.081734e-03 1.890804e-02 3.639987e-02 0 4.135629e-04
> [14,] 0 6.060981e-05 2.147643e-04 4.147659e-04 0 6.447330e-06
> [15,] 0 0.000000e+00 0.000000e+00 9.481893e-05 0 0.000000e+00
> [16,] 0 1.921807e-04 3.453292e-03 1.240334e-02 0 9.024762e-05
> [,7] [,8] [,9] [,10] [,11]
> [1,] 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00
> [2,] 5.634711e-05 1.650553e-04 1.730332e-03 3.610492e-06 0.000000e+00
> [3,] 1.136717e-04 3.505316e-03 9.893722e-03 0.000000e+00 7.252853e-05
> [4,] 1.963181e-04 6.965150e-03 1.024567e-02 0.000000e+00 1.715682e-04
> [5,] 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00
> [6,] 5.317863e-06 4.963554e-05 1.186528e-04 4.094508e-08 4.279235e-08
> [7,] 4.500508e-05 1.415121e-05 2.307315e-04 3.146218e-06 7.068059e-06
> [8,] 1.415121e-05 1.477724e-03 2.402628e-03 0.000000e+00 4.483901e-05
> [9,] 2.307315e-04 2.402628e-03 2.097028e-02 1.873252e-05 0.000000e+00
> [10,] 3.146218e-06 0.000000e+00 1.873252e-05 3.631716e-07 6.123209e-07
> [11,] 7.068059e-06 4.483901e-05 0.000000e+00 6.123209e-07 8.361588e-06
> [12,] 2.584984e-05 1.221048e-04 0.000000e+00 1.886436e-06 1.497551e-05
> [13,] 7.628549e-04 6.429194e-03 2.397756e-02 2.103676e-05 0.000000e+00
> [14,] 1.334649e-05 7.645985e-05 4.116451e-04 9.223849e-07 0.000000e+00
> [15,] 2.826756e-05 4.773172e-05 0.000000e+00 2.543093e-06 1.671065e-05
> [16,] 8.967669e-05 2.169055e-03 2.225626e-03 0.000000e+00 4.210868e-05
> [,12] [,13] [,14] [,15] [,16]
> [1,] 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00
> [2,] 0.000000e+00 3.081734e-03 6.060981e-05 0.000000e+00 1.921807e-04
> [3,] 2.517339e-04 1.890804e-02 2.147643e-04 0.000000e+00 3.453292e-03
> [4,] 6.060791e-04 3.639987e-02 4.147659e-04 9.481893e-05 1.240334e-02
> [5,] 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00
> [6,] 3.806287e-06 4.135629e-04 6.447330e-06 0.000000e+00 9.024762e-05
> [7,] 2.584984e-05 7.628549e-04 1.334649e-05 2.826756e-05 8.967669e-05
> [8,] 1.221048e-04 6.429194e-03 7.645985e-05 4.773172e-05 2.169055e-03
> [9,] 0.000000e+00 2.397756e-02 4.116451e-04 0.000000e+00 2.225626e-03
> [10,] 1.886436e-06 2.103676e-05 9.223849e-07 2.543093e-06 0.000000e+00
> [11,] 1.497551e-05 0.000000e+00 0.000000e+00 1.671065e-05 4.210868e-05
> [12,] 4.252656e-05 3.863758e-04 0.000000e+00 3.253185e-05 2.049827e-04
> [13,] 3.863758e-04 6.092687e-02 9.607362e-04 0.000000e+00 1.205494e-02
```

```
> [14,] 0.000000e+00 9.607362e-04 1.764852e-05 0.000000e+00 1.522182e-04
> [15,] 3.253185e-05 0.000000e+00 0.000000e+00 7.099986e-05 1.342205e-05
> [16,] 2.049827e-04 1.205494e-02 1.522182e-04 1.342205e-05 3.847039e-03
>
> $cont_cv[[6]]
> [,1] [,2] [,3] [,4] [,5] [,6]
> [1,] 0 0.000000e+00 0.000000e+00 0.000000e+00 0 0.0000000e+00
> [2,] 0
> [3,] 0 7.322215e-05 3.248397e-04 1.210390e-04 0-2.812943e-05
> [4,] 0 1.488615e-04 1.210390e-04 4.580279e-04 0-2.217178e-05
> [5,] 0 0.000000e+00 0.000000e+00 0.000000e+00 0 0.000000e+00
> [6,] 0 -2.649522e-04 -2.812943e-05 -2.217178e-05 0 -1.623512e-04
> [7,] 0 2.661410e-04 9.588160e-05 2.629802e-05 0 -1.306995e-05
> [8,] 0 1.438229e-06 6.113476e-05 6.744676e-05 0 -4.534679e-05
> [9,] 0 2.414007e-03 1.293467e-04 4.636022e-04 0 -2.946997e-05
> [10,] 0 1.027713e-04 0.000000e+00 1.315238e-05 0
> [11,] 0 0.000000e+00 1.033163e-04 3.507868e-05 0
> [12,] 0 0.000000e+00 -3.974459e-06 -1.166898e-05 0 -4.319227e-07
> [13,] 0 1.045309e-03 3.845520e-04 1.126217e-03 0-1.418423e-04
> [14,] 0 1.085610e-04 1.407154e-05 2.611105e-05 0-1.157586e-05
> [15,] 0 0.000000e+00 8.274998e-05 8.042606e-06 0
> [16,] 0 -2.724399e-05 -9.849824e-05 -7.000508e-05 0 -8.494265e-05
> [,7] [,8] [,9] [,10] [,11]
> [1,] 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00
> [2,] 2.661410e-04 1.438229e-06 2.414007e-03 1.027713e-04 0.000000e+00
> [3,] 9.588160e-05 6.113476e-05 1.293467e-04 0.000000e+00 1.033163e-04
> [4,] 2.629802e-05 6.744676e-05 4.636022e-04 1.315238e-05 3.507868e-05
> [5,] 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00
> [6,] -1.306995e-05 -4.534679e-05 -2.946997e-05 1.463100e-06 2.715708e-07
> [7,] 3.392465e-04 9.656073e-06 8.086247e-04 3.828697e-05 5.478993e-05
> [8,] 9.656073e-06 -1.871453e-05 0.000000e+00 0.000000e+00 4.816933e-05
> [9,] 8.086247e-04 0.000000e+00 6.605233e-03 2.791002e-04 0.000000e+00
> [10,] 3.828697e-05 0.000000e+00 2.791002e-04 1.489885e-05 0.000000e+00
> [11,] 5.478993e-05 4.816933e-05 0.000000e+00 0.000000e+00 9.272783e-05
> [12,] -2.210831e-07 -4.324237e-06 0.000000e+00 0.000000e+00 -2.513977e-07
> [13,] 4.473256e-04 6.836944e-05
> [14,] 3.855365e-05 6.115778e-06
> [15,] 0.000000e+00 1.393610e-05
> [16,] -2.501411e-06 -1.277806e-04
> [,12] [,13]
> [1,] 0.000000e+00 0.000000e+00
> [2,] 0.000000e+00 1.045309e-03
> [3,] -3.974459e-06 3.845520e-04
> [4,] -1.166898e-05 1.126217e-03
> [5,] 0.000000e+00 0.000000e+00
> [6,] -4.319227e-07 -1.418423e-04 -
> [7,] -2.210831e-07 4.473256e-04
> [8,] -4.324237e-06 6.836944e-05
> [9,] 0.000000e+00 3.549238e-03
> [10,] 0.000000e+00 9.966229e-05
> [11,] -2.513977e-07 0.000000e+00
> [12,] -7.976940e-07 -2.048541e-05 0.000000e+00 -3.182625e-07 -2.577829e-05
```



```
> [12,] -7.976940e-07 -2.048541e-05 0.000000e+00 -3.182625e-07 -2.577829e-05
> [13,] -2.048541e-05 6.011797e-02 1.617952e-03 0.000000e+00 1.256641e-02
> [14,] 0.000000e+00 1.617952e-03 5.215953e-05 0.000000e+00 1.248913e-04
> [15,] -3.182625e-07 0.000000e+00 0.000000e+00 -2.713243e-06 -2.339366e-06
> [16,] -2.577829e-05 1.256641e-02 1.248913e-04 -2.339366e-06 1.987251e-02
>
> $cont_cv[[8]]
> [,1] [,2] [,3] [,4] [,5] [,6]
> [1,] 0 0.000000e+00 0.000000e+00 0.000000e+00 0 0.000000ee+00
> [2,] 0 1.699246e-04 3.744591e-04 2.611540e-06 0-4.748094e-06
> [3,] 0 3.744591e-04 2.095984e-03 4.850717e-04 0-3.186382e-06
> [4,] 0 2.611540e-06 4.850717e-04 1.964089e-04 0-6.999959e-06
> [5,] 0 0.000000e+00 0.000000e+00 0.000000e+00 0 0.000000e+00
> [6,] 0 -4.748094e-06 -3.186382e-06 -6.999959e-06 0 -5.336882e-07
> [7,] 0 1.093175e-04 2.887397e-04 2.834944e-05 0 -6.780004e-07
> [8,] 0 1.514483e-04 8.383701e-04 1.953753e-04 0 -2.040398e-06
> [9,] 0
> [10,] 0 -1.180539e-06 0.000000e+00 0.000000e+00 0-9.770120e-09
> [11,] 0 0.000000e+00 -8.031880e-07 -3.381406e-06 0 -5.864399e-09
> [12,] 0 0.000000e+00 1.690513e-04 1.754537e-04 0-4.319227e-07
> [13,] 0 -4.992734e-04 -1.835372e-04 -5.242070e-04 0 -3.417864e-05
> [14,] 0 8.264203e-05 5.135574e-05 4.695502e-06 0 -3.627865e-07
> [15,] 0 0.000000e+00 0.000000e+00 -2.653719e-06 0 0.000000e+00
> [16,] 0 5.878985e-05 3.387565e-03 3.189316e-04 0 -1.666335e-05
> [,7] [,8] [,9] [,10] [,11]
> [1,] 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00
> [2,] 1.093175e-04 1.514483e-04 1.436267e-04 -1.180539e-06 0.000000e+00
> [3,] 2.887397e-04 8.383701e-04 9.697368e-05 0.000000e+00 -8.031880e-07
> [4,] 2.834944e-05 1.953753e-04 1.419697e-05 0.000000e+00 -3.381406e-06
> [5,] 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00
> [6,] -6.780004e-07 -2.040398e-06 -4.452698e-06 -9.770120e-09 -5.864399e-09
> [7,] 6.795997e-05 8.757861e-05 8.104251e-05 -9.394819e-08 0.000000e+00
> [8,] 8.757861e-05 4.490046e-04 0.000000e+00 0.000000e+00 -1.493589e-06
> [9,] 8.104251e-05 0.000000e+00 4.827564e-04 -1.505893e-06 0.000000e+00
> [10,] -9.394819e-08 0.000000e+00 -1.505893e-06 -2.793308e-08 0.000000e+00
> [11,] 0.000000e+00 -1.493589e-06 0.000000e+00 0.000000e+00 -1.358213e-07
> [12,] 8.990633e-06 9.697468e-05 4.407806e-05 0.000000e+00 -2.513977e-07
> [13,] -6.281520e-05 -1.163632e-04 -9.741690e-04 -2.411530e-06 0.000000e+00
> [14,] 2.882234e-05 1.319673e-13 9.888604e-05 -8.409706e-08 0.000000e+00
> [15,] 0.000000e+00 -2.609384e-06 0.000000e+00 0.000000e+00 -4.828747e-07
> [16,] 5.273201e-04 1.279543e-03 0.000000e+00 0.000000e+00 -6.948820e-06
> [,12] [,13] [,14] [,15] [,16]
> [1,] 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00
> [2,] 0.000000e+00 -4.992734e-04 8.264203e-05 0.000000e+00 5.878985e-05
> [3,] 1.690513e-04 -1.835372e-04 5.135574e-05 0.000000e+00 3.387565e-03
> [4,] 1.754537e-04 -5.242070e-04 4.695502e-06 -2.653719e-06 3.189316e-04
> [5,] 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00
> [6,] -4.319227e-07 -3.417864e-05 -3.627865e-07 0.000000e+00 -1.666335e-05
> [7,] 8.990633e-06 -6.281520e-05 2.882234e-05 0.000000e+00 5.273201e-04
> [8,] 9.697468e-05 -1.163632e-04 1.319673e-13 -2.609384e-06 1.279543e-03
> [9,] 4.407806e-05 -9.741690e-04 9.888604e-05 0.000000e+00 0.000000e+00
> [10,] 0.000000e+00 -2.411530e-06 -8.409706e-08 0.000000e+00 0.000000e+00
```



```
> [10,] 0.000000e+00 -2.411530e-06 -8.409706e-08 0.000000e+00 0.000000e+00
> [11,] -2.513977e-07 0.000000e+00 0.000000e+00 -4.828747e-07 -6.948820e-06
> [12,] -7.976940e-07 -2.048541e-05 0.000000e+00 -3.182625e-07 -2.577829e-05
> [13,] -2.048541e-05 5.783399e-02 -2.821753e-05 0.000000e+00 2.326858e-02
> [14,] 0.000000e+00 -2.821753e-05 -4.627729e-07 0.000000e+00 -2.789972e-06
> [15,] -3.182625e-07 0.000000e+00 0.000000e+00 7.780972e-05 2.255881e-05
> [16,] -2.577829e-05 2.326858e-02 -2.789972e-06 2.255881e-05 1.456702e-02
>
>
> $cont_corr
> $cont_corr[[1]]
> [,1] [,2] [,3] [,4] [,5] [,6]
> [1,] 0 0.000000e+00 0.000000e+00 0.000000e+00 0 0.000000e+00
> [2,] 0 2.300178e-19 2.143679e-05 -5.388917e-05 0 8.211573e-06
> [3,] 0 2.143679e-05 0.000000e+00 -3.077477e-04 0 -2.518930e-06
> [4,] 0 -5.388917e-05 -3.077477e-04 5.828148e-19 0 -9.538747e-05
> [5,] 0 0.000000e+00 0.0000000e+00 0.000000e+00 0 0.0000000e+00
> [6,] 0 8.211573e-06 -2.518930e-06 -9.538747e-05 0 0.000000e+00
> [7,] 0 -5.346499e-05 5.787234e-05 -2.706236e-05 0 4.745604e-06
> [8,] 0 2.692399e-04 3.462178e-05 -4.179439e-04 0 2.208194e-05
[9,] 0 -1.185447e-03 -1.472976e-04 0.000000e+00 0 -8.689344e-05
[10,] 0-1.180539e-06 0.000000e+00 0.000000e+00 0-9.770120e-09
[11,] 0 0.000000e+00 -2.545510e-05 -5.030837e-06 0 -2.180144e-07
[12,] 0 5.158981e-04 -4.951632e-05 -8.648673e-04 0 2.635835e-05
[13,] 0 -2.523640e-03 -1.840467e-03 -4.370038e-03 0 -3.839803e-04
[14,] 0 -9.471790e-05 -2.402122e-05 -1.320558e-05 0 -8.398019e-06
[15,] 0 0.000000e+00 0.000000e+00 4.280447e-04 0
[16,] 0 1.878580e-03 -1.482875e-04 -4.375662e-03 0 7.798457e-05
> 
> [2,] -5.346499e-05 2.692399e-04 -1.185447e-03 -1.180539e-06 0.0000000e+00
> [3,] 5.787234e-05 3.462178e-05 -1.472976e-04 0.000000e+00 -2.545510e-05
> [4,] -2.706236e-05 -4.179439e-04 0.000000e+00 0.000000e+00 -5.030837e-06
> [5,] 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00
> [6,] 4.745604e-06 2.208194e-05 -8.689344e-05 -9.770120e-09 -2.180144e-07
> [7,] -1.284441e-20 1.218537e-04 -3.514833e-04 -9.394819e-08 0.000000e+00
> [8,] 1.218537e-04 -7.830059e-20 0.000000e+00 0.000000e+00 -3.177722e-05
> [9,] -3.514833e-04 0.000000e+00 -3.408834e-19 -1.505893e-06 0.000000e+00
> [10,] -9.394819e-08 0.000000e+00 -1.505893e-06 -2.793308e-08 0.000000e+00
> [11,] 0.000000e+00 -3.177722e-05 0.000000e+00 0.000000e+00 -7.575815e-22
> [12,] 1.743254e-04 -2.134047e-05 0.000000e+00 0.000000e+00 -4.129742e-05
> [13,] -9.398358e-04 -6.483076e-04 1.074093e-03 -2.411530e-06 0.000000e+00
> [14,] -2.936351e-05 0.000000e+00 -1.411952e-05 -8.409706e-08 1.113267e-06
> [15,] 0.000000e+00 -5.726985e-05 0.000000e+00 0.000000e+00 6.386759e-06
> [16,] 5.428293e-04 2.334148e-05 0.000000e+00 0.000000e+00 -1.572962e-04
> [,12] [,13] [,14] [,15] [,16]
> [1,] 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00
> [2,] 5.158981e-04 -2.523640e-03 -9.471790e-05 0.000000e+00 1.878580e-03
> [3,] -4.951632e-05 -1.840467e-03 -2.402122e-05 0.000000e+00 -1.482875e-04
> [4,] -8.648673e-04 -4.370038e-03 -1.320558e-05 4.280447e-04 -4.375662e-03
> [5,] 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00
> [6,] 2.635835e-05 -3.839803e-04 -8.398019e-06 0.000000e+00 7.798457e-05
```



|  | [6,] | 06 | -2.793489e-04 | 07 | 2 | $1.637126 \mathrm{e}-05$ |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| > | [7,] | $1.433759 \mathrm{e}-05$ | -2.379964e-04 | -6.884346e-07 | $0.000000 \mathrm{e}+00$ | -1.258126e-04 |
| > | [8,] | -6.102347e-05 | $2.130089 \mathrm{e}-03$ | $0.000000 \mathrm{e}+00$ | -5.638546e-05 | -8.758039e-04 |
| > | [9,] | $0.000000 \mathrm{e}+00$ | $2.771857 \mathrm{e}-04$ | -7.001931e-06 | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |
| $>$ | [10,] | $1.897913 \mathrm{e}-06$ | -4.751683e-05 | -8.409706e-08 | $6.194770 \mathrm{e}-06$ | $3.166226 \mathrm{e}-05$ |
| > | [11,] | -2.724796e-06 | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $2.755739 \mathrm{e}-06$ | $2.416689 \mathrm{e}-05$ |
| > | [12,] | -1.876587e-21 | -3.082990e-04 | $0.000000 \mathrm{e}+00$ | $6.012757 \mathrm{e}-06$ | -9.242086e-05 |
| > | [13,] | -3.082990e-04 | $0.000000 \mathrm{e}+00$ | $-2.821753 \mathrm{e}-05$ | $0.000000 \mathrm{e}+00$ | -3.776841e-03 |
| > | [14,] | $0.000000 \mathrm{e}+00$ | -2.821753e-05 | $-4.627729 \mathrm{e}-07$ | $0.000000 \mathrm{e}+00$ | -2.789972e-06 |
| > | [15,] | $6.012757 \mathrm{e}-06$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $1.114035 \mathrm{e}-20$ | $2.584012 \mathrm{e}-04$ |
| > | [16,] | -9.242086e-05 | -3.776841e-03 | -2.789972e-06 | $2.584012 \mathrm{e}-04$ | $0.000000 \mathrm{e}+00$ |
|  |  |  |  |  |  |  |
| \$cont_corr[[3]] |  |  |  |  |  |  |
|  |  |  | [,2] | [,3] | [,4] [,5] | [, ${ }^{\text {] }}$ |
| $>$ | [1, ] | 00.00000 | +00 0.0000 | e+00 0.000000 | +00 0 | 00000+00 |
|  | [2,] | 01.89139 | -19 8.5827 | -05 1.1434 | -03 0-2 | 44842e-05 |
|  | [3,] | 08.5827 | -05-4.8183 | -20-2.8063 | -04 0-2 | 84184e-05 |
|  | [4, ] | 01.1434 | -03-2.8063 | 042.8837 | $190-5$ | 02698e-05 |
|  | [5,] | 00.00000 | +00 0.0000 | +00 0.0000 | +00 0 | 00000e+00 |
|  | [6, ] | $0-2.24484$ | -05-2.2841 | -05-5.5026 | -05 0 | 00000e+00 |
| $>$ | [7, ] | $0-2.16395$ | -05-7.7760 | -05 9.9536 | -05 0-5 | 81279e-06 |
|  | [8,] | 06.18949 | -04 -5.2872 | -04-4.8668 | -04 0-1 | 27465e-05 |
|  | [9,] | $0-5.43791$ | -04 1.8432 | -04 0.0000 | +00 0 -1 | 05251e-05 |
|  | [10,] | 01.17829 | -05 2.2294 | -05 6.1889 | -05 0-1. | 21025e-07 |
|  | [11,] | 09.16133 | -05 -3.66708 | -05-1.0341 | -05 0 | 10703e-07 |
|  | [12,] | 00.00000 | +00 -2.40682 | -04 -5.12684 | -04 0 | 88063e-06 |
|  | [13,] | $0-6.486015$ | 5e-04 3.5011 | -04 7.63266 | -04 0-1 | 193733e-04 |
|  | [14,] | $0-1.397939$ | 9-05 3.227167 | 7e-05 1.23992 | -04 0-2 | 756863e-06 |
|  | [15,] | 00.000000 | e+00 4.92883 | e-05 -2.12176 | -05 0 | .000000e+00 |
|  | $[16$, | $0-1.052548 \mathrm{e}-04$ |  | -1.976812e-03 | -03 $01.163909 \mathrm{e}-05$ |  |
| > | $[, 7]$$0.000000 \mathrm{e}+00$ |  | [,8] | $[, 9]$ | [,11] |  |
| $>$ |  |  | $0.000000 \mathrm{e}+00$ |  | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |
| > | [2,] | -2.163951e-05 | $6.189491 \mathrm{e}-04$ | -5.437912e-04 | $1.178294 \mathrm{e}-05$ | $9.161336 \mathrm{e}-05$ |
| > | [3, ] | -7.776083e-05 | -5.287245e-04 | $1.843246 \mathrm{e}-04$ | $2.229487 \mathrm{e}-05$ | -3.667089e-05 |
| > | [4, ] | $9.953644 \mathrm{e}-05$ | -4.866843e-04 | $0.000000 \mathrm{e}+00$ | $6.188988 \mathrm{e}-05$ | -1.034166e-05 |
|  | [5, ] | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |
| > | [6, ] | -5.781279e-06 | -1.427465e-05 | -1.305251e-05 | -1.021025e-07 | $2.910703 \mathrm{e}-07$ |
| > | [7, ] | $0.000000 \mathrm{e}+00$ | $1.612362 \mathrm{e}-04$ | -8.689805e-05 | $1.831441 \mathrm{e}-06$ | $2.510385 \mathrm{e}-05$ |
| > | [8, ] | $1.612362 \mathrm{e}-04$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $3.237485 \mathrm{e}-05$ | $1.968190 \mathrm{e}-05$ |
| > | [9, ] | -8.689805e-05 | $0.000000 \mathrm{e}+00$ | -1.768348e-19 | $-2.829241 \mathrm{e}-05$ | $0.000000 \mathrm{e}+00$ |
| > | [10,] | $1.831441 \mathrm{e}-06$ | $3.237485 \mathrm{e}-05$ | -2.829241e-05 | $0.000000 \mathrm{e}+00$ | $5.423454 \mathrm{e}-06$ |
| > | [11,] | $2.510385 \mathrm{e}-05$ | $1.968190 \mathrm{e}-05$ | $0.000000 \mathrm{e}+00$ | $5.423454 \mathrm{e}-06$ | $4.001784 \mathrm{e}-21$ |
| > | [12,] | $1.189762 \mathrm{e}-05$ | -2.064253e-04 | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $-1.835910 \mathrm{e}-05$ |
| > | [13,] | -4.492965e-04 | -8.300523e-04 | -2.189055e-04 | $5.618236 \mathrm{e}-05$ | $0.000000 \mathrm{e}+00$ |
| > | [14,] | -1.103738e-05 | 2.452670 -05 | -3.151644e-05 | 8.251013e-07 | $3.401144 \mathrm{e}-06$ |
| > | [15,] | $0.000000 \mathrm{e}+00$ | -3.753529e-05 | $8.869140 \mathrm{e}-05$ | $0.000000 \mathrm{e}+00$ | -1.391203e-05 |
| > | [16,] | -6.996685e-06 | -4.927299e-04 | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $-2.375882 e-05$ |
| > |  | [,12] | [,13] | [,14] | [,15] | [,16] |
|  | [1,] | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |
|  | [2,] | $0.000000 \mathrm{e}+00$ | -6.486015e-04 | -1.397939e-05 | $0.000000 \mathrm{e}+00$ | $-1.052548 \mathrm{e}-04$ |
|  | [3,] | -2.406825e-04 | $3.501120 \mathrm{e}-04$ | $3.227167 \mathrm{e}-05$ | $4.928832 \mathrm{e}-05$ | -7.118087e-04 |
|  | [4, ] | -5.126843e-04 | $7.632664 \mathrm{e}-04$ | $1.239926 \mathrm{e}-04$ | -2.121769e-05 | -1.976812e-03 |



```
> [4,] -7.052581e-04 -3.983330e-03 -5.886149e-07 -2.653719e-06 -2.889366e-03
> [5,] 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00
> [6,] -4.319227e-07 -3.417864e-05 -3.627865e-07 0.000000e+00 -1.666335e-05
> [7,] -2.210831e-07 -4.866713e-05 -6.884346e-07 0.000000e+00 -1.218213e-05
> [8,] -1.086669e-04 -3.791010e-04 0.000000e+00 -2.609384e-06 -3.255091e-04
[9,] 0.000000e+00 -1.149406e-03 -7.001931e-06 0.000000e+00 0.000000e+00
[10,] 2.463599e-06 2.017173e-05 -8.409706e-08 0.000000e+00 7.835168e-06
[11,] 1.384217e-06 6.577628e-05 0.000000e+00 -4.828747e-07 -4.384926e-05
> [12,] 4.863441e-21 1.163584e-04 0.000000e+00 -3.182625e-07 -2.863408e-04
> [13,] 1.163584e-04 -7.210958e-19 -2.821753e-05 0.000000e+00 -3.018754e-03
> [14,] 0.000000e+00 -2.821753e-05 -4.627729e-07 0.000000e+00 -2.789972e-06
> [15,] -3.182625e-07 0.000000e+00 0.000000e+00 -2.713243e-06 -2.339366e-06
> [16,] -2.863408e-04 -3.018754e-03 -2.789972e-06 -2.339366e-06 -1.773719e-18
>
> $cont_corr[[5]]
> [,1] [,2] [,3] [,4] [,5] [,6]
> [1,] 0 0.000000e+00 0.000000e+00 0.000000e+00 0 0.000000ee+00
> [2,] 0 9.515467e-20 3.223705e-04 9.789013e-04 0-5.801361e-06
> [3,] 0 3.223705e-04 -1.700723e-18 3.136264e-04 0 7.557557e-06
> [4,] 0 9.789013e-04 3.136264e-04 1.966174e-18 0
> [5,] 0 0.000000e+00 0.000000e+00 0.000000e+00 0 0.000000ee+00
> [6,] 0 -5.801361e-06 7.557557e-06 5.231700e-05 0 0.000000e+00
> [7,] 0 -7.994849e-05 -1.276128e-04 -2.070616e-04 0 -6.836470e-06
> [8,] 0 2.420552e-04 2.563053e-04 8.040593e-05 0 1.761598e-05
[9,] 0 -7.944449e-04 8.423085e-03 1.116988e-02 0 4.722592e-05
> [10,] 0 -6.554186e-06 0.000000e+00 0.000000e+00 0 -6.257674e-08
> [11,] 0 0.000000e+00 -8.646688e-05 -1.860184e-04 0 -6.281203e-08
> [12,] 0 0.000000e+00 -2.828327e-04 -6.394810e-04 0 -4.901785e-06
> [13,] 0 -1.204076e-03 2.696046e-03 1.103961e-02 0 7.557862e-06
> [14,] 0 -2.603361e-05 1.139113e-04 3.853005e-04 0 1.327915e-06
> [15,] 0 0.000000e+00 0.000000e+00 -1.013754e-04 0 0.000000e+00
> [16,] 0 2.647255e-04 -9.194970e-06 -2.798478e-05 0 1.994534e-05
> [,7] [,8] [,9] [,10] [,11]
> [1,] 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00
> [2,] -7.994849e-05 2.420552e-04 -7.944449e-04 -6.554186e-06 0.000000e+00
> [3,] -1.276128e-04 2.563053e-04 8.423085e-03 0.000000e+00 -8.646688e-05
> [4,] -2.070616e-04 8.040593e-05 1.116988e-02 0.000000e+00 -1.860184e-04
> [5,] 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00
> [6,] -6.836470e-06 1.761598e-05 4.722592e-05 -6.257674e-08 -6.281203e-08
> [7,] -2.764196e-21 -1.580450e-05 -2.718837e-04 1.126678e-06 8.241822e-06
> [8,] -1.580450e-05 -1.851092e-19 2.874564e-03 0.000000e+00 -5.303493e-05
> [9,] -2.718837e-04 2.874564e-03 -3.040410e-18 -2.467656e-05 0.000000e+00
> [10,] 1.126678e-06 0.000000e+00 -2.467656e-05 8.038932e-23 7.924374e-07
> [11,] 8.241822e-06 -5.303493e-05 0.000000e+00 7.924374e-07 1.172725e-21
> [12,] 1.769432e-05 -1.364742e-04 0.000000e+00 2.237316e-06 2.341375e-06
> [13,] -8.803550e-04 3.400009e-03 4.329107e-03 -2.675170e-05 0.000000e+00
> [14,] -1.493585e-05 8.645990e-05 4.723636e-06 -1.113747e-06 0.000000e+00
> [15,] 3.209461e-05 -5.483447e-05 0.000000e+00 3.146366e-06 2.309211e-06
> [16,] -1.159248e-04 4.944010e-05 3.386580e-03 0.000000e+00 -6.234062e-05
> [,12] [,13] [,14] [,15] [,16]
> [1,] 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00
> [2,] 0.000000e+00 -1.204076e-03 -2.603361e-05 0.000000e+00 2.647255e-04
```





|  |  | 12] | 13] | 4] | 15] | 6] |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| > | [1, ] | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |
| > | [2,] | $0.000000 \mathrm{e}+00$ | -5.392020e-04 | -5.120266e-05 | $0.000000 \mathrm{e}+00$ | $2.573994 \mathrm{e}-05$ |
| > | [3,] | -1.816419e-04 | -2.671515e-04 | -5.532473e-05 | $0.000000 \mathrm{e}+00$ | -2.725877e-04 |
| > | [4, ] | -1.236881e-05 | -4.182499e-04 | -6.986593e-06 | $-2.653719 \mathrm{e}-06$ | -1.495193e-03 |
| > | [5, ] | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |
| > | [6, ] | -4.319227e-07 | -3.417864e-05 | -3.627865e-07 | $0.000000 \mathrm{e}+00$ | -1.666335e-05 |
| > | [7, ] | -9.954508e-06 | -8.174011e-05 | -3.203069e-05 | $0.000000 \mathrm{e}+00$ | 3.109782e-04 |
| > | [8,] | -1.148326e-04 | -1.363258e-04 | $1.572352 \mathrm{e}-13$ | -2.609384e-06 | -8.882060e-04 |
| > | [9,] | $5.723757 \mathrm{e}-05$ | $1.558532 \mathrm{e}-04$ | $-4.206300 \mathrm{e}-05$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |
| > | [10,] | $0.000000 \mathrm{e}+00$ | -2.411530e-06 | -8.409706e-08 | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |
| > | [11,] | -2.513977e-07 | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | -4.828747e-07 | -6.948820e-06 |
| > | [12,] | $0.000000 \mathrm{e}+00$ | $2.027330 \mathrm{e}-06$ | $0.000000 \mathrm{e}+00$ | -3.182625e-07 | -8.985395e-04 |
| > | [13,] | $2.027330 \mathrm{e}-06$ | $0.000000 \mathrm{e}+00$ | -2.682267e-05 | $0.000000 \mathrm{e}+00$ | $2.911801 \mathrm{e}-03$ |
| > | [14,] | $0.000000 \mathrm{e}+00$ | -2.682267e-05 | -3.698077e-21 | $0.000000 \mathrm{e}+00$ | 04 |
| > | [15,] | -3.182625e-07 | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | -2.713243e-06 | -2.339366e-06 |
| > | $[16]-,8.985395 \mathrm{e}-04-2.911801 \mathrm{e}-03-1.046510 \mathrm{e}-04-2.339366 \mathrm{e}-06 \quad 0.000000 \mathrm{e}+00$ |  |  |  |  |  |
| [16,] 8.985395e-04-2.911801e-03-1.046510e-04-2.339366e-06-0.000000e00 |  |  |  |  |  |  |
| \$cont_corr [[9]] |  |  |  |  |  |  |
| > |  | [,1] | [,2] | [,3] | [,4] [,5] | [,6] |
| > | [1, ] | 00.00000 | e+00 0.0000 | +00 0.0000 | +00 0 | 0000e+00 |
| > | [2,] | 03.11529 | $7 \mathrm{e}-19-1.46623$ | 6-05 3.01868 | -03 0 | 0642e-05 |
| $>$ | [3, ] | $0-1.46623$ | 6e-05 -2.3370 | 5e-05-6.04915 | -05 0-3 | 86382e-06 |
| $>$ | [4, ] | 03.01868 | 4e-03-6.0491 | -05-05 8.55947 | -18 0 | 55744e-04 |
| > | [5,] | 00.00000 | e+00 0.000000 | e+00 0.00000 | +00 0 | 00000e+00 |
| > | [6, ] | 06.130642 | 2e-05-3.186382 | $2 \mathrm{e}-062.45574$ | -04 0-1 | 57966e-20 |
| > | [7, ] | $0-9.244825$ | 5-06 -3.0276 | e-06-4.503347 | -06 0-6 | 80004e-07 |
| > | [8,] | 00.000000 | e+00-2.05067 | e-05 -7.636251 | -03 0-3 | . $48627 \mathrm{e}-04$ |
| $>$ | [9,] | $0-6.89571$ | 1e-04-8.759383 | e-06 0.00000 | +00 0-5 | .008135e-05 |
|  | [10,] | $0-1.180539$ | e-06 0.000000 | e+00 0.00000 | +00 $0-9$ | 770120e-09 |
|  | [11,] | 00.000000 | e+00-8.03188 | e-07-3.38140 | e-06 0-5 | 864399e-09 |
|  | [12,] | 00.000000 | 0e+00 -3.97445 | e-06-1.16689 | -05 0-4 | 19227e-07 |
|  | [13,] | $0-1.616056$ | 6e-04 -1.81714 | 5-04 1.234967 | -02 0 3 | 55946e-05 |
|  | [14,] | $0-6.279654$ | 4e-06-1.196694 | $4 \mathrm{e}-06-5.886149$ | -07 0-3. | 627865e-07 |
|  | [15,] | 00.000000 | e+00 0.000000 | e+00 -1.278891 | -04 0 0 | .000000e+00 |
|  | [16,] | 03.390146 | e-03-1.377719 | 9e-04-6.774324 | -04 02 | 6787e-04 |
| > |  | [,7] | [,8] | [,9] | [,10] | [,11] |
| > | [1, ] | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |
| > | [2,] | -9.244825e-06 | $0.000000 \mathrm{e}+00$ | -6.895711e-04 | -1.180539e-06 | $0.000000 \mathrm{e}+00$ |
| $>$ | [3, ] | -3.027610e-06 | -2.050672e-05 | -8.759383e-06 | $0.000000 \mathrm{e}+00$ | -8.031880e-07 |
| > | [4, ] | -4.503347e-06 | -7.636251e-03 | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | -3.381406e-06 |
| > | [5, ] | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ |
| > | [6, ] | -6.780004e-07 | -3.048627e-04 | -5.008135e-05 | -9.770120e-09 | -5.864399e-09 |
| > | [7, ] | -1.100872e-06 | -6.379408e-07 | -9.841261e-06 | -9.394819e-08 | $0.000000 \mathrm{e}+00$ |
| > | [8,] | -6.379408e-07 | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | -1.493589e-06 |
| > | [9, ] | -9.841261e-06 | $0.000000 \mathrm{e}+00$ | -4.856731e-19 | -1.505893e-06 | $0.000000 \mathrm{e}+00$ |
| > | [10,] | -9.394819e-08 | $0.000000 \mathrm{e}+00$ | -1.505893e-06 | -2.793308e-08 | $0.000000 \mathrm{e}+00$ |
|  | [11,] | $0.000000 \mathrm{e}+00$ | -1.493589e-06 | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | -1.358213e-07 |
| > | [12,] | -2.210831e-07 | -4.324237e-06 | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | -2.513977e-07 |
|  | [13,] | -4.866713e-05 | -3.674845e-03 | -4.293275e-03 | -2.411530e-06 | $0.000000 \mathrm{e}+00$ |
|  | [14,] | -6.884346e-07 | $0.000000 \mathrm{e}+00$ | -7.001931e-06 | -8.409706e-08 | $0.000000 \mathrm{e}+00$ |
|  | [15,] | $0.000000 \mathrm{e}+00$ | $1.935813 \mathrm{e}-04$ | $0.000000 \mathrm{e}+00$ | $0.000000 \mathrm{e}+00$ | -4.828747e-07 |

```
> [16,] -1.218213e-05 -6.056160e-03 
> [,12] [,13] [,14] [,15] [,16]
> [1,] 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00
> [2,] 0.000000e+00 -1.616056e-04 -6.279654e-06 0.000000e+00 3.390146e-03
> [3,] -3.974459e-06 -1.817145e-04 -1.196694e-06 0.000000e+00 -1.377719e-04
[4,] -1.166898e-05 1.234967e-02 -5.886149e-07 -1.278891e-04 -6.774324e-04
[5,] 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00
[6,] -4.319227e-07 3.455946e-05 -3.627865e-07 0.000000e+00 2.676787e-04
[7,] -2.210831e-07 -4.866713e-05 -6.884346e-07 0.000000e+00 -1.218213e-05
> [8,] -4.324237e-06 -3.674845e-03 0.000000e+00 1.935813e-04 -6.056160e-03
[9,] 0.000000e+00 -4.293275e-03 -7.001931e-06 0.000000e+00 0.000000e+00
[10,] 0.000000e+00 -2.411530e-06 -8.409706e-08 0.000000e+00 0.000000e+00
[11,] -2.513977e-07 0.000000e+00 0.000000e+00 -4.828747e-07 -6.948820e-06
[12,] -7.976940e-07 -2.048541e-05 0.000000e+00 -3.182625e-07 -2.577829e-05
[13,] -2.048541e-05 8.366839e-18 -2.821753e-05 0.000000e+00 8.391062e-03
[14,] 0.000000e+00 -2.821753e-05 -4.627729e-07 0.000000e+00 -2.789972e-06
[15,] -3.182625e-07 0.000000e+00 0.000000e+00 0.000000e+00 -3.145326e-05
[16,] -2.577829e-05 8.391062e-03 -2.789972e-06 -3.145326e-05 3.290101e-18
>
>
> $r_values_m
> [1] -0.2773834410 0.0787122092 0.0470845080 -0.1625580814 -0.1294995080
[6] 0.1555179053-0.0001870075 -0.1225349871 0.0406509428
>
> $r_value_ref
> [1] 0.1061762
>
> $ahstages
> stage_id stage_id stage original_size original_size_b original_size_c min_age
> 1 1 1 1 N Sdl 
> 2 2 2 Veg NA
> 3 3 3 SmFlo NA NA NA
>4 4 4 LFlo 3 NA NA N
> max_age repstatus obsstatus propstatus immstatus matstatus entrystage
\begin{tabular}{llllllll}
\(>\) & 1 & NA & 0 & 1 & 0 & 1 & 0 \\
\(>\) & 2 & NA & 0 & 1 & 0 & 0 & 1 \\
\(>\) & 3 & NA & 1 & 1 & 0 & 0 & 1 \\
\(>\) & 4 & NA & 1 & 1 & 0 & 0 & 1
\end{tabular}
> indataset binhalfwidth_raw sizebin_min sizebin_max sizebin_center
\begin{tabular}{llllll}
\(>1\) & 1 & 0.5 & 0.5 & 1.5 & 1
\end{tabular}
\begin{tabular}{llllll}
\(>2\) & 1 & 0.5 & 0.5 & 1.5 & 1
\end{tabular}
\begin{tabular}{llllll}
\(>3\) & 1 & 0.5 & 1.5 & 2.5 & 2
\end{tabular}
\begin{tabular}{llllll}
\(>\) & 1 & 0.5 & 2.5 & 3.5 & 3
\end{tabular}
\begin{tabular}{rrrrr}
\(>\) & sizebin_width & binhalfwidthb_raw & sizebinb_min & sizebinb_max \\
\(>\) & sizebinb_center \\
\(>\) & 1 & 1 & \(N A\) & NA
\end{tabular}
> sizebinb_width binhalfwidthc_raw sizebinc_min sizebinc_max sizebinc_center
\begin{tabular}{llllll}
\(>\) & 1 & NA & NA & NA & NA \\
\(>2\) & NA & NA & NA & NA \\
\(>3\) & NA & NA & NA & NA & NA \\
& & & NA
\end{tabular}
```



The output here is longer than in the previous cases. Particularly, we have four sets of contribution matrices. The first set, in list cont_mean, tracks the contributions of shifts in the mean matrices, as before. The next three sets, cont_elas, cont_cv, and cont_corr, also referred to as the stochastic contribution matrices, track the contributions of shifts in matrix element elasticities, matrix element variation (assessed as the coefficient of variation), and matrix element correlations. Contribution matrices of these latter three types are more difficult to interpret because they are composed of elements that relate not simply to the previous elements, but to pairs of elements, leading to contribution matrices with squared dimensions.

Let's now look at how to interpret these matrices.

### 11.4.1.4 Interpreting LTRE contributions

lefkoLTRE objects are large and can take a great deal of effort to look over and understand. Therefore, we will show three approaches to assessing these objects, using an approach similar to that used to assess elasticities. These methods can be used to assess patterns in all nine populations, but for brevity we will focus only on the first population here. First, we will identify the elements most strongly impacting the population growth rate in each case. These are the elements with the highest absolute value in the contribution matrices. Because we are interested in their signs, we will look for the minimum and maximum values. We can do this for each patch, but here we do this for just the first patch (population C). Note that we will look at the cont_mean lists in all three LTRE types, as well as the cont_sd list in the sLTRE. We will not include the three stochastic contribution matrices from the SNA-LTRE except in the contribution sums for now, as they are more complicated to deal with.

```
p_c <- 1
# Highest (i.e most positive) deterministic LTRE contribution:
max(trialltre_det$cont_mean[[p_c]])
> [1] 0.02099388
# Highest deterministic LTRE contribution is associated with element:
which(trialltre_det$cont_mean[[p_c]] == max(trialltre_det$cont_mean[[p_c]]))
> [1] 3
# Lowest (i.e. most negative) deterministic LTRE contribution:
min(trialltre_det$cont_mean[[p_c]])
> [1] -0.2786045
# Lowest deterministic LTRE contribution is associated with element:
which(trialltre_det$cont_mean[[p_c]] == min(trialltre_det$cont_mean[[p_c]]))
> [1] 13
# Highest stochastic mean LTRE contribution:
max(trialltre_sto$cont_mean[[p_c]])
> [1] 0.0179472
# Highest stochastic mean LTRE contribution is associated with element:
which(trialltre_sto$cont_mean[[p_c]] == max(trialltre_sto$cont_mean[[p_c]]))
> [1] 3
# Lowest stochastic mean LTRE contribution:
min(trialltre_sto$cont_mean[[p_c]])
> [1] -0.3709379
# Lowest stochastic mean LTRE contribution is associated with element:
which(trialltre_sto$cont_mean[[p_c]] == min(trialltre_sto$cont_mean[[p_c]]))
> [1] 13
# Highest stochastic SD LTRE contribution:
max(trialltre_sto$cont_sd[[p_c]])
> [1] 0.009858186
# Highest stochastic SD LTRE contribution is associated with element:
which(trialltre_sto$cont_sd[[p_c]] == max(trialltre_sto$cont_sd[[p_c]]))
> [1] 13
# Lowest stochastic SD LTRE contribution:
min(trialltre_sto$cont_sd[[p_c]])
> [1] -0.01256449
# Lowest stochastic SD LTRE contribution is associated with element:
which(trialltre_sto$cont_sd[[p_c]] == min(trialltre_sto$cont_sd[[p_c]]))
> [1] 16
# Highest small noise approx mean LTRE contribution:
max(trialltre_sna$cont_mean[[p_c]])
> [1] 0.01698357
# Highest small noise approx mean LTRE contribution is associated with element:
which(trialltre_sna$cont_mean[[p_c]] == max(trialltre_sna$cont_mean[[p_c]]))
> [1] 3
# Lowest small noise approx mean LTRE contribution:
min(trialltre_sna$cont_mean[[p_c]])
> [1] -0.3440408
# Lowest small noise approx mean LTRE contribution is associated with element:
```

```
which(trialltre_sna$cont_mean[[p_c]] == min(trialltre_sna$cont_mean[[p_c]]))
> [1] 13
# Total positive deterministic LTRE contributions:
sum(trialltre_det$cont_mean[[p_c]][which(trialltre_det$cont_mean[[p_c]] > 0)])
> [1] 0.06338566
# Total negative deterministic LTRE contributions:
sum(trialltre_det$cont_mean[[p_c]][which(trialltre_det$cont_mean[[p_c]] < 0)])
> [1] -0.4079573
# Total positive stochastic mean LTRE contributions:
sum(trialltre_sto$cont_mean[[p_c]][which(trialltre_sto$cont_mean[[p_c]] > 0)])
> [1] 0.04472458
# Total negative stochastic mean LTRE contributions:
sum(trialltre_sto$cont_mean[[p_c]][which(trialltre_sto$cont_mean[[p_c]] < 0)])
> [1] -0.5097578
# Total positive stochastic SD LTRE contributions:
sum(trialltre_sto$cont_sd[[p_c]][which(trialltre_sto$cont_sd[[p_c]] > 0)])
> [1] 0.01060831
# Total negative stochastic SD LTRE contributions:
sum(trialltre_sto$cont_sd[[p_c]][which(trialltre_sto$cont_sd[[p_c]] < 0)])
> [1] -0.02347767
# Total positive small noise approx mean LTRE contributions:
sum(trialltre_sna$cont_mean[[p_c]][which(trialltre_sna$cont_mean[[p_c]] > 0)])
> [1] 0.04009463
# Total negative small noise approx mean LTRE contributions:
sum(trialltre_sna$cont_mean[[p_c]][which(trialltre_sna$cont_mean[[p_c]] < 0)])
> [1] -0.4758159
# Total positive small noise approx elasticity LTRE contributions:
sum(trialltre_sna$cont_elas[[p_c]][which(trialltre_sna$cont_elas[[p_c]] > 0)])
> [1] 0.02937135
# Total negative small noise approx elasticity LTRE contributions:
sum(trialltre_sna$cont_elas[[p_c]][which(trialltre_sna$cont_elas[[p_c]] < 0)])
> [1] -0.01448451
# Total positive small noise approx CV LTRE contributions:
sum(trialltre_sna$cont_cv[[p_c]][which(trialltre_sna$cont_cv[[p_c]] > 0)])
> [1] 0.1238029
# Total negative small noise approx CV LTRE contributions:
sum(trialltre_sna$cont_cv[[p_c]][which(trialltre_sna$cont_cv[[p_c]] < 0)])
> [1] -1.059949e-05
# Total positive small noise approx correlation LTRE contributions:
sum(trialltre_sna$cont_corr[[p_c]][which(trialltre_sna$cont_corr[[p_c]] > 0)])
> [1] 0.0105883
# Total negative small noise approx correlation LTRE contributions:
sum(trialltre_sna$cont_corr[[p_c]][which(trialltre_sna$cont_corr[[p_c]] < 0)])
> [1] -0.05488722
```

The output for the deterministic LTRE shows that element 3, which is the growth transition from seedling to small flowering adult (column 1, row 3), has the most positive influence. The strongest influence, however, is negative, and is associated with element 13 (column 4, row 1), which is the fecundity transition of large flowering adults. The same pattern holds for the stochastic and small noise
approximation LTREs, where we see the strongest influence of shifts in the mean value of element 13, and this influence is negative. The strongest positive contribution is associated with the mean value of element 3. Variability in elements also contributes to shifts in $\log \lambda$, though less so than shifts in mean elements. The strongest positive contribution in the stochastic LTRE is from variation in element 13 , the fecundity transition from large flowering adult to seedling (column 4 , row 1 ), while the most negative contribution is from stasis as a large flowering adult (row and column 4). A comparison of summed LTRE elements shows that negative contributions of mean elements were most influential in the stochastic and small noise approximation cases. The stochastic LTRE suggested little influence of shifts in variability, while the small noise approximation LTRE suggested little influence of shifts in elasticity, but a stronger, positive contribution of shifts in variability, and a relatively important negative impact of shifts in correlations.

Let's also take a look at the next patch.

```
p_c <- 2
# Highest (i.e most positive) deterministic LTRE contribution:
max(trialltre_det$cont_mean[[p_c]])
> [1] 0.1139692
# Highest deterministic LTRE contribution is associated with element:
which(trialltre_det$cont_mean[[p_c]] == max(trialltre_det$cont_mean[[p_c]]))
> [1] 13
# Lowest (i.e. most negative) deterministic LTRE contribution:
min(trialltre_det$cont_mean[[p_c]])
> [1] -0.08143273
# Lowest deterministic LTRE contribution is associated with element:
which(trialltre_det$cont_mean[[p_c]] == min(trialltre_det$cont_mean[[p_c]]))
> [1] 16
# Highest stochastic mean LTRE contribution:
max(trialltre_sto$cont_mean[[p_c]])
> [1] 0.1057347
# Highest stochastic mean LTRE contribution is associated with element:
which(trialltre_sto$cont_mean[[p_c]] == max(trialltre_sto$cont_mean[[p_c]]))
> [1] 13
# Lowest stochastic mean LTRE contribution:
min(trialltre_sto$cont_mean[[p_c]])
> [1] -0.1192655
# Lowest stochastic mean LTRE contribution is associated with element:
which(trialltre_sto$cont_mean[[p_c]] == min(trialltre_sto$cont_mean[[p_c]]))
> [1] 16
# Highest stochastic SD LTRE contribution:
max(trialltre_sto$cont_sd[[p_c]])
> [1] 0.003725215
# Highest stochastic SD LTRE contribution is associated with element:
which(trialltre_sto$cont_sd[[p_c]] == max(trialltre_sto$cont_sd[[p_c]]))
> [1] 16
# Lowest stochastic SD LTRE contribution:
min(trialltre_sto$cont_sd[[p_c]])
> [1] -0.02849515
# Lowest stochastic SD LTRE contribution is associated with element:
which(trialltre_sto$cont_sd[[p_c]] == min(trialltre_sto$cont_sd[[p_c]]))
```

```
> [1] 13
# Highest small noise approx mean LTRE contribution:
max(trialltre_sna$cont_mean[[p_c]])
> [1] 0.09806777
# Highest small noise approx mean LTRE contribution is associated with element:
which(trialltre_sna$cont_mean[[p_c]] == max(trialltre_sna$cont_mean[[p_c]]))
> [1] 13
# Lowest small noise approx mean LTRE contribution:
min(trialltre_sna$cont_mean[[p_c]])
> [1] -0.1138892
# Lowest small noise approx mean LTRE contribution is associated with element:
which(trialltre_sna$cont_mean[[p_c]] == min(trialltre_sna$cont_mean[[p_c]]))
> [1] 16
# Total positive deterministic LTRE contributions:
sum(trialltre_det$cont_mean[[p_c]][which(trialltre_det$cont_mean[[p_c]] > 0)])
> [1] 0.1312566
# Total negative deterministic LTRE contributions:
sum(trialltre_det$cont_mean[[p_c]][which(trialltre_det$cont_mean[[p_c]] < 0)])
> [1] -0.1615379
# Total positive stochastic mean LTRE contributions:
sum(trialltre_sto$cont_mean[[p_c]][which(trialltre_sto$cont_mean[[p_c]] > 0)])
> [1] 0.1328677
# Total negative stochastic mean LTRE contributions:
sum(trialltre_sto$cont_mean[[p_c]][which(trialltre_sto$cont_mean[[p_c]] < 0)])
> [1] -0.1883152
# Total positive stochastic SD LTRE contributions:
sum(trialltre_sto$cont_sd[[p_c]][which(trialltre_sto$cont_sd[[p_c]] > 0)])
> [1] 0.004307065
# Total negative stochastic SD LTRE contributions:
sum(trialltre_sto$cont_sd[[p_c]][which(trialltre_sto$cont_sd[[p_c]] < 0)])
> [1] -0.04302883
# Total positive small noise approx mean LTRE contributions:
sum(trialltre_sna$cont_mean[[p_c]][which(trialltre_sna$cont_mean[[p_c]] > 0)])
> [1] 0.1096399
# Total negative small noise approx mean LTRE contributions:
sum(trialltre_sna$cont_mean[[p_c]][which(trialltre_sna$cont_mean[[p_c]] < 0)])
> [1] -0.1803455
# Total positive small noise approx elasticity LTRE contributions:
sum(trialltre_sna$cont_elas[[p_c]][which(trialltre_sna$cont_elas[[p_c]] > 0)])
> [1] 0.02192178
# Total negative small noise approx elasticity LTRE contributions:
sum(trialltre_sna$cont_elas[[p_c]][which(trialltre_sna$cont_elas[[p_c]] < 0)])
> [1] -0.007874812
# Total positive small noise approx CV LTRE contributions:
sum(trialltre_sna$cont_cv[[p_c]][which(trialltre_sna$cont_cv[[p_c]] > 0)])
> [1] 0.1862284
# Total negative small noise approx CV LTRE contributions:
```

```
sum(trialltre_sna$cont_cv[[p_c]][which(trialltre_sna$cont_cv[[p_c]] < 0)])
> [1] -0.0001686659
# Total positive small noise approx correlation LTRE contributions:
sum(trialltre_sna$cont_corr[[p_c]][which(trialltre_sna$cont_corr[[p_c]] > 0)])
> [1] 0.01403254
# Total negative small noise approx correlation LTRE contributions:
sum(trialltre_sna$cont_corr[[p_c]][which(trialltre_sna$cont_corr[[p_c]] < 0)])
> [1] -0.05359317
```

We see some similarities and some differences. Particularly, element 3 is no longer among the most influential elements. However, elements 13 and 16 are strongly influential, suggesting a strong role to the fecundity of large adults and stasis within the large adult stage. In the small noise approximation case, where as the correlation impacts were the strongest next to shifts in the means in the case of the first patch, here we see that correlations have the second strongest impact.

Next, let's will identify which stages exerted the strongest impact on the population growth rate. Let's focus on the first patch again. Because of difficulty in interpreting the stochatic SNA-LTRE matrices by single stages, we will exclude those contributions from this analysis.

```
p_c <- 1
ltre_pos <- trialltre_det$cont_mean[[p_c]]
ltre_neg <- trialltre_det$cont_mean[[p_c]]
ltre_pos[which(ltre_pos < 0)] <- 0
ltre_neg[which(ltre_neg > 0)] <- 0
sltre_meanpos <- trialltre_sto$cont_mean[[p_c]]
sltre_meanneg <- trialltre_sto$cont_mean[[p_c]]
sltre_meanpos[which(sltre_meanpos < 0)] <- 0
sltre_meanneg[which(sltre_meanneg > 0)] <- 0
sltre_sdpos <- trialltre_sto$cont_sd[[p_c]]
sltre_sdneg <- trialltre_sto$cont_sd[[p_c]]
sltre_sdpos[which(sltre_sdpos < 0)] <- 0
sltre_sdneg[which(sltre_sdneg > 0)] <- 0
sna_meanpos <- trialltre_sna$cont_mean[[p_c]]
sna_meanneg <- trialltre_sna$cont_mean[[p_c]]
sna_meanpos[which(sna_meanpos < 0)] <- 0
sna_meanneg[which(sna_meanneg > 0)] <- 0
ltresums_pos <- cbind(colSums(ltre_pos), colSums(sltre_meanpos),
    colSums(sltre_sdpos), colSums(sna_meanpos))
ltresums_neg <- cbind(colSums(ltre_neg), colSums(sltre_meanneg),
    colSums(sltre_sdneg), colSums(sna_meanneg))
ltre_as_names <- trialltre_det$ahstages$stage
barplot(t(ltresums_pos), beside = T, col = c("black", "grey", "red", "white"),
    ylim = c(-0.50, 0.10))
barplot(t(ltresums_neg), beside = T, col = c("black", "grey", "red", "white"),
    add = TRUE)
abline(0, 0, lty= 3)
```

```
text(cex=1, y = -0.57, x = seq(from = 2, to = 4.98*length(ltre_as_names),
    by = 5), ltre_as_names, xpd=TRUE, srt=45)
legend("bottomleft", fill = c("black", "grey", "red", "white"),
    legend = c("deterministic", "stochastic mean", "stochastic SD", "SNA mean"),
    bty = "n")
```



Figure 11.3: LTRE contributions by stage for population 1
The output above shows that large flowering adults exerted the strongest influence on both $\lambda$ and $\log \lambda$, with the latter influence being through the impact of shifts in the mean. This impact is overwhelmingly negative. The next largest impact comes from seedlings in the deterministic case, and from small flowering adults in the stochastic case, in both cases the influence being negative on the whole.

Finally, we will assess what transition types exert the greatest impact on population growth rate.

```
det_ltre_summary <- summary(trialltre_det)
sto_ltre_summary <- summary(trialltre_sto)
sna_ltre_summary <- summary(trialltre_sna)
ltresums_tpos <- cbind(det_ltre_summary$ahist_mean$matrix1_pos,
    sto_ltre_summary$ahist_mean$matrix1_pos,
    sto_ltre_summary$ahist_sd$matrix1_pos,
    sna_ltre_summary$ahist_mean$matrix1_pos)
ltresums_tneg <- cbind(det_ltre_summary$ahist_mean$matrix1_neg,
    sto_ltre_summary$ahist_mean$matrix1_neg,
    sto_ltre_summary$ahist_sd$matrix1_neg,
    sna_ltre_summary$ahist_mean$matrix1_neg)
barplot(t(ltresums_tpos), beside = T, col = c("black", "grey", "red", "white"),
    ylim = c(-0.55, 0.10))
```

```
barplot(t(ltresums_tneg), beside = T, col = c("black", "grey", "red", "white"),
    add = TRUE)
abline(0, 0, lty = 3)
text(cex=0.85, y = -0.64, x = seq(from = 2, to = 4.98*length(det_ltre_summary$ahist_mean$category),
        by = 5), det_ltre_summary$ahist_mean$category, xpd=TRUE, srt=45)
legend("bottomleft", fill = c("black", "grey", "red", "white"),
    legend = c("deterministic", "stochastic mean", "stochastic SD", "SNA mean"),
    bty = "n")
```

        (
    





Figure 11.4: LTRE contributions by transition type
The overall greatest impact on the population growth rate is from fecundity transitions, which have generally had a negative impact. Clearly temporal variation has strong effects here that deserve to be assessed properly. Note that these impacts are relative to the grand mean.

### 11.5 Points to remember

1. Life table response experiments (LTREs) allow the demographic impacts of specific factors to be tested, provided that the factors yield matrices that can be compared against some control. These analyses generally show the impacts of such factors on the deterministic or stochastic population growth growth rate via contributions from shifts in matrix elements.
2. Stochastic life table response experiments (sLTREs) assess the impacts of factors on the stochastic population growth rate via shifts in both the temporal mean and the temporal standard deviation in matrix elements.
3. LTRE contributions may be summed within matrices to assess the impacts of specific stages or transitions as they respond to some treatment.

## Chapter 12

# Further Issues I: Importing Matrices and MPMs 

"We all reinvent our pasts."

- John le Carré

Package lefko3 includes powerful functions to create MPMs of all major kinds. However, suppose that you already have matrices and wish to use analyze them with lefko3. How can these matrices be imported properly? Our package includes functions that allow users to build lefkoMat objects with already existing matrices. Once these lefkoMat objects are built, they may be applied to the various functions found in the package and analyzed.

To start, we will need to get whatever matrices we have into R itself. This may or may not be easy, depending on what format the matrices are in. Users will find it easiest to import matrices saved in standard matrix format and in lists within $R$ object files, such as .Rda or .Rdata files. These files may be loaded into memory with the load() function. Other users may be working in R with some sort of MPM database, like COMPADRE (Salguero-Gómez et al., 2015). In other cases, users may need to import from a separate file, such as a comma-separated value (.csv) file. In this case, one approach that usually works is to use the scan() function together with matrix(), for example as matrix_a <- matrix (scan("myfile.txt"), nrow = 10). This function call will import a single matrix in a standard tab-delimited text file as a vector, and then use it to create a matrix with ten rows (this assumes that the right number of elements exists to import into a ten-row matrix).

Regardless of how you accomplish this step, the goal is to take your matrix and turn it into a matrix class object in $R$. If a matrix or series of matrices is not too large, then perhaps the simplest way to accomplish this is to enter the matrix manually.

### 12.1 Creating a new lefkoMat object from imported matrices

Let's take an example using a published set of matrices. Here, we will recreate our data object anthyllis (section 1.8.3). These matrices are from Davison et al. (2010), and they report stochastic contributions made by differences in vital rate means and variances among nine natural populations of the perennial herb Anthyllis vulneraria from calcareous grasslands in Belgium. It is a short-lived, rosette-forming legume with a complex life cycle including stasis and retrogression between four stages but no seedbank (seedlings, juveniles, small adults and large adults; figure 12.1).

Our goal in this exercise will be to import the published MPMs available for these nine populations of Anthyllis vulneraria, and to create a lefkoMat object for further study.


Figure 12.1: Life history model of Anthyllis vulneraria. Solid arrows indicate survival transitions while dashed arrows indicate fecundity transitions.

The most important part of importing matrices into lefko3 is also to import the underlying life history model. This will require reading over whatever material describes the analyses, whether a published research paper, an unpublished thesis, or another source, and carefully listing the stages, their descriptions, and their relationships as given in survival transitions and fecundity rates. In the case of our Anthyllis analysis, this means developing the life history model in the figure above, and then creating a stageframe to match it.

We do not have the original demographic dataset that produced the published matrices, and so we do not really know the exact sizes used to classify stages in this analysis. However, it turns out that we do not need to know the exact sizes of plants matching the small and large adult stages, because the size data itself will not really be used in any capacity in our own analyses. So, we will use proxy values for size in the stageframe. These proxy values need to be unique and non-negative, and they need non-overlapping bins usable as size classes defining each stage. However, since we are not analyzing size itself here, they do not need any further basis in reality. Other characteristics must be exact and realistic to make sure that the analyses work properly, including all other stage descriptions such as reproductive status, propagule status, and observation status.

```
sizevector <- c(1, 1, 2, 3) # These sizes are not from the original paper
stagevector <- c("Sdl", "Veg", "SmFlo", "LFlo")
repvector <- c(0, 0, 1, 1)
obsvector <- c(1, 1, 1, 1)
matvector <- c(0, 1, 1, 1)
immvector <- c(1, 0, 0, 0)
propvector <- c(0, 0, 0, 0)
indataset <- c(1, 1, 1, 1)
binvec <- c(0.5, 0.5, 0.5, 0.5)
comments <- c("Seedling", "Vegetative adult", "Small flowering",
```

```
    "Large flowering")
```


sizebin_center sizebin_width binhalfwidthb_raw sizebinb_min sizebinb_max
$\begin{array}{llllll}1 & 1 & 1 & N A & N A & \text { NA } \\ 2 & 1 & 1 & N A & N A & \text { NA } \\ 3 & 2 & 1 & N A & N A & \text { NA } \\ 4 & 3 & 1 & N A & N A & \text { NA }\end{array}$
sizebinb_center sizebinb_width binhalfwidthc_raw sizebinc_min sizebinc_max
1 NA NA NA NA NA
> NA NA NA NA NA
$\begin{array}{lllll}>3 & N A & \text { NA } & \text { NA } & \text { NA }\end{array}$
4 NA NA NA NA NA
> sizebinc_center sizebinc_width group comments
1 NA NA 0 Seedling
$>2$ NA NA 0 Vegetative adult
$>3$ NA NA 0 small flowering
> 4 NA NA 0 Large flowering

Next we can enter the matrices, which were published in Davison et al. (2010). We will enter these matrices as standard R matrix class objects. All matrices are square with four columns. I personally find it easier to enter matrices by row when copying from text, and so I have chosen to do so here and have indicated so in the matrix() function using byrow $=$ TRUE (the default is to fill by column). Users familiar and comfortable with Matlab will likely be quite comfortable with filling matrices by row. Here is the first matrix, covering 2003 to 2004 for population C.

```
XC3 <- matrix(c(0, 0, 1.74, 1.74,
    0.208333333, 0, 0, 0.057142857,
    0.041666667, 0.076923077, 0, 0,
    0.083333333, 0.076923077, 0.066666667, 0.028571429), 4, 4, byrow = TRUE)
XC3
> [,1] [,2] [,3] [,4]
> [1,] 0.0000000000.00000000 1.74000000 1.74000000
> [2,] 0.20833333 0.00000000 0.00000000 0.05714286
> [3,] 0.04166667 0.07692308 0.00000000 0.00000000
> [4,] 0.08333333 0.07692308 0.06666667 0.02857143
```

This is an A matrix, meaning that it includes all survival-transitions and fecundity for the population
as a whole. The corresponding $U$ and $F$ matrices were not provided in that paper, although it is most likely that the elements valued at 1.74 in the top right-hand corner are only composed of fecundity values while the rest of the matrix is only composed of survival transitions (this might not be the case if clonal reproduction were possible). The order of rows and columns corresponds to the order of stages in the stageframe anthframe. Let's now load the remaining matrices.

```
# POPN C 2004-2005
XC4 <- matrix(c(0, 0, 0.3, 0.6,
    0.32183908, 0.142857143, 0, 0,
    0.16091954, 0.285714286, 0, 0,
    0.252873563, 0.285714286, 0.5, 0.6), 4, 4, byrow = TRUE)
# POPN C 2005-2006
XC5 <- matrix(c(0, 0, 0.50625, 0.675,
    0, 0, 0, 0.035714286,
    0.1, 0.068965517, 0.0625, 0.107142857,
    0.3, 0.137931034, 0, 0.071428571), 4, 4, byrow = TRUE)
# POPN E 2003-2004
XE3 <- matrix(c(0, 0, 2.44, 6.569230769,
    0.196428571, 0, 0, 0,
    0.125, 0.5, 0, 0,
    0.160714286, 0.5, 0.133333333, 0.076923077), 4, 4, byrow = TRUE)
# POPN E 2004-2005
XE4 <- matrix(c(0, 0, 0.45, 0.646153846,
    0.06557377, 0.090909091, 0.125, 0,
    0.032786885, 0, 0.125, 0.076923077,
    0.049180328, 0, 0.125, 0.230769231), 4, 4, byrow = TRUE)
# POPN E 2005-2006
XE5 <- matrix(c(0, 0, 2.85, 3.99,
    0.083333333, 0, 0, 0,
    0, 0, 0, 0,
    0.416666667, 0.1, 0, 0.1), 4, 4, byrow = TRUE)
# POPN F 2003-2004
XF3 <- matrix(c(0, 0, 1.815, 7.058333333,
    0.075949367, 0, 0.05, 0.083333333,
    0.139240506, 0, 0, 0.25,
    0.075949367, 0, 0, 0.083333333), 4, 4, byrow = TRUE)
# POPN F 2004-2005
XF4 <- matrix(c(0, 0, 1.233333333, 7.4,
    0.223880597, 0, 0.1111111111, 0.142857143,
    0.134328358, 0.272727273, 0.166666667, 0.142857143,
    0.119402985, 0.363636364, 0.055555556, 0.142857143), 4, 4, byrow = TRUE)
# POPN F 2005-2006
XF5 <- matrix(c(0, 0, 1.06, 3.372727273,
    0.073170732, 0.025, 0.033333333, 0,
    0.036585366, 0.15, 0.1, 0.136363636,
```

```
    0.06097561, 0.225, 0.166666667, 0.272727273), 4, 4, byrow = TRUE)
# POPN G 2003-2004
XG3 <- matrix(c(0, 0, 0.245454545, 2.1,
    0, 0, 0.045454545, 0,
    0.125, 0, 0.090909091, 0,
    0.125, 0, 0.090909091, 0.333333333), 4, 4, byrow = TRUE)
# POPN G 2004-2005
XG4 <- matrix(c(0, 0, 1.1, 1.54,
    0.111111111, 0, 0, 0,
    0, 0, 0, 0,
    0.1111111111, 0, 0, 0), 4, 4, byrow = TRUE)
# POPN G 2005-2006
XG5 <- matrix(c(0, 0, 0, 1.5,
    0, 0, 0, 0,
    0.090909091, 0, 0, 0,
    0.545454545, 0.5, 0, 0.5), 4, 4, byrow = TRUE)
# POPN L 2003-2004
XL3 <- matrix(c(0, 0, 1.785365854, 1.856521739,
    0.128571429, 0, 0, 0.010869565,
    0.028571429, 0, 0, 0,
    0.014285714, 0, 0, 0.02173913), 4, 4, byrow = TRUE)
# POPN L 2004-2005
XL4 <- matrix(c(0, 0, 14.25, 16.625,
    0.131443299, 0.057142857, 0, 0.25,
    0.144329897, 0, 0, 0,
    0.092783505, 0.2, 0, 0.25), 4, 4, byrow = TRUE)
# POPN L 2005-2006
XL5 <- matrix(c(0, 0, 0.594642857, 1.765909091,
    0, 0, 0.017857143, 0,
    0.021052632, 0.018518519, 0.035714286, 0.045454545,
    0.021052632, 0.018518519, 0.035714286, 0.068181818), 4, 4, byrow = TRUE)
# POPN O 2003-2004
XO3 <- matrix(c(0, 0, 11.5, 2.775862069,
    0.6, 0.285714286, 0.333333333, 0.24137931,
    0.04, 0.142857143, 0, 0,
    0.16, 0.285714286, 0, 0.172413793), 4, 4, byrow = TRUE)
# POPN O 2004-2005
X04 <- matrix(c(0, 0, 3.78, 1.225,
    0.28358209, 0.171052632, 0, 0.166666667,
    0.084577114, 0.026315789, 0, 0.055555556,
    0.139303483, 0.447368421, 0, 0.305555556), 4, 4, byrow = TRUE)
# POPN O 2005-2006
```

```
XO5 <- matrix(c(0, 0, 1.542857143, 1.035616438,
    0.126984127, 0.105263158, 0.047619048, 0.054794521,
    0.095238095, 0.157894737, 0.19047619, 0.082191781,
    0.111111111, 0.223684211, 0, 0.356164384), 4, 4, byrow = TRUE)
# POPN Q 2003-2004
XQ3 <- matrix(c(0, 0, 0.15, 0.175,
    0, 0, 0, 0,
    0, 0, 0, 0,
    1, 0, 0, 0), 4, 4, byrow = TRUE)
# POPN Q 2004-2005
XQ4 <- matrix(c(0, 0, 0, 0.25,
    0, 0, 0, 0,
    0, 0, 0, 0,
    1, 0.666666667, 0, 1), 4, 4, byrow = TRUE)
# POPN Q 2005-2006
XQ5 <- matrix(c(0, 0, 0, 1.428571429,
    0, 0, 0, 0.142857143,
    0.25, 0, 0, 0,
    0.25, 0, 0, 0.571428571), 4, 4, byrow = TRUE)
# POPN R 2003-2004
XR3 <- matrix(c(0, 0, 0.7, 0.6125,
    0.25, 0, 0, 0.125,
    0, 0, 0, 0,
    0.25, 0.166666667, 0, 0.25), 4, 4, byrow = TRUE)
# POPN R 2004-2005
XR4 <- matrix(c(0, 0, 0, 0.6,
    0.285714286, 0, 0, 0,
    0.285714286, 0.333333333, 0, 0,
    0.285714286, 0.333333333, 0, 1), 4, 4, byrow = TRUE)
# POPN R 2005-2006
XR5 <- matrix(c(0, 0, 0.7, 0.6125,
    0, 0, 0, 0,
    0, 0, 0, 0,
    0.333333333, 0, 0.333333333, 0.625), 4, 4, byrow = TRUE)
# POPN S 2003-2004
XS3 <- matrix(c(0, 0, 2.1, 0.816666667,
    0.166666667, 0, 0, 0,
    0, 0, 0, 0,
    0, 0, 0, 0.166666667), 4, 4, byrow = TRUE)
# POPN S 2004-2005
XS4 <- matrix(c(0, 0, 0, 7,
    0.333333333, 0.5, 0, 0,
    0, 0, 0, 0,
```

```
    0.333333333, 0, 0, 1), 4, 4, byrow = TRUE)
# POPN S 2005-2006
XS5 <- matrix(c(0, 0, 0, 1.4,
    0, 0, 0, 0,
    0, 0, 0, 0.2,
    0.111111111, 0.75, 0, 0.2), 4, 4, byrow = TRUE)
```

Our next step will be to incorporate these matrices into a single list of matrices. As a reminder, a list is an object in which each element can be of a different class and of different length. Let's build our list and then inspect it.

```
mats_list <- list(XC3, XC4, XC5, XE3, XE4, XE5, XF3, XF4, XF5, XG3, XG4, XG5,
    XL3, XL4, XL5, X03, X04, X05, XQ3, XQ4, XQ5, XR3, XR4, XR5, XS3, XS4, XS5)
mats_list
> [[1]]
> [,1] [,2] [,3] [,4]
    [1,] 0.00000000 0.00000000 1.74000000 1.74000000
    [2,] 0.20833333 0.00000000 0.00000000 0.05714286
> [3,] 0.04166667 0.07692308 0.00000000 0.00000000
> [4,] 0.08333333 0.07692308 0.06666667 0.02857143
>
> [[2]]
> [,1] [,2] [,3] [,4]
    [1,] 0.0000000 0.0000000 0.3 0.6
```



```
> [3,] 0.1609195 0.2857143 0.0
> [4,] 0.2528736 0.2857143 0.5
>
> [[3]]
> [,1] [,2] [,3] [,4]
> [1,] 0.0 0.00000000 0.50625 0.67500000
> [2,] 0.0 0.00000000 0.00000 0.03571429
> [3,] 0.1 0.06896552 0.06250 0.10714286
> [4,] 0.3 0.13793103 0.00000 0.07142857
>
> [[4]]
> [,1] [,2] [,3] [,4]
> [1,] 0.0000000 0.0 2.4400000 6.56923077
> [2,] 0.1964286 0.0 0.0000000 0.00000000
> [3,] 0.1250000 0.5 0.0000000 0.00000000
> [4,] 0.1607143 0.5 0.1333333 0.07692308
>
> [[5]]
> [,1] [,2] [,3] [,4]
> [1,] 0.00000000 0.00000000 0.450 0.64615385
> [2,] 0.06557377 0.09090909 0.125 0.00000000
> [3,] 0.03278689 0.00000000 0.125 0.07692308
> [4,] 0.04918033 0.00000000 0.125 0.23076923
>
> [[6]]
> [,1] [,2] [,3] [,4]
```

```
> [1,] 0.00000000 0.0 2.85 3.99
> [2,] 0.08333333 0.0 0.00 0.00
> [3,] 0.00000000 0.0 0.00 0.00
> [4,] 0.41666667 0.1 0.00 0.10
>
> [[7]]
> [,1] [,2] [,3] [,4]
> [1,] 0.00000000 0 1.815 7.05833333
> [2,] 0.07594937 0 0.050 0.08333333
> [3,] 0.13924051 0 0.000 0.25000000
> [4,] 0.07594937 0 0.000 0.08333333
>
> [[8]]
> [,1] [,2] [,3] [,4]
> [1,] 0.0000000 0.0000000 1.23333333 7.4000000
> [2,] 0.2238806 0.0000000 0.11111111 0.1428571
> [3,] 0.1343284 0.2727273 0.16666667 0.1428571
> [4,] 0.1194030 0.3636364 0.05555556 0.1428571
>
> [[9]]
> [,1] [,2] [,3] [,4]
> [1,] 0.00000000 0.000 1.06000000 3.3727273
> [2,] 0.07317073 0.025 0.03333333 0.0000000
> [3,] 0.03658537 0.150 0.10000000 0.1363636
> [4,] 0.06097561 0.225 0.16666667 0.2727273
>
> [[10]]
> [,1] [,2] [,3] [,4]
> [1,] 0.000 0 0.24545454 2.1000000
> [2,] 0.000 0 0.04545454 0.0000000
> [3,] 0.125 0 0.09090909 0.0000000
> [4,] 0.125 0 0.09090909 0.3333333
>
> [[11]]
> [,1] [,2] [,3] [,4]
> [1,] 0.0000000 0 1.1 1.54
> [2,] 0.11111111 0}00.00.0
> [3,] 0.0000000 0}00.00.0
> [4,] 0.1111111 0}00.00.0
>
> [[12]]
> [,1] [,2] [,3] [,4]
> [1,] 0.00000000 0.0 0 1.5
> [2,] 0.00000000 0.0 0}00.
> [3,] 0.09090909 0.0 0 0.0
> [4,] 0.54545455 0.5 0 0.5
>
> [[13]]
> [,1] [,2] [,3] [,4]
> [1,] 0.00000000 0 1.785366 1.85652174
> [2,] 0.12857143 0 0.000000 0.01086956
```

```
> [3,] 0.02857143 0 0.000000 0.00000000
> [4,] 0.01428571 0 0.000000 0.02173913
>
> [[14]]
> [,1] [,2] [,3] [,4]
> [1,] 0.00000000 0.00000000 14.25 16.625
> [2,] 0.13144330 0.05714286 0.00 0.250
> [3,] 0.14432990 0.00000000 0.00 0.000
> [4,] 0.09278351 0.20000000 0.00 0.250
>
> [[15]]
> [,1] [,2] [,3] [,4]
> [1,] 0.00000000 0.00000000 0.59464286 1.76590909
> [2,] 0.00000000 0.00000000 0.01785714 0.00000000
> [3,] 0.02105263 0.01851852 0.03571429 0.04545454
> [4,] 0.02105263 0.01851852 0.03571429 0.06818182
>
> [[16]]
> [,1] [,2] [,3] [,4]
> [1,] 0.00 0.0000000 11.5000000 2.7758621
> [2,] 0.60 0.2857143 0.3333333 0.2413793
> [3,] 0.04 0.1428571 0.0000000 0.0000000
> [4,] 0.16 0.2857143 0.0000000 0.1724138
>
> [[17]]
> [,1] [,2] [,3] [,4]
> [1,] 0.00000000 0.00000000 3.78 1.22500000
> [2,] 0.28358209 0.17105263 0.00 0.16666667
> [3,] 0.08457711 0.02631579 0.00 0.05555556
> [4,] 0.13930348 0.44736842 0.00 0.30555556
>
> [[18]]
> [,1] [,2] [,3] [,4]
> [1,] 0.00000000 0.0000000 1.54285714 1.03561644
> [2,] 0.12698413 0.1052632 0.04761905 0.05479452
> [3,] 0.09523809 0.1578947 0.19047619 0.08219178
> [4,] 0.11111111 0.2236842 0.00000000 0.35616438
>
> [[19]]
> [,1] [,2] [,3] [,4]
> [1,] 0 0 0.15 0.175
> [2,] 0 0 0.00 0.000
> [3,] 0 0 0.00 0.000
> [4,] 1 0 0.00 0.000
>
> [[20]]
> [,1] [,2] [,3] [,4]
> [1,] 0 0.0000000 00.25
> [2,] 0 0.0000000 0 0.00
> [3,] 0 0.0000000 0 0.00
> [4,] 1 0.6666667 0 1.00
```

```
>
> [[21]]
> [,1] [,2] [,3] [,4]
> [1,] 0.00 0 0 1.4285714
> [2,] 0.00 0 0 0.1428571
> [3,] 0.25 0 0 0.0000000
> [4,] 0.25 0 0 0.5714286
>
> [[22]]
> [,1] [,2] [,3] [,4]
> [1,] 0.00 0.0000000 0.7 0.6125
> [2,] 0.25 0.0000000 0.0 0.1250
> [3,] 0.00 0.0000000 0.0 0.0000
> [4,] 0.25 0.1666667 0.0 0.2500
>
> [[23]]
> [,1] [,2] [,3] [,4]
> [1,] 0.0000000 0.0000000 0 0.6
> [2,] 0.2857143 0.0000000 0 0.0
> [3,] 0.2857143 0.3333333 0 0.0
> [4,] 0.2857143 0.3333333 0 1.0
>
> [[24]]
> [,1] [,2] [,3] [,4]
> [1,] 0.0000000 0 0.7000000 0.6125
> [2,] 0.0000000 0 0.0000000 0.0000
> [3,] 0.0000000 0 0.0000000 0.0000
> [4,] 0.3333333 0 0.3333333 0.6250
>
> [[25]]
> [,1] [,2] [,3] [,4]
> [1,] 0.0000000 0 2.1 0.8166667
> [2,] 0.1666667 0}0.00.000000
> [3,] 0.0000000 0 0.0 0.0000000
> [4,] 0.0000000 0}00.00.166666
>
> [[26]]
> [,1] [,2] [,3] [,4]
> [1,] 0.0000000 0.0 0 7
> [2,] 0.3333333 0.5 0 0
> [3,] 0.0000000 0.0 0 0
> [4,] 0.3333333 0.0 0
>
> [[27]]
> [,1] [,2] [,3] [,4]
> [1,] 0.0000000 0.00 0 1.4
> [2,] 0.0000000 0.00 0}00.
> [3,] 0.0000000 0.00 0}00.
> [4,] 0.1111111 0.75 0}00.
```

We might now wish to check that the top-rightmost two elements are fecundity rates. One way of doing this is to replicate this list into another list, then change the two top-rightmost elements to
zero, and then check both the range of values in the final matrices and check the column sums. If we are correct, then both the ranges of values and the column sums will be between 0.0 and 1.0 . R provides a number of functions to make working with lists relatively easy, and chief among these are the lapply() and unlist() functions. Let's try this approach.

```
all_potential_Us <- lapply(mats_list, function(X) {
    X[1, 3] <- 0
    X[1,4] <- 0
    return(X)
})
summary(unlist(all_potential_Us))
> Min. 1st Qu. Median Mean 3rd Qu. Max.
> 0.00000 0.00000 0.00000 0.08003 0.10000 1.00000
all_colSums <- lapply(all_potential_Us, colSums)
summary(unlist(all_colSums))
> Min. 1st Qu. Median Mean 3rd Qu. Max.
> 0.00000 0.04803 0.29557 0.32012 0.50000 1.00000
```

We see that we are left with values that appear to be probabilities (i.e. all range from 0.0 to 1.0 ), so everything looks good.

Next we will incorporate all of these matrices into a lefkoMat object. We will create the lefkoMat object to hold these matrices by calling function create_1M() with the list object we created. We will also include metadata describing the order of populations (here treated as patches), and the order of monitoring occasions.

```
anth_lefkoMat <- create_lM(mats = mats_list, stageframe = anthframe,
    hstages = NA, historical = FALSE, poporder = 1,
    patchorder = c("C", "C", "C", "E", "E", "E", "F", "F", "F", "G", "G", "G",
        "L", "L", "L", "O", "O", "O", "Q", "Q", "Q", "R", "R", "R", "S", "S", "S"),
    yearorder = c(2003, 2004, 2005, 2003, 2004, 2005, 2003, 2004, 2005, 2003,
        2004, 2005, 2003, 2004, 2005, 2003, 2004, 2005, 2003, 2004, 2005, 2003,
        2004, 2005, 2003, 2004, 2005))
anth_lefkoMat
> $A
> $A[[1]]
> [,1] [,2] [,3] [,4]
> [1,] 0.00000000 0.00000000 1.74000000 1.74000000
> [2,] 0.20833333 0.00000000 0.00000000 0.05714286
> [3,] 0.04166667 0.07692308 0.00000000 0.00000000
> [4,] 0.08333333 0.07692308 0.06666667 0.02857143
>
> $A[[2]]
> [,1] [,2] [,3] [,4]
> [1,] 0.0000000 0.0000000 0.3 0.6
> [2,] 0.3218391
> [3,] 0.1609195 0.2857143 0.0 0.0
> [4,] 0.2528736 0.2857143 0.5 0.6
>
> $A[[3]]
> [,1] [,2] [,3] [,4]
```

```
> [1,] 0.0 0.00000000 0.50625 0.67500000
> [2,] 0.0 0.00000000 0.00000 0.03571429
> [3,] 0.1 0.06896552 0.06250 0.10714286
> [4,] 0.3 0.13793103 0.00000 0.07142857
>
> $A[[4]]
> [,1] [,2] [,3] [,4]
> [1,] 0.0000000 0.0 2.4400000 6.56923077
> [2,] 0.1964286 0.0 0.0000000 0.00000000
> [3,] 0.1250000 0.5 0.0000000 0.00000000
> [4,] 0.1607143 0.5 0.1333333 0.07692308
>
> $A[[5]]
> [,1] [,2] [,3] [,4]
> [1,] 0.00000000 0.00000000 0.450 0.64615385
> [2,] 0.06557377 0.09090909 0.125 0.00000000
> [3,] 0.03278689 0.00000000 0.125 0.07692308
> [4,] 0.04918033 0.00000000 0.125 0.23076923
>
> $A[[6]]
> [,1] [,2] [,3] [,4]
> [1,] 0.00000000 0.0 2.85 3.99
> [2,] 0.08333333 0.0 0.00 0.00
> [3,] 0.00000000 0.0 0.00 0.00
> [4,] 0.41666667}0.10.0000.1
>
> $A[[7]]
> [,1] [,2] [,3] [,4]
> [1,] 0.00000000 0 1.815 7.05833333
> [2,] 0.07594937 0 0.050 0.08333333
> [3,] 0.13924051 0 0.000 0.25000000
> [4,] 0.07594937 0 0.000 0.08333333
>
> $A[[8]]
> [,1] [,2] [,3] [,4]
> [1,] 0.0000000 0.0000000 1.23333333 7.4000000
> [2,] 0.2238806 0.0000000 0.11111111 0.1428571
> [3,] 0.1343284 0.2727273 0.16666667 0.1428571
> [4,] 0.1194030 0.3636364 0.05555556 0.1428571
>
> $A[[9]]
> [,1] [,2] [,3] [,4]
> [1,] 0.00000000 0.000 1.06000000 3.3727273
> [2,] 0.07317073 0.025 0.03333333 0.0000000
> [3,] 0.03658537 0.150 0.10000000 0.1363636
> [4,] 0.06097561 0.225 0.16666667 0.2727273
>
> $A[[10]]
> [,1] [,2] [,3] [,4]
> [1,] 0.000 0 0.24545454 2.1000000
> [2,] 0.000 0 0.04545454 0.0000000
```

```
> [3,] 0.125 0 0.09090909 0.0000000
> [4,] 0.125 0 0.09090909 0.3333333
>
> $A[[11]]
> [,1] [,2] [,3] [,4]
> [1,] 0.0000000 0
> [2,] 0.11111111 0 0.0 0.00
> [3,] 0.0000000 0 0.0 0.00
> [4,] 0.11111111 0 0.0 0.00
>
> $A[[12]]
> [,1] [,2] [,3] [,4]
> [1,] 0.00000000}00.0 0 1.5
> [2,] 0.00000000 0.0 0}00.
> [3,] 0.09090909 0.0 0}00.
> [4,] 0.54545455 0.5 0}0.
>
> $A[[13]]
> [,1] [,2] [,3] [,4]
> [1,] 0.00000000 0 1.785366 1.85652174
> [2,] 0.12857143 0 0.000000 0.01086956
> [3,] 0.02857143 0 0.000000 0.00000000
> [4,] 0.01428571 00.000000 0.02173913
>
> $A[[14]]
> [,1] [,2] [,3] [,4]
> [1,] 0.00000000 0.00000000 14.25 16.625
> [2,] 0.13144330}00.05714286 0.00 0.250
> [3,] 0.14432990 0.00000000 0.00 0.000
> [4,] 0.09278351 0.20000000 0.00 0.250
>
> $A[[15]]
> [,1] [,2] [,3] [,4]
> [1,] 0.00000000 0.00000000 0.59464286 1.76590909
> [2,] 0.00000000 0.00000000 0.01785714 0.00000000
> [3,] 0.02105263 0.01851852 0.03571429 0.04545454
> [4,] 0.02105263 0.01851852 0.03571429 0.06818182
>
> $A[[16]]
> [,1] [,2] [,3] [,4]
> [1,] 0.00 0.0000000 11.5000000 2.7758621
> [2,] 0.60 0.2857143 0.3333333 0.2413793
> [3,] 0.04 0.1428571 0.0000000 0.0000000
> [4,] 0.16 0.2857143 0.0000000 0.1724138
>
> $A[[17]]
> [,1] [,2] [,3] [,4]
> [1,] 0.00000000 0.00000000 3.78 1.22500000
> [2,] 0.28358209 0.17105263 0.00 0.16666667
> [3,] 0.08457711 0.02631579 0.00 0.05555556
> [4,] 0.13930348 0.44736842 0.00 0.30555556
```

```
>
$A[[18]]
> [,1] [,2] [,3] [,4]
> [1,] 0.00000000 0.0000000 1.54285714 1.03561644
> [2,] 0.12698413 0.1052632 0.04761905 0.05479452
> [3,] 0.09523809 0.1578947 0.19047619 0.08219178
> [4,] 0.11111111 0.2236842 0.00000000 0.35616438
>
> $A[[19]]
> [,1] [,2] [,3] [,4]
> [1,] 0 0 0.15 0.175
> [2,] 0 0 0.00 0.000
> [3,] 0 0 0.00 0.000
> [4,] 1 0 0.00 0.000
>
> $A[[20]]
> [,1] [,2] [,3] [,4]
> [1,] 0 0.0000000 0 0.25
> [2,] 0 0.0000000 0 0.00
> [3,] 0 0.0000000 0 0.00
> [4,] 1 0.6666667 0 1.00
>
> $A[[21]]
> [,1] [,2] [,3] [,4]
> [1,] 0.00 0 0 1.4285714
> [2,] 0.00 0 0 0.1428571
> [3,] 0.25 0 0 0.0000000
> [4,] 0.25 0 0 0.5714286
>
> $A[[22]]
    [,1] [,2] [,3] [,4]
> [1,] 0.00 0.0000000 0.7 0.6125
> [2,] 0.25 0.0000000 0.0 0.1250
> [3,] 0.00 0.0000000 0.0 0.0000
> [4,] 0.25 0.1666667 0.0}00.250
>
> $A[[23]]
> [,1] [,2] [,3] [,4]
> [1,] 0.0000000 0.0000000 0 0.6
> [2,] 0.2857143 0.0000000 0 0.0
> [3,] 0.2857143 0.3333333 0 0.0
> [4,] 0.2857143 0.3333333 0 1.0
>
> $A[[24]]
> [,1] [,2] [,3] [,4]
> [1,] 0.0000000 0 0.7000000 0.6125
> [2,] 0.0000000 0 0.0000000 0.0000
> [3,] 0.0000000 00.0000000 0.0000
> [4,] 0.3333333 0 0.3333333 0.6250
>
> $A[[25]]
```

```
> [,1] [,2] [,3] [,4]
> [1,] 0.0000000 0 2.1 0.8166667
> [2,] 0.1666667 0}0.00.000000
> [3,] 0.0000000 0 0.0 0.0000000
> [4,] 0.0000000 0 0.0 0.1666667
>
> $A[[26]]
> [,1] [,2] [,3] [,4]
> [1,] 0.0000000 0.0 0 7
> [2,] 0.3333333 0.5 0 0
> [3,] 0.0000000 0.0 0}
> [4,] 0.3333333
>
> $A[[27]]
> [,1] [,2] [,3] [,4]
> [1,] 0.0000000 0.00 0 1.4
> [2,] 0.0000000 0.00 0}00.
> [3,] 0.0000000 0.00 0}00.
> [4,] 0.11111111 0.75 0
>
>
> $U
$U[[1]]
> [,1] [,2] [,3] [,4]
> [1,] 0.00000000 0.00000000 0.00000000 0.000000000
> [2,] 0.20833333 0.00000000 0.00000000 0.05714286
> [3,] 0.04166667 0.07692308 0.00000000 0.00000000
> [4,] 0.08333333 0.07692308 0.06666667 0.02857143
>
> $U[[2]]
> [,1] [,2] [,3] [,4]
> [1,] 0.0000000 0.0000000 0.0 0.0
> [2,] 0.3218391 0.1428571 0.0
> [3,] 0.1609195 0.2857143 0.0
> [4,] 0.2528736 0.2857143 0.5 0.6
>
> $U[[3]]
> [,1] [,2] [,3] [,4]
> [1,] 0.0 0.00000000 0.0000 0.00000000
> [2,] 0.0 0.00000000 0.0000 0.03571429
> [3,] 0.1 0.06896552 0.0625 0.10714286
> [4,] 0.3 0.13793103 0.0000 0.07142857
>
> $U[[4]]
> [,1] [,2] [,3] [,4]
> [1,] 0.0000000 0.0 0.0000000 0.00000000
> [2,] 0.1964286 0.0 0.0000000 0.00000000
> [3,] 0.1250000 0.5 0.0000000 0.00000000
> [4,] 0.1607143 0.5 0.1333333 0.07692308
>
> $U[[5]]
```

```
> [,1] [,2] [,3] [,4]
> [1,] 0.00000000 0.00000000 0.000 0.00000000
> [2,] 0.06557377 0.09090909 0.125 0.00000000
> [3,] 0.03278689 0.00000000 0.125 0.07692308
> [4,] 0.04918033 0.00000000 0.125 0.23076923
>
> $U[[6]]
> [,1] [,2] [,3] [,4]
> [1,] 0.00000000 0.0 0 0.0
> [2,] 0.08333333 0.0 0 0.0
> [3,] 0.00000000 0.0 0 0.0
> [4,] 0.41666667
>
> $U[[7]]
> [,1] [,2] [,3] [,4]
> [1,] 0.00000000 0 0.00 0.00000000
> [2,] 0.07594937 0 0.05 0.08333333
> [3,] 0.13924051 0 0.00 0.25000000
> [4,] 0.07594937 0 0.00 0.08333333
>
> $U[[8]]
> [,1] [,2] [,3] [,4]
> [1,] 0.0000000 0.0000000 0.00000000 0.0000000
> [2,] 0.2238806 0.0000000 0.11111111 0.1428571
> [3,] 0.1343284 0.2727273 0.16666667 0.1428571
> [4,] 0.1194030 0.3636364 0.05555556 0.1428571
>
> $U[[9]]
> [,1] [,2] [,3] [,4]
> [1,] 0.00000000 0.000 0.00000000 0.0000000
> [2,] 0.07317073 0.025 0.03333333 0.0000000
> [3,] 0.03658537 0.150 0.10000000 0.1363636
> [4,] 0.06097561 0.225 0.16666667 0.2727273
>
> $U[[10]]
> [,1] [,2] [,3] [,4]
> [1,] 0.000 0 0.00000000 0.0000000
> [2,] 0.000 0 0.04545454 0.0000000
> [3,] 0.125 0 0.09090909 0.0000000
> [4,] 0.125 0 0.09090909 0.3333333
>
> $U[[11]]
> [,1] [,2] [,3] [,4]
> [1,] 0.0000000 0 0 0
> [2,] 0.1111111 0 0 0
> [3,] 0.0000000 0 0 0
> [4,] 0.1111111 0}0
>
> $U[[12]]
> [,1] [,2] [,3] [,4]
> [1,] 0.00000000 0.0 0 0.0
```

```
> [2,] 0.00000000 0.0 0}00.
> [3,] 0.09090909 0.0 0}00.
> [4,] 0.54545455 0.5 0}0.
>
> $U[[13]]
> [,1] [,2] [,3] [,4]
> [1,] 0.00000000 0 0 0.00000000
> [2,] 0.12857143 0 0 0.01086956
> [3,] 0.02857143 0 0 0.00000000
> [4,] 0.01428571 0 0 0.02173913
>
> $U[[14]]
> [,1] [,2] [,3] [,4]
> [1,] 0.00000000 0.00000000 0 0.00
> [2,] 0.13144330 0.05714286 0 0.25
> [3,] 0.14432990 0.00000000 0 0.00
> [4,] 0.09278351 0.20000000 0 0.25
>
> $U[[15]]
> [,1] [,2] [,3] [,4]
> [1,] 0.00000000 0.00000000 0.00000000 0.00000000
> [2,] 0.00000000 0.00000000 0.01785714 0.00000000
> [3,] 0.02105263 0.01851852 0.03571429 0.04545454
> [4,] 0.02105263 0.01851852 0.03571429 0.06818182
>
> $U[[16]]
    [,1] [,2] [,3] [,4]
> [1,] 0.00 0.0000000 0.0000000 0.0000000
> [2,] 0.60 0.2857143 0.3333333 0.2413793
> [3,] 0.04 0.1428571 0.0000000 0.0000000
> [4,] 0.16 0.2857143 0.0000000 0.1724138
>
> $U[[17]]
> [,1] [,2] [,3] [,4]
> [1,] 0.00000000 0.00000000 0 0.00000000
> [2,] 0.28358209 0.17105263 0 0.16666667
> [3,] 0.08457711 0.02631579 0 0.05555556
> [4,] 0.13930348 0.44736842 0 0.30555556
>
> $U[[18]]
> [,1] [,2] [,3] [,4]
> [1,] 0.00000000 0.0000000 0.00000000 0.00000000
> [2,] 0.12698413 0.1052632 0.04761905 0.05479452
> [3,] 0.09523809 0.1578947 0.19047619 0.08219178
> [4,] 0.11111111 0.2236842 0.00000000 0.35616438
>
> $U[[19]]
> [,1] [,2] [,3] [,4]
> [1,] 0 0 0 0
> [2,] 0
> [3,] 0
```

```
> [4,] 1 1 0 0
>
> $U[[20]]
> [,1] [,2] [,3] [,4]
> [1,] 0 0.0000000 0 0
> [2,] 0 0.0000000 0 0
> [3,] 0 0.0000000 0 0
> [4,] 1 0.6666667 0 1
>
> $U[[21]]
> [,1] [,2] [,3] [,4]
> [1,] 0.00 0 0 0.0000000
> [2,] 0.00 0 0 0.1428571
> [3,] 0.25 0 0 0.0000000
> [4,] 0.25 0 0 0.5714286
>
> $U[[22]]
        [,1] [,2] [,3] [,4]
> [1,] 0.00 0.0000000 0 0.000
> [2,] 0.25 0.0000000 0 0.125
> [3,] 0.00 0.0000000 0 0.000
> [4,] 0.25 0.1666667 0 0.250
>
> $U[[23]]
> [,1] [,2] [,3] [,4]
> [1,] 0.0000000 0.0000000 0 0
> [2,] 0.2857143 0.0000000 0 0
> [3,] 0.2857143 0.3333333 0 0
> [4,] 0.2857143 0.3333333 0 1
>
> $U[[24]]
> [,1] [,2] [,3] [,4]
> [1,] 0.0000000 0 0.0000000 0.000
> [2,] 0.0000000 0 0.0000000 0.000
> [3,] 0.0000000 0 0.0000000 0.000
> [4,] 0.3333333 0 0.3333333 0.625
>
> $U[[25]]
> [,1] [,2] [,3] [,4]
> [1,] 0.0000000 0 0 0.0000000
> [2,] 0.1666667 0 0 0.0000000
> [3,] 0.0000000 0 0 0.0000000
> [4,] 0.0000000 0 0 0.1666667
>
> $U[[26]]
> [,1] [,2] [,3] [,4]
> [1,] 0.0000000 0.0 0 0
> [2,] 0.3333333 0.5 0 0
> [3,] 0.0000000 0.0 0
> [4,] 0.3333333
>
```

```
> $U[[27]]
> [,1] [,2] [,3] [,4]
> [1,] 0.0000000 0.00 0 0.0
> [2,] 0.0000000 0.00 0}00.
> [3,] 0.0000000 0.00 0}00.
> [4,] 0.1111111 0.75 0}00.
>
>
> $F
$F[[1]]
> [,1] [,2] [,3] [,4]
> [1,] 0 0 1.74 1.74
> [2,] 0 0 0.00 0.00
> [3,] 0 0 0.00 0.00
> [4,] 0 0 0.00 0.00
>
> $F[[2]]
> [,1] [,2] [,3] [,4]
> [1,] 0 0 0.3 0.6
> [2,] 0 0}00.
> [3,] 0 0}00.
> [4,] 0 0}00.
>
> $F[[3]]
> [,1] [,2] [,3] [,4]
> [1,] 0 0 0.50625 0.675
> [2,] 0 0 0.00000 0.000
> [3,] 0 0 0.00000 0.000
> [4,] 0 0 0.00000 0.000
>
> $F[[4]]
> [,1] [,2] [,3] [,4]
> [1,] 0 0 2.44 6.569231
> [2,] 0 0 0.00 0.000000
> [3,] 0 0 0.00 0.000000
> [4,] 0 0 0.00 0.000000
>
> $F[[5]]
> [,1] [,2] [,3] [,4]
> [1,] 0 0 0.45 0.6461538
> [2,] 0 0 0.00 0.0000000
> [3,] 0 0 0.00 0.0000000
> [4,] 0 0 0.00 0.0000000
>
> $F[[6]]
> [,1] [,2] [,3] [,4]
> [1,] 0 0 2.85 3.99
> [2,] 0 0 0.00 0.00
> [3,] 0 0 0.00 0.00
> [4,] 0 0 0.00 0.00
>
```

```
> $F[[7]]
[,1] [,2] [,3] [,4]
> [1,] 0 0 1.815 7.058333
> [2,] 0 0 0.000 0.000000
> [3,] 0 0 0.000 0.000000
> [4,] 0 0 0.000 0.000000
$F[[8]]
[,1] [,2] [,3] [,4]
> [1,] 0 0 1.233333 7.4
> [2,] 0 0 0.000000 0.0
> [3,] 0}000.000000 0.
> [4,] 0}000.000000 0.
>
> $F[[9]]
> [,1] [,2] [,3] [,4]
> [1,] 0 0 1.06 3.372727
> [2,] 0 0 0.00 0.000000
> [3,] 0 0 0.00 0.000000
> [4,] 0 0 0.00 0.000000
>
> $F[[10]]
> [,1] [,2] [,3] [,4]
> [1,] 0 0 0.2454545 2.1
> [2,] 0}000.000000
> [3,] 0}000.0000000 0.
> [4,] 0 0 0.0000000 0.0
>
> $F[[11]]
> [,1] [,2] [,3] [,4]
> [1,] 0 0 1.1 1.54
> [2,] 0 0 0.0 0.00
> [3,] 0 0 0.0 0.00
> [4,] 0 0 0.0 0.00
>
> $F[[12]]
> [,1] [,2] [,3] [,4]
> [1,] 0}000
> [2,] 0}0000
> [3,] 0
> [4,] 0 0 0
>
> $F[[13]]
> [,1] [,2] [,3] [,4]
> [1,] 0 0 1.785366 1.856522
> [2,] 0 0 0.000000 0.000000
> [3,] 0 0 0.000000 0.000000
> [4,] 0 0 0.000000 0.000000
>
> $F[[14]]
> [,1] [,2] [,3] [,4]
```

```
> [1,] 0 0 14.25 16.625
> [2,] 0
> [3,] 0}0000.00 0.00
> [4,] 0}0
>
> $F[[15]]
[,1] [,2] [,3] [,4]
> [1,] 0 0 0.5946429 1.765909
> [2,] 0 0 0.0000000 0.000000
> [3,] 0 0 0.0000000 0.000000
> [4,] 0 0 0.0000000 0.000000
>
> $F[[16]]
> [,1] [,2] [,3] [,4]
> [1,] 0 0 11.5 2.775862
> [2,] 0 0 0.0 0.000000
> [3,] 0 0 0.0 0.000000
> [4,] 0 0 0.0 0.000000
>
> $F[[17]]
> [,1] [,2] [,3] [,4]
> [1,] 0 0 3.78 1.225
> [2,] 0 0 0.00 0.000
> [3,] 0 0 0.00 0.000
> [4,] 0 0 0.00 0.000
>
> $F[[18]]
> [,1] [,2] [,3] [,4]
> [1,] 0 0 1.542857 1.035616
> [2,] 0 0 0.000000 0.000000
> [3,] 0 0 0.000000 0.000000
> [4,] 0 0 0.000000 0.000000
>
> $F[[19]]
> [,1] [,2] [,3] [,4]
> [1,] 0 0 0.15 0.175
> [2,] 0}000.000.00
> [3,] 0 0 0.00 0.000
> [4,] 0 0 0.00 0.000
>
> $F[[20]]
> [,1] [,2] [,3] [,4]
> [1,] 0 0 0 0.25
> [2,] 0 0 0 0.00
> [3,] 0 0 0 0.00
> [4,] 0 0 0 0.00
>
> $F[[21]]
> [,1] [,2] [,3] [,4]
> [1,] 0 0 0 1.428571
> [2,] 0 0 0 0.000000
```

```
> [3,] 0 0 0 0.000000
> [4,] 0}0000.00000
>
> $F[[22]]
> [,1] [,2] [,3] [,4]
> [1,] 0 0 0.7 0.6125
> [2,] 0 0 0.0 0.0000
> [3,] 0 0 0.0 0.0000
> [4,] 0 0 0.0 0.0000
>
> $F[[23]]
> [,1] [,2] [,3] [,4]
> [1,] 0 0 0 0.6
> [2,] 0
> [3,] 0
> [4,] 0 0 0 0.0
>
> $F[[24]]
    [,1] [,2] [,3] [,4]
> [1,] 0 0 0.7 0.6125
> [2,] 0}0
> [3,] 0 0 0.0 0.0000
> [4,] 0 0 0.0 0.0000
>
> $F[[25]]
> [,1] [,2] [,3] [,4]
> [1,] 0 0 2.1 0.8166667
> [2,] 0 0 0.0 0.0000000
> [3,] 0 0
> [4,] 0 0 0.0 0.0000000
>
> $F[[26]]
> [,1] [,2] [,3] [,4]
> [1,] 0
> [2,] 0
> [3,] 0
>[4,] 0
>
$F[[27]]
> [,1] [,2] [,3] [,4]
> [1,] 0 0 0 1.4
> [2,] 0}00\quad0\quad0.
> [3,] 0}00 0 0.
> [4,] 0}00\quad0\quad0.
>
>
> $hstages
> [1] NA
>
> $agestages
> [1] NA
```



| $>17$ | 1 | 0 | 2004 |
| :---: | :---: | :---: | :---: |
| $>18$ | 1 | 0 | 2005 |
| > 19 | 1 | Q | 2003 |
| > 20 | 1 | Q | 2004 |
| > 21 | 1 | Q | 2005 |
| > 22 | 1 | R | 2003 |
| > 23 | 1 | R | 2004 |
| > 24 | 1 | R | 2005 |
| $>25$ | 1 | S | 2003 |
| $>26$ | 1 | S | 2004 |
| $>27$ | 1 | S | 2005 |
| > |  |  |  |
| > \$matrixqc |  |  |  |
| > [1] | 167 | 48 | 27 |
| > |  |  |  |
| > \$dataqc |  |  |  |
| $>$ [1] NA NA |  |  |  |
| > |  |  |  |
| > attr(,"class") |  |  |  |
| > [1] "lefkoMat" |  |  |  |

The resulting object has all of the elements of a standard lefkoMat object except for those elements related to quality control in the demographic dataset and linear modeling. The option UFdecomp was left at its default (UFdecomp = TRUE), and so create_1M() used the stageframe to infer where fecundity values were located in the matrices and created $U$ and $F$ matrices separating those values. This separation was performed on the basis of the stageframe, which shows which two stages are reproductive and which one stage acts as the entry stage into the population. The default option for historical is set to FALSE, yielding an NA in place of the hstages element, which would typically list the order of historical stage pairs.

Let's now take a look at a summary of this lefkoMat object.

```
summary(anth_lefkoMat)
>
> This ahistorical lefkoMat object contains 27 matrices.
>
> Each matrix is square with 4 rows and columns, and a total of 16 elements.
> A total of 167 survival transitions were estimated, with 6.185 per matrix.
> A total of 48 fecundity transitions were estimated, with 1.778 per matrix.
> This lefkoMat object covers 1 population, 9 patches, and 3 time steps.
>
> This lefkoMat object appears to have been imported. Number of unique individuals and transitions nc
>
> Survival probability sum check (each matrix represented by column in order):
\(>\) [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11]
Min. 0.0667 0.500 0.0625 0.0769 0.0909 0.000 0.0000 0.333 0.171 0.000 0.0000
1st Qu. 0.0810 0.575 0.1708 0.1192 0.1334 0.075 0.0375 0.405 0.268 0.170 0.0000
Median 0.1198 0.657 0.2106 0.3077 0.2276 0.100 0.1706 0.453 0.350 0.239 0.0000
Mean 0.1599 0.637 0.2209 0.4231 0.2303 0.175 0.1895 0.469 0.320 0.203 0.0556
3rd Qu. 0.1987 0.720 0.2607 0.6116 0.3245 0.200 0.3225 0.517 0.402 0.271 0.0556
Max. 0.3333 0.736 0.4000 1.0000 0.3750 0.500 0.4167 0.636 0.409 0.333 0. 2222
[,12] [,13] [,14] [,15] [,16] [,17] [,18] [,19] [,20] [,21] [,22]
Min. 0.000 0.0000 0.000 0.0370 0.333 0.000 0.238 0.00 0.000 0.000 0.000
```

```
> 1st Qu. 0.375 0.0000 0.193 0.0408 0.394 0.381 0.310 0.00 0.500 0.000 0.125
> Median 0.500 0.0163 0.313 0.0657 0.564 0.518}00.410 0.00 0.833 0.250 0.271
> Mean 0.409 0.0510 0.281 0.0705 0.565 0.420}00.388 0.25 0.667 0.304 0.260
> 3rd Qu. 0.534 0.0673 0.401 0.0954 0.736 0.557 0.488 0.25 1.000 0.554 0.406
>Max. 0.636 0.1714 0.500 0.1136 0.800 0.645 0.493 1.00 1.000 0.714 0.500
> [,23] [,24] [,25] [,26] [,27]
> Min. 0.000 0.000 0.0000 0.000 0.0000
> 1st Qu. 0.500 0.250 0.0000 0.375 0.0833
> Median 0.762 0.333 0.0833 0.583 0.2556
> Mean 0.631 0.323 0.0833 0.542 0.3153
> 3rd Qu. 0.893 0.406 0.1667 0.750 0.4875
> Max. 1.000 0.625 0.1667 1.000 0.7500
```

The summary of this new lefkoMat object shows us that we have 27 matrices with four rows and columns each. The estimated numbers of transitions corresponds to the non-zero entries in each matrix. We see that we are covering one population, nine patches, and three time steps. All of the survival probabilities observed fall within the bounds of 0 to 1 , so everything seems alright. At this point, we can use this object in lefko3's various projection analyses.

### 12.2 Importing matrices from COMPADRE and COMADRE

Users may be aware of the amazing MPM databases COMPADRE and COMADRE, which were created and are maintained with the aim of holding all published MPMs in an open repository (SalgueroGómez et al., 2015; Salguero-Gómez et al., 2016). The former covers MPMs for plants, fungi, and other microbes, while the latter covers animals and their closest relatives. The create_1M() function, which is the means to important lists of matrices, also allows matrices to be imported into lefkoMat format from these databases. We encourage the user to explore package Rcompadre, which includes a host of functions to allow the exploration of these databases (Jones et al., 2022). We will also make use of one function from this package in our code below, although this is not required.

To start, we need to access one of these databases for use. There are two approaches that we might use for this purpose. One approach is to download the database of interest from the web - they are available at https://compadre-db.org/. An alternative approach is to download them using the cdb_fetch() function in Rcompadre. In the next block, we load both database after downloading it manually.

```
load("COMPADRE_v.6.22.5.0.RData")
load("COMADRE_v.4.21.8.0.RData")
summary(compadre)
> Length Class Mode
> metadata 58 data.frame list
> matrixClass 8851 -none- list
> mat 8851 -none- list
> version 7 -none- list
summary(comadre)
> Length Class Mode
> metadata 58 data.frame list
> matrixClass 3317 -none- list
> mat 3317 -none- list
> version 7 -none- list
```

In the next block, we use Rcompadre to download the databases.

```
Compadre <- Rcompadre::cdb_fetch("compadre")
> This is COMPADRE version 6.23.5.0 (release date May_06_2023)
> See user agreement at https://compadre-db.org/Help/UserAgreement
> See how to cite with `citation(Rcompadre)`
Comadre <- Rcompadre::cdb_fetch("comadre")
> This is COMADRE version 4.23.3.1 (release date Mar_24_2023)
> See user agreement at https://compadre-db.org/Help/UserAgreement
> See how to cite with `citation(Rcompadre)`
summary(Compadre)
> Length Class Mode
> 1 CompadreDB S4
summary(Comadre)
> Length Class Mode
> 1 CompadreDB S4
```

Users will notice discrepancies in the summaries of the databases downloaded manually from the web vs those downloaded using Rcompadre. The database structure differs somewhat, but the data is essentially the same, and lefko3 can handle both structures.

Once the databases are in R's global environment, the user can explore them searching for the MPMs of interest. Essentially, the database may be searched to find published papers, species or other taxa, locations, and other characteristics of interest. Regardless of what characteristics the user is looking for, it is important to remember that lefko3 can only produce lefkoMat objects with matrices of the same dimension, because a single lefkoMat object is meant to hold a single MPM. Therefore, matrices of different dimensions will lead to fatal errors in processing.

In the code below, we create a new lefkoMat object using 4 separate matrices. These matrices cover a population of the seaweed Ascophyllum nodosum, surveyed near the city of Göteberg, Sweden (Åberg, 1990).

```
mpm_a <- create_lM(matrix_id = c(238271, 238272, 238273, 238274), mats = compadre)
summary(mpm_a)
>
> This ahistorical lefkoMat object contains 4 matrices.
>
> Each matrix is square with 5 rows and columns, and a total of 25 elements.
> A total of 63 survival transitions were estimated, with 15.75 per matrix.
> A total of 16 fecundity transitions were estimated, with 4 per matrix.
> This lefkoMat object covers 1 population, 1 patch, and 1 time step.
>
> This lefkoMat object appears to have been imported. Number of unique individuals and transitions no
>
> Survival probability sum check (each matrix represented by column in order):
> [,1] [,2] [,3] [,4]
>Min. 0.630 0.800 0.28 0.240
> 1st Qu. 0.630 0.830 0.45 0.290
> Median 0.770 0.960 0.50 0.310
l Mean 0.754 0.918 0.51 0.362
> 3rd Qu. 0.820 1.000 0.52 0.370
> Max. 0.920 1.000 0.80 0.600
```

Note that the summary looks exactly as a lefkoMat object summary should. Let's dig into the MPM structure a bit, as well.

```
mpm_a
> $A
> $A[[1]]
> A1 A2 A3 A4 A5
> [1,] 0.48 0.30 0.70 2.39 7.84
> [2,] 0.13 0.28}00.07[0.03 0.01
> [3,]}00.02 0.33 0.40 0.19 0.04
> [4,]}00.00 0.02 0.29 0.41 0.57
> [5,] 0.00 0.00}00.010.0.19 0.3
>
> $A[[2]]
> A1 A2 A3 A4 A5
> [1,] 0.55 0.32 0.64 2.37 7.82
> [2,] 0.25 0.33 0.06 0.00 0.00
> [3,]}0.030.45 0.50 0.18 0.00
> [4,] 0.00 0.02 0.40}00.55 0.6
> [5,] 0.00 0.00 0.00 0.27 0.40
>
> $A[[3]]
> A1 A2 A3 A4 A5
> [1,] 0.38 0.23 0.79 2.40 7.82
> [2,] 0.11 0.18}00.100.07 0.0
> [3,]}0.01 0.10 0.23 0.26 0.1
> [4,] 0.00 0.00 0.10 0.15 0.60
> [5,] 0.00 0.00 0.02 0.04 0.10
>
> $A[[4]]
\(>\quad\) A1 A2 A3 A4 A5
> [1,] 0.28 0.46 0.85 2.58 8.05
> [2,]}00.02 0.15 0.09 0.11 0.15
> [3,]}00.01 0.09 0.20 0.04 0.1
> [4,] 0.00 0.00 0.00 0.18}00.1
> [5,] 0.00 0.00 0.00}00.040.1
>
>
> $U
> $U[[1]]
> U1 U2 U3 U4 U5
> [1,] 0.48 0.00 0.00 0.00 0.00
> [2,] 0.13 0.28 0.07 0.03 0.01
> [3,] 0.02 0.33 0.40}00.19 0.04
> [4,]}00.00 0.02 0.29 0.41 0.57
> [5,] 0.00 0.00 0.01 0.19 0.30
>
> $U[[2]]
> U1 U2 U3 U4 U5
> [1,] 0.55 0.00 0.00 0.00 0.0
> [2,] 0.25 0.33 0.06 0.00 0.0
> [3,] 0.03 0.45 0.50 0.18 0.0
> [4,] 0.00 0.02 0.40 0.55 0.6
> [5,] 0.00 0.00 0.00 0.27 0.4
```

```
>
$U[[3]]
> U1 U2 U3 U4 U5
> [1,] 0.38 0.00 0.00 0.00 0.0
> [2,] 0.11 0.18}00.10\quad0.070.
> [3,] 0.01 0.10}00.23 0.26 0.1
> [4,] 0.00 0.00 0.10 0.15 0.6
> [5,] 0.00 0.00 0.02 0.04 0.1
>
> $U[[4]]
> U1 U2 U3 U4 U5
> [1,] 0.28 0.00 0.00 0.00 0.00
> [2,]}00.02 0.15 0.09 0.11 0.1
> [3,] 0.01 0.09 0.20}00.04\quad0.1
> [4,] 0.00 0.00 0.00 0.18 0.15
> [5,] 0.00 0.00 0.00 0.04 0.15
>
>
> $F
$F[[1]]
            F1 F2 F3 F4 F5
> [1,] 0 0.3 0.7 2.39 7.84
> [2,] 0 0.0 0.0 0.00 0.00
> [3,] 0 0.0 0.0 0.00 0.00
> [4,] 0 0.0 0.0 0.00 0.00
> [5,] 0 0.0 0.0 0.00 0.00
>
> $F[[2]]
> F1 F2 F3 F4 F5
> [1,] 0 0.32 0.64 2.37 7.82
> [2,] 0 0.00 0.00 0.00 0.00
> [3,] 0 0.00 0.00 0.00 0.00
> [4,] 0 0.00 0.00 0.00 0.00
> [5,] 0 0.00 0.00 0.00 0.00
>
$F[[3]]
> F1 F2 F3 F4 F5
> [1,] 0 0.23 0.79 2.4 7.82
> [2,] 0 0.00 0.00 0.0 0.00
> [3,] 0 0.00 0.00 0.0 0.00
> [4,] 0 0.00 0.00 0.0 0.00
> [5,] 0 0.00 0.00 0.0 0.00
>
> $F[[4]]
            F1 F2 F3 F4 F5
> [1,] 0 0.46 0.85 2.58 8.05
> [2,] 0 0.00 0.00 0.00 0.00
> [3,] 0 0.00 0.00 0.00 0.00
> [4,] 0 0.00 0.00 0.00 0.00
> [5,] 0 0.00 0.00 0.00 0.00
>
```

```
>
> $ahstages
> stage size size_b size_c min_age max_age repstatus obsstatus propstatus
```



```
> sizebin_center sizebin_width binhalfwidthb_raw sizebinb_min sizebinb_max
\begin{tabular}{llllll}
\(>1\) & 1 & 1 & 0 & 0 & 0
\end{tabular}
\begin{tabular}{llllll}
\(>2\) & 2 & 1 & 0 & 0 & 0
\end{tabular}
\begin{tabular}{llllll}
\(>3\) & 3 & 1 & 0 & 0 & 0
\end{tabular}
\begin{tabular}{llllll}
\(>\) & 4 & 4 & 1 & 0 & 0 \\
\(>\) & 5 & 1 & 0 & 0 & 0
\end{tabular}
> sizebinb center sizebinb width binhalfwidthc raw sizebinc min sizebinc max
> 1 0 0 0 0 0 0
\(>2\)
> 3 0
>4 
> sizebinc_center sizebinc_width group comments
> 1 0 0 0 0 0-<5 g
>2 0 0 0 5-<15 g
> 3 0 0 0 15-<54 g
>4 0 0 0 54-<<190 g
> 5 0 0 0 >= 190g
>
    $hstages
> NA
1 NA
>
$agestages
    NA
    1 NA
>
> $labels
> pop patch year2
> 1 Göteborg Unmanipulated 1985
> 2 Göteborg Unmanipulated 1985
> 3 Göteborg Unmanipulated 1985
> 4 Göteborg Unmanipulated 1985
>
> $matrixqc
> [1] 63 16 4
>
```

```
> $dataqc
> [1] NA NA
>
> attr(,"class")
> [1] "lefkoMat"
```

The structure of the lefkoMat object shows that R has accomplished quite a lot for us. Under the default settings for create_lM(), it has imported the stage structure associated with the matrices, and has used that in combination with the $F$ and $C$ matrices in the COMPADRE entry to determine which stages are reproductive, which stages are entry stages, and what the U and F matrices should look like. It has also imported metadata to construct the labels object, and has identified the non-zero entries in the U and F matrices.

Let's try this again, but this time using a single matrix from the COMADRE database. Here, we import a single matrix covering a population of moose, Alces alces, surveyed in Alaska, USA (Ballard et al., 1991).

```
mpm_b <- create_lM(matrix_id = 240296, mats = comadre)
mpm_b
> $A
    $A[[1]]
    A1 A2 A3
    [1,] 0.000 0.000 1.120
    [2,] 0.342 0.000 0.000
    [3,] 0.000 0.951 0.948
>
>
    $U
    $U[[1]]
> U1 U2 U3
    [1,] 0.000 0.000 0.000
    [2,] 0.342 0.000 0.000
> [3,] 0.000 0.951 0.948
>
>
> $F
$F[[1]]
> F1 F2 F3
> [1,] 0 0 1.12
> [2,] 0 0 0.00
> [3,] 0 0 0.00
>
>
> $ahstages
> stage size size_b size_c min_age max_age repstatus obsstatus propstatus
\begin{tabular}{llllllllll}
\(>\) & 1 & stage 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1
\end{tabular}
\begin{tabular}{llllllllll}
\(>2\) & stage 2 & 2 & 0 & 0 & 0 & 0 & 0 & 1 & 0
\end{tabular}
> 3 stage 3 3 3 0 <rlllllll
> immstatus matstatus indataset binhalfwidth_raw sizebin_min sizebin_max
l 1 1 0
>2 0 1 1 0
> 3 0 1 1 0
> sizebin_center sizebin_width binhalfwidthb_raw sizebinb_min sizebinb_max
```

```
> 1 >lllll
> sizebinb_center sizebinb_width binhalfwidthc_raw sizebinc_min sizebinc_max
> 1 0 0 0 0
> 2 0 0 0 0
> 3 0 0 0 0 0
sizebinc_center sizebinc_width group comments
> 1 0 0 0 Calf: 0-1 years
> 2 0 0 0 Yearling: 1-2 years
3 0 0 0 Adult: 2+ years
>
$hstages
NA
1 NA
>
$agestages
    NA
    1 NA
$labels
> pop patch year2
1 Susitna River, Alaska Unmanipulated 1976
$matrixqc
[1] 3 1 1
>
$dataqc
[1] NA NA
attr(,"class")
[1] "lefkoMat"
```

So we see that we have imported these properly.
Function create_1M() includes a default setting to add the matC and matF portions of the COMPADRE and COMPADRE databases to produce the associated F matrix in lefko3. In other words, lefko3 by default adds clonal and sexual reproductive pathways to produce the fecundity matrix. Users may prefer to create an F matrix only from the associated sexual reproduction pathway, using the matF entries in these databases. Let's take a look at how we can work with this, using an example of a Mimulus guttatus population surveyed (Pantoja et al., 2018). First, the default setting.

```
mimulus_a <- create_lM(matrix_id = c(240006, 240007, 240008), mats = compadre)
mimulus_a$F
> [[1]]
> F1 F2 F3
> [1,] 0 8.755405 8.755405
> [2,] 0 17.380133 17.380133
> [3,] 0 1.112686 1.112686
>
> [[2]]
> F1 F2 F3
> [1,] 0
```

```
> [2,] 0 23.250850 23.250850
> [3,] 0
>
> [[3]]
> F1 F2 F3
> [1,] 0 1.885204 1.885204
> [2,] 0 23.250850 23.250850
> [3,] 0 8.369627 8.369627
```

Now, let's define the F matrices using only what we know about sexual reproduction. Note the use of the add_FC option, below.

Let's now explore how to edit matrices within MPMs.

```
mimulus_b <- create_lM(matrix_id = c(240006, 240007, 240008), mats = compadre,
    add_FC = FALSE)
mimulus_b$F
> [[1]]
> F1 F2 F3
> [1,] 0 8.755405 8.755405
> [2,] 0 17.380133 17.380133
> [3,] 0}00.0000000.00000
>
> [[2]]
> F1 F2 F3
> [1,] 0
> [2,] 0 23.250850 23.250850
> [3,] 0 0.000000 0.000000
>
> [[3]]
> F1 F2 F3
> [1,] 0 1.885204 1.885204
> [2,] 0 23.250850 23.250850
> [3,] 0}00.0000000.00000
```

There are obvious differences here, most notably that the bottom row is now all zeroes. We can get a sense of why this is by looking at the comments section of the stageframe, which imports COMPADRE and COMADRE's stage names.

```
mimulus_a$ahstages$comments
> [1] "Seed" "Seedling" "Rosette"
```

It appears that reproductive transitions in the third row correspond to clonal fission in adult rosettes. This may be quite important to reproduction. We can quickly assess the importance by estimating the deterministic population growth rate in both MPMs and looking at the differences.

```
lambda3(mimulus_a)
> pop patch year2 lambda
> 1 Native-annual Cross Waterlogged 2015 19.15122
> 2 Native-perennial Cross Waterlogged 2015 32.90595
> 3 Introduced-perennial Cross Waterlogged 2015 32.62125
lambda3(mimulus_b)
> pop patch year2 lambda
```

```
> 1 Native-annual Cross Waterlogged 2015 19.15122
> 2 Native-perennial Cross Waterlogged 2015 32.90595
> 3 Introduced-perennial Cross Waterlogged 2015 32.62125
```

Amazingly enough, these matrices have amazingly high $\lambda$ values, and they appear to be unaffected by the lack of clonal reproduction in the second MPM. The authors developed an experimental study which, while conducted in the field, was nonetheless conducted as a study in which greenhouse seedlings were planted as an artificial population, and so it may be that the vital rates were monitored prior to any sort of equilibrium establishing itself. In any case, these sorts of resuls warrant further study.

### 12.3 Points to remember

1. Matrices may be imported for use in lefko3 via a variety of approaches, ranging from loading CSV files to direct text input to input from open access, online databases.
2. Properly important matrices into lefko3 requires the creation of a stageframe that characterizes each stage used. All characteristics other than size need to be properly documented.
3. Function create_1M() will also separate fecundity from survival using the stageframe as a guide, creating $U$ and $F$ matrices in addition to $A$ matrices.

## Chapter 13

## Further Issues II: Importing IPMs and Function-based MPMs

More and more of our imports come from overseas.

\author{

- George W. Bush
}

It is relatively common for population ecologists running matrix projection analyses to find themselves wanting to build IPMs and function-based MPMs (fbMPMs) published by other authors. For example, users might find themselves reading an interesting paper utilizing IPMs, such as Childs et al. (2003) or Dahlgren and Ehrlén (2011), and wanting to replicate the analyses. Fortunately, lefko3 includes tools to make this relatively easy.

The strategy used in lefko3 is to take all of the characteristics of an IPM or a fbMPM and input those characteristics as data in a vrm_input object. This S3 object is essentially a list that contains up to 13 elements, most of which are structured data frames that need to be edited manually. The function that creates these objects is vrm_import().

To use function vrm_import (), users should first determine the monitoring occasions or years for which information will be entered, the number of patches to be covered, whether there are any stage groups, which vital rate models should be developed, whether information is provided for interactions between independent terms, whether any independent factors are categorical, and the distributions and associated parameters assumed for size and fecundity. Although a stageframe is not required to create vrm_input objects, a stageframe will be required to create the IPMs, fbMPMs, or projections resulting from them, so users should pay serious attention to creating realistic stageframes.

Let's consider this process using the simplest vrm_input object, which includes only terms for main effects, three years, and a single patch, with Gaussian primary size and fecundity. Let's say that our monitoring occurred over the years 2020, 2021, and 2022. This yields only two years at time $t$, since 2022 includes only data on which stages have been transitioned to, and so we include only those two years in the input options (fates in 2022 act only as responses within the vital rate models, so we cannot add 2022 to the input).

```
vrm_example <- vrm_import(years = c(2020, 2021))
vrm_example
> $vrm_frame
> main_effect_1
> 1 intercept
> 2 size2
> 3 size1
> 4 sizeb2
\begin{tabular}{rrrrrr} 
main_1_defined & surv & obs & sizea & sizeb & sizec \\
y-intercept & 0 & 0 & 0 & 0 & 0 \\
sizea in time t & 0 & 0 & 0 & 0 & 0 \\
sizea in time t-1 & 0 & 0 & 0 & 0 & 0 \\
sizeb in time t & 0 & 0 & 0 & 0 & 0
\end{tabular}
```



```
>
$group1_frame
    groups surv obs sizea sizeb sizec repst fec jsurv jobs jsizea jsizeb jsizec
    1 0}0
        jrepst jmatst
1 0 0
$dist_frame
    response dist
1 surv binom
2 obs constant
3 sizea gaussian
> 4 sizeb constant
5 sizec constant
6 repst constant
7 fec gaussian
8 jsurv constant
9 jobs constant
10 jsizea constant
11 jsizeb constant
12 jsizec constant
13 jrepst constant
14 jmatst constant
>
$st_frame
> surv 
jsizec jrepst jmatst
> 1 1 1 1 
>
> attr(,"class")
[1] "vrm_input"
```

Our example vrm_input objects is a list holding only 7 elements. The first, vrm_frame, is a data frame showing us the main effects factors that we can set for each of up to 14 vital rate models. The 14 vital rate models are the same as those listed in the chapter on function-based MPMs (5): survival (surv), observation status (obs), primary size (sizea), secondary size (sizeb), tertiary size (sizec), reproductive status (repst), fecundity (fec), juvenile survival (jsurv), juvenile observation status ( jobs ), juvenile primary size ( jsizea ), juvenile secondary size ( jsizeb ), juvenile tertiary size (jsizec), juvenile reproductive status (jrepst), and juvenile maturity status (jmatst). We see rows in which we can alter the coefficients to be used in linear models of vital rates, including the yintercept (intercept), primary size in time $t$ (size2), tertiary size in time $t$-1 (sizec1), spatial density (density), etc. The general strategy is for users to change the coefficients that they have explicit values for, and to leave the rest as 0 .

Object year_frame allows us to set specific coefficients for each year associated with each vital rate model. This is particularly useful as the IPM in question used a mixed modeling approach with year as a random term, although it serves the same function with year as a fixed categorical term. Object patch_frame functions in the same way, except for patches or subpopulations (we do not currently offer a top-level population category, but the patch level can be used for populations if the latter are not to be subdivided). Objects group2_frame and group1_frame provide coefficients associated with different life history stage groups in time $t$ and time $t-1$, respectively, if such information is provided and stages are grouped in a way that is incorporated into linear modeling.

Object dist_frame gives the distribution of each vital rate model's response term. Any vital rate that uses a linear model for estimation needs to be set to binom (for binomial distribution), gaussian (for the Gaussian distribution), gamma (for the gamma distribution), poisson (for the Poisson distribution), or negbin (for the negative binomial distribution). Unused vital rates should be set to constant. Object st_frame allows the user to set $\sigma$ if the distribution is Gaussian, or $\theta$ if the distribution is negative binomial (all other cases should be left as 1).

Let's now take a look at some examples.

### 13.1 Generating standard ahistorical IPMs and function-based MPMs

Let's start off by recreating the ahistorical IPM set up in Chapter 7. This example will focus on Lathyrus vernus (section 1.8.2). To remind ourselves of this plant and its dataset, let's take a look at a data summary. Note that in a real example from the literature, we might not have the original dataset to work with, but would need to infer certain characteristics about the dataset from the publication.

```
data(lathyrus)
summary(lathyrus)
> SUBPLOT
> Min. :1.000
> 1st Qu.:2.000
> Median :3.000
> Mean :3.223
> 3rd Qu.:4.000
> Max. :6.000
>
> FCODE88
> Min. :0.0000
> 1st Qu.:0.0000
> Median :0.0000
> Mean :0.3399
> 3rd Qu.:1.0000
> Max. :1.0000
> NA's :404
> Missing1988 Seedling1988
> Mode:logical Min. :1.000
> NA's:1119
> 
> Median :2.000
> 3rd Qu.:3.000
> Max. :3.000
> NA'
> FCODE89
> Min. :0.0000
> 1st Qu.:0.0000
> Median :0.0000
> Mean :0.2667
> 3rd Qu.:1.0000
> Max. :1.0000
> NA's :294 NA's :906
D Dormant1989 Missing1989 Seedling1989 Volume90 lnVol90
```



The data were collected over four years - 1988 through 1991. Size is given in two formats - either by the total leaf volume (Volume), or by the natural logarithm of the total leaf volume (lnVol). As in the previous case, let's use the total volume for this example. We also need to note the minimum and maximum size, and whether an unobservable size class is included. The minimum leaf volume appears to be be 1.8 , while the maximum appears to be 7032.0 , and the literature on this plant shows that it can experience vegetative dormancy, during which there would be no aboveground tissue and hence leaf volume would be 0 .

As before, we need to create a stageframe describing the life history and the actual stages that will compose the discretized IPM. We do so below, utilizing the information that we have gleaned from our study of the publication involved, and then show four key variables within that output. Note that this stageframe covers a complex IPM, involving 100 size-classified stages and three stages that do not fit in the size gradient (dormant seed, seedling, and vegetative dormancy).

```
sizevector <- c(0, 100, 0, 1, 7100)
stagevector <- c("Sd", "Sdl", "Dorm", "ipm", "ipm")
repvector <- c(0, 0, 0, 1, 1)
obsvector <- c(0, 1, 0, 1, 1)
matvector <- c(0, 0, 1, 1, 1)
immvector <- c(1, 1, 0, 0, 0)
propvector <- c(1, 0, 0, 0, 0)
indataset <- c(0, 1, 1, 1, 1)
binvec <- c(0, 100, 0.5, 1, 1)
comments <- c("Dormant seed", "Seedling", "Dormant", "ipm adult stage",
    "ipm adult stage")
lathframeipm <- sf_create(sizes = sizevector, stagenames = stagevector,
    repstatus = repvector, obsstatus = obsvector, propstatus = propvector,
    immstatus = immvector, matstatus = matvector, comments = comments,
    indataset = indataset, binhalfwidth = binvec, ipmbins = 100, roundsize = 3)
lathframeipm[,c("stage", "size", "sizebin_min", "sizebin_max", "comments")]
```

> stage size sizebin_min sizebin_max comments
$\begin{array}{llll}>1 & \text { Sd } & 0.000 & 0.00\end{array}$
$\begin{array}{llllll}>2 & \text { Sdl } & 100.000 & 0.00 & 200.00 & \text { Seedling }\end{array}$
$\begin{array}{llllll}> & \text { Dorm } & 0.000 & -0.50 & 0.50 & \text { Dormant }\end{array}$
$\begin{array}{llll}> & \text { sza_36.495_0 } & 36.495 & 1.00 \\ 71.99 & \text { ipm adult stage }\end{array}$
> 5 sza_107.48_0 $107.485 \quad 71.99 \quad 142.98$ ipm adult stage
$>6$ sza_178.47_0 $178.475 \quad 142.98 \quad 213.97$ ipm adult stage
$>7$ sza_249.46_0 $249.465 \quad 213.97 \quad 284.96$ ipm adult stage
$>8$ sza_320.45_0 $320.455 \quad 284.96 \quad 355.95$ ipm adult stage
$>9$ sza_391.44_0 $391.445 \quad 355.95 \quad 426.94$ ipm adult stage
> 10 sza_462.43_0 $462.435 \quad 426.94 \quad 497.93$ ipm adult stage
> 11 sza_533.42_0 $533.425 \quad 497.93 \quad 568.92$ ipm adult stage
> 12 sza_604.41_0 $604.415 \quad 568.92 \quad 639.91$ ipm adult stage
$>13$ sza_675.40_0 675.405 639.91 710.90 ipm adult stage
> 14 sza_746.39_0 746.395
sza_817.38_0 817.385
sza_888.37_0 888.375
sza_959.36_0 959.365
sza_1030.3_0 1030.355
sza_1101.3_0 1101.345
sza_1172.3_0 1172.335
. 85
1136.84
sza_1243.3_0 $1243.325 \quad 1207.83 \quad 1278.82$ ipm adult stage
sza_1314.3_0 1314.315 $1278.82 \quad 1349.81$ ipm adult stage
sza_1385.3_0 $1385.305 \quad 1349.81 \quad 1420.80$ ipm adult stage
sza_1456.2_0 1456.295 $1420.80 \quad 1491.79$ ipm adult stage
sza_1527.2_0 1527.285 1491.79 1562.78 ipm adult stage
sza_1598.2_0 1598.275 $1562.78 \quad 1633.77$ ipm adult stage
sza_1669.2_0 1669.265 1633.77
sza_1740.2_0 $1740.255 \quad 1704.76$
sza_1811.2_0 $1811.245 \quad 1775.75$
sza_1882.2_0 $1882.235 \quad 1846.74$
sza_1953.2_0 1953.225 $1917.73 \quad 1988.72$ ipm adult stage
> 32 sza_2024.2_0 $2024.215 \quad 1988.72 \quad 2059.71$ ipm adult stage
> 33 sza_2095.2_0 2095.205 $2059.71 \quad 2130.70$ ipm adult stage

```
> 34
> 35
> 36 sza_2308.1_0 2308.175
> 37 sza_2379.1_0 2379.165
> 38 sza_2450.1_0 2450.155
> 39 sza_2521.1_0 2521.145
> 40 sza_2592.1_0 2592.135
> 41 sza_2663.1_0 2663.125
> 42 sza_2734.1_0 2734.115
> 43 sza_2805.1_0 2805.105
> 44
>45
>46
>47
> 48
> 49
> 50
> 51
> 52
> 53
> 54
> 55
> 56
> 57
> 58
> 59
> 60
> 61
> 62
> 63
> 64
> 65
> 66
> 67
> 68
> }6
> 70
> 71
> 72
> 73
> 74
> 75
> 76
> 77
> 78
> 79
> 80
> 81
> 82
> 83
> 84
```

sza_2166.1_0 2166.195
sza_2237.1_0 2237.185 sza_2308.1_0 2308.175 sza_2379.1_0 2379.165 sza_2450.1_0 2450.155 sza_2521.1_0 2521.145 sza_2592.1_0 2592.135
sza_2663.1_0 2663.125
sza_2734.1_0 2734.115
sza_2805.1_0 2805.105
sza_2876.0_0 2876.095 sza_2947.0_0 2947.085 sza_3018.0_0 3018.075 sza_3089.0_0 3089.065 sza_3160.0_0 3160.055 sza_3231.0_0 3231.045 sza_3302.0_0 3302.035 sza_3373.0_0 3373.025 sza_3444.0_0 3444.015 sza_3515.0_0 3515.005 sza_3585.9_0 3585.995 sza_3656.9_0 3656.985 sza_3727.9_0 3727.975 sza_3798.9_0 3798.965 sza_3869.9_0 3869.955 sza_3940.9_0 3940.945 sza_4011.9_0 4011.935 sza_4082.9_0 4082.925 sza_4153.9_0 4153.915 sza_4224.9_0 4224.905 sza_4295.8_0 4295.895 sza_4366.8_0 4366.885 sza_4437.8_0 4437.875 sza_4508.8_0 4508.865 sza_4579.8_0 4579.855 sza_4650.8_0 4650.845 sza_4721.8_0 4721.835 sza_4792.8_0 4792.825 sza_4863.8_0 4863.815 sza_4934.8_0 4934.805 sza_5005.7_0 5005.795 sza_5076.7_0 5076.785 sza_5147.7_0 5147.775 sza_5218.7_0 5218.765 sza_5289.7_0 5289.755 sza_5360.7_0 5360.745 sza_5431.7_0 5431.735 sza_5502.7_0 5502.725 sza_5573.7_0 5573.715 sza_5644.7_0 5644.705 sza_5715.6_0 5715.695
2130.70
2201.69 ipm adult stage 2201.69 2272.68 2343.67 2414.66 2485.65
2556.64
2627.63
2698.62
2769.61
2840.60
2911.59
2982.58
3053.57
3124.56
3195.55
3266.54
3337.53
3408.52
3479.51
3550.50
3621.49
3692.48
3763.47
3834.46
3905.45
3976.44
4047.43
4118.42
4189.41
4260.40
4331.39
4402.38
4473.37
4544.36
4615.35
4686.34
4757.33
4828.32
4899.31
4970.30
5041.29
5112.28
5183.27
5254.26
5325.25
5396.24
5467.23
$5538.22 \quad 5609.21 \mathrm{ipm}$ adult stage
$5609.21 \quad 5680.20$ ipm adult stage
$5680.20 \quad 5751.19$ ipm adult stage 2272.68 ipm adult stage 2343.67 ipm adult stage 2414.66 ipm adult stage 2485.65 ipm adult stage 2556.64 ipm adult stage 2627.63 ipm adult stage 2698.62 ipm adult stage 2769.61 ipm adult stage 2840.60 ipm adult stage 2911.59 ipm adult stage 2982.58 ipm adult stage 3053.57 ipm adult stage 3124.56 ipm adult stage 3195.55 ipm adult stage 3266.54 ipm adult stage 3337.53 ipm adult stage 3408.52 ipm adult stage 3479.51 ipm adult stage 3550.50 ipm adult stage 3621.49 ipm adult stage 3692.48 ipm adult stage 3763.47 ipm adult stage 3834.46 ipm adult stage 3905.45 ipm adult stage 3976.44 ipm adult stage 4047.43 ipm adult stage 4118.42 ipm adult stage 4189.41 ipm adult stage 4260.40 ipm adult stage 4331.39 ipm adult stage 4402.38 ipm adult stage 4473.37 ipm adult stage 4544.36 ipm adult stage 4615.35 ipm adult stage 4686.34 ipm adult stage 4757.33 ipm adult stage 4828.32 ipm adult stage 4899.31 ipm adult stage 4970.30 ipm adult stage 5041.29 ipm adult stage 5112.28 ipm adult stage 5183.27 ipm adult stage 5254.26 ipm adult stage 5325.25 ipm adult stage 5396.24 ipm adult stage 5467.23 ipm adult stage 5538.22 ipm adult stage

```
> 85 sza_5786.6_0 5786.685
> 86 sza_5857.6_0 5857.675
> 87 sza_5928.6_0 5928.665
> 88 sza_5999.6_0 5999.655
> 89 sza_6070.6_0 6070.645
> 90 sza_6141.6_0 6141.635
> 91 sza_6212.6_0 6212.625
> 92 sza_6283.6_0 6283.615
> 93 sza_6354.6_0 6354.605
> 94 sza_6425.5_0 6425.595
> 95 sza_6496.5_0 6496.585
> 96 sza_6567.5_0 6567.575
> 97 sza_6638.5_0 6638.565
> 98 sza_6709.5_0 6709.555
> 99 sza_6780.5_0 6780.545
> 100 sza_6851.5_0 6851.535
> 101 sza_6922.5_0 6922.525
> 102 sza_6993.5_0 6993.515
> 103 sza_7064.5_0 7064.505
```

5751.19
5822.18
5893.17
5964.16
6035.15
6106.14
6177.13
6248.12
6319.11
6390.10
6461.09
6532.08
6603.07
6674.06
6745.05
6816.04
$6887.03 \quad 6958.02$ ipm adult stage
$6958.02 \quad 7029.01 \mathrm{ipm}$ adult stage
$7029.01 \quad 7100.00$ ipm adult stage

### 13.1.1 Incorporating vital rate functions used in IPMs and fbMPMs

To be useful, publications involving IPMs need to show the parameterizations of their kernels, which are generally composed of vital rate models. These parameterizations vary in complexity, as different authors have different preferences for developing vital rate models. Some choose approaches involving splines or general additive modeling - these approaches produce potentially complex patterns and are difficult to characterize except via saved data structures describing the localized relationships between independent terms and response (for an excellent example of a well-done non-linear IPM, see Garcia et al., 2011). Other approaches utilize linear modeling, and may involve generalized linear modeling (GLM) or generalized linear mixed modeling (GLMM) (e.g., Dahlgren and Ehrlén, 2011; Shefferson et al., 2014). Package lefko3 is currently set up to allow imports of GLM and GLMM models. We anticipate developing methods to import spline-based kernels in the near future.

Let's assume that our IPM is composed of a kernel that is itself composed of four vital rates: survival probability, sprouting probability, size transition, and fecundity. The kernel for a typical sprouting individual would be given as the following.

$$
\begin{equation*}
K\left(x_{j}, x_{i}\right)=s\left(x_{i}, t\right) r\left(x_{j}, t+1\right) g\left(x_{j}, x_{i}\right)+f\left(x_{i}, t\right) \tag{13.1}
\end{equation*}
$$

Here, $s\left(x_{i}, t\right)$ is the survival of an individual of discretized size $x_{i}$ at time $t$ to time $t+1, r\left(x_{j}, t+1\right)$ is the sprouting probability of an individual in discretized size $x_{j}$ at time $t+1$ that has survived from time $t, g\left(x_{j}, x_{i}\right)$ is the probability of size transition from discretized size $x_{i}$ at time $t$ to discretized size $x_{j}$ in time $t+1$ assuming survival and sprouting, and $f\left(x_{i}, t\right)$ is the average offspring production in time $t+1$ of an individual in discretized size $x_{i}$ at time $t$. For a vegetatively dormant individual, the kernel differs because a lack of sprouting prevents any size transition or offspring production from occurring, as in the following.

$$
\begin{equation*}
K\left(x_{j}, x_{i}\right)=s\left(x_{i}, t\right)\left(1-r\left(x_{j}, t+1\right)\right) \tag{13.2}
\end{equation*}
$$

Next we need to scour the publication to find the definitions of the vital rate models defining the kernels. In our case, this means finding the slope coefficients and response distributions for all vital rate models used, including survival probability, sprouting probability, size transition, fecundity, etc. The probability distributions, as shown in our IPM chapter (Chapter 7), should be binomial for vital rates
such as survival, observation status (sprouting), reproductive status, and maturity status. They could be Gaussian, gamma, Poisson, or negative binomial for size and fecundity, or might be zero-truncated or zero-inflated versions of the Poisson or negative binomial. We also need to know whether juvenile vital rate models will be used, and what those distributions correspond to.

Chapter 7 provides model summaries (just preceding section 7.2.4). For this example, following the model summaries in the IPM chapter, we see that we need survival, observation status, size transition, and fecundity, and that we will also use juvenile survival, juvenile observation status, and juvenile size transition to model the seedling class. The modeling approach is the generalized linear mixed model, with year as a random categorical factor.

Let's start off by building a skeleton vrm_input object. Note that we input only the years capable of being transitioned from in the dataset, which means the three years 1988, 1989, and 1990. We will make this object a bit bigger than before, because we will allow fecundity to be zero-inflated (setting zi $=$ TRUE will add seven new columns governing the zero-inflation binomial model for all parameters capable of being zero-inflated). We will also set use.juv = TRUE to change some defaults to allow the use of juvenile transitions. Here is the code to produce our skeleton object.


```
\begin{tabular}{llllllllllll}
\(>\) & 14 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\(>15\) & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\(>16\) & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\(>17\) & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{tabular}
\begin{tabular}{|c|c|c|c|c|c|}
\hline > 1 & 0 & 0 & 0 & 0 & 0 \\
\hline \(>2\) & 0 & 0 & 0 & 0 & 0 \\
\hline \(>3\) & 0 & 0 & 0 & 0 & 0 \\
\hline \(>4\) & 0 & 0 & 0 & 0 & 0 \\
\hline \(>5\) & 0 & 0 & 0 & 0 & 0 \\
\hline \(>6\) & 0 & 0 & 0 & 0 & 0 \\
\hline > 7 & 0 & 0 & 0 & 0 & 0 \\
\hline > 8 & 0 & 0 & 0 & 0 & 0 \\
\hline \(>9\) & 0 & 0 & 0 & 0 & 0 \\
\hline > 10 & 0 & 0 & 0 & 0 & 0 \\
\hline > 11 & 0 & 0 & 0 & 0 & 0 \\
\hline \(>12\) & 0 & 0 & 0 & 0 & 0 \\
\hline > 13 & 0 & 0 & 0 & 0 & 0 \\
\hline > 14 & 0 & 0 & 0 & 0 & 0 \\
\hline > 15 & 0 & 0 & 0 & 0 & 0 \\
\hline > 16 & 0 & 0 & 0 & 0 & 0 \\
\hline > 17 & 0 & 0 & 0 & 0 & 0 \\
\hline
\end{tabular}
> $year_frame
> years surv obs sizea sizeb sizec repst fec jsurv jobs jsizea jsizeb jsizec
\begin{tabular}{llllllllllllll}
\(>\) & 1988 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{tabular}
\begin{tabular}{llllllllllllll}
\(>\) & 2 & 1989 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\(>\) & 3 & 1990 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0
\end{tabular}
> jrepst jmatst sizea_zi sizeb_zi sizec_zi fec_zi jsizea_zi jsizeb_zi jsizec_zi
> 1 
> 2 
>
> $patch_frame
    patches surv obs sizea sizeb sizec repst fec jsurv jobs jsizea jsizeb jsizec
1 1 1 0 0 0 0rrlllllllllllll
    jrepst jmatst sizea_zi sizeb_zi sizec_zi fec_zi jsizea_zi jsizeb_zi jsizec_zi
```



```
    $group2_frame
        groups surv obs sizea sizeb sizec repst fec jsurv jobs jsizea jsizeb jsizec
    1 0
    jrepst jmatst sizea_zi sizeb_zi sizec_zi fec_zi jsizea_zi jsizeb_zi jsizec_zi
1 [llllllllll
$group1_frame
    groups surv obs sizea sizeb sizec repst fec jsurv jobs jsizea jsizeb jsizec
1 0}0
        jrepst jmatst sizea_zi sizeb_zi sizec_zi fec_zi jsizea_zi jsizeb_zi jsizec_zi
1 0
>
```

```
> $dist_frame
> response dist
> surv binom
> 2 obs constant
> 3 sizea gaussian
> 4 sizeb constant
> 5 sizec constant
> 6 repst constant
> 7 fec negbin
> 8 jsurv binom
> 9 jobs constant
> 10 jsizea gaussian
> 11 jsizeb constant
> 12 jsizec constant
> 13 jrepst constant
> 14 jmatst constant
>
> $st_frame
> surv obs sizea sizeb sizec repst fec jsurv jobs jsizea jsizeb
```



```
>
>
> attr(,"class")
[1] "vrm_input"
```

Now that we have our skeleton vrm_input object, let's take a look at the vital rate models. In a typical publication, the authors might present equations showing the linear relationships among terms. Alternatively, they may present some of the output from modeling, giving us the slope coefficients. Since chapter 7 includes the latter, let's instead assume that we have a paper that presents the vital rate models as equations. Below, we see how the survival and sprouting models might be presented, using the real estimated terms.

$$
\begin{gather*}
\operatorname{logit}\left(s\left(x_{i}, t\right)\right)=2.32571+0.00109 \operatorname{size}(t)+\operatorname{year}(t)+\operatorname{indiv}(i)  \tag{13.3}\\
\operatorname{logit}\left(r\left(x_{j}, t+1\right)\right)=2.230+\operatorname{year}(t)+\operatorname{indiv}(i) \tag{13.4}
\end{gather*}
$$

In these models, we see that both are binomial models using the logit link. Both have y-intercepts (2.32571 in the case of survival, and 2.230 in the case of sprouting). The survival model involves a relationship with size in time $t$, and that relationship is linear with a slope of 0.00109 . Both equations include categorical values for year in time $t$ and individual, because both survival and sprouting probabilities were estimated as mixed models with year in time $t$ and individual as random terms. Let's input all of the main terms into our skeleton vrm_input object, particularly getting the y-intercepts and slope coefficients into the appropriate parts of the vrm_frame. Let's also change the distribution of the sprouting model from constant (the current setting) to binom.

```
int.elem <- which(lath_vrm$vrm_frame$main_effect_1 == "intercept")
size2.elem <- which(lath_vrm$vrm_frame$main_effect_1 == "size2")
lath_vrm$vrm_frame$surv[int.elem] <- 2.32571
lath_vrm$vrm_frame$surv[size2.elem] <- 0.00109
```

lath_vrm\$vrm_frame\$obs[int.elem] <- 2.230
lath_vrm\$dist_frame\$dist[2] <- "binom"



We have added the fixed main effects to our survival and sprouting models. Next, we will add the appropriate year terms. Year is a random term in both cases, meaning that the average effect of year has actually already been absorbed by the y-intercept and the mean of the year terms should be approximately zero. So, if we cannot find these terms in the paper, then we can simply assume it is 0 , or we can produce random numbers if we have information on the variance of the year term in the model. In our case, we see in the output for the survival and sprouting models in the IPM chapter that the standard deviation of the year term is 0 , meaning that these coefficients were inestimable under the mixed structure used. So, we will skip adding these terms here. Because we are not interested in predicting individual survival probabilities, we will also not incorporate any individual terms.

Let's move on to size. Our size model has a Gaussian response and so uses the identity link. Thus, our predicted size in time $t+1$ is given by the equation below.

$$
\begin{equation*}
E\left(\operatorname{size}\left(x_{j}, t+1\right)\right)=164.0695+0.6211 \operatorname{size}\left(z_{i}, t\right)+\operatorname{year}(t)+\operatorname{indiv}(i) \tag{13.5}
\end{equation*}
$$

The probability of becoming size $j$ in time $t+1$ assuming a Gaussian distribution is the following.

$$
\begin{equation*}
g\left(x_{j}, x_{i}\right)=\frac{1}{\sqrt{2 \pi} \sigma\left(x_{j}\right)} e^{-E\left(\operatorname{size}\left(x_{j}, t+1\right)\right)} \tag{13.6}
\end{equation*}
$$

Our size model includes year terms, which are 96.3244, -240.8036 , and 144.4792 for years 1988 , 1989 , and 1990. We also see that $\sigma=503.6167$, which is shown as the standard deviation of the residual component in the random effects section of the model summary output. We will add these terms below. Note that primary size is set to the Gaussian distribution by default.

```
lath_vrm$vrm_frame$sizea[int.elem] <- 164.0695
lath_vrm$vrm_frame$sizea[size2.elem] <- 0.6211
lath_vrm$year_frame$sizea <- c(96.3244, -240.8036, 144.4792)
lath_vrm$st_frame[3] <- 503.6167
```




The next model to add is the fecundity model. This will be a zero-inflated negative binomial mixed model. Zero-inflation models are actually composed of two linear models - a binomial model governing the occurrence of zeros, and a second model with the target distribution covering all non-zeros (in this case, a negative binomial with a log link). So, we will need to parameterize both. Here are the two models, as shown in the model output summary in chapter 7 .

$$
\begin{gather*}
\operatorname{logit}\left(f\left(x_{i}\right)=0\right)=6.252765-0.007313 \operatorname{size}(t)+\operatorname{year}(t)+\operatorname{indiv}(i)  \tag{13.7}\\
\log \left(f\left(x_{i}\right)>0\right)=1.517+\operatorname{year}(t)+\operatorname{indiv}(i) \tag{13.8}
\end{gather*}
$$

The dispersion parameter for the negative binomial is $\theta$, and we see in the IPM chapter that this is given as 0.2342114 . The year terms for the zero-inflation binomial model are $3.741475 \times 10^{-7}$, $-7.804715 \times 10^{-8}$, and $-2.533755 \times 10^{-7}$ for 1988,1989 , and 1990 , respectively. The year terms for the conditional model (governing non-zero responses) are $-0.41749627,0.51421684$, and -0.07964038 , respectively. Let's incorporate all of these values.

```
lath_vrm$vrm_frame$fec[int.elem] <- 1.517
lath_vrm$vrm_frame$fec_zi[int.elem] <- 6.252765
lath_vrm$vrm_frame$fec_zi[size2.elem] <- -0.007313
lath_vrm$year_frame$fec <- c(-0.41749627, 0.51421684, -0.07964038)
lath_vrm$year_frame$fec_zi <- c(3.741475e-07, -7.804715e-08, -2.533755e-07)
lath_vrm$st_frame[7] <- 0.2342114
lath_vrm
> $vrm_frame
> main_effect_1
    main_1_defined surv obs sizea sizeb
> 1 intercept
    y-intercept 2.32571 2.23 164.0695 0
```

| $>2$ |  | size2 |  |  | size | zea in t | time t | 0.00109 | 90.00 |  | 0.6211 | 0 |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| $>3$ |  | size1 |  |  | sizea | in tim | me t-1 | 0.00000 | 00.00 |  | 0.0000 | 0 |
| $>4$ |  | sizeb2 |  |  | size | zeb in t | time t | t 0.00000 | 00.00 |  | 0.0000 | 0 |
| $>5$ |  | sizeb1 |  |  | sizeb | in tim | me t-1 | 0.00000 | 00.00 |  | 0.0000 | 0 |
| $>6$ |  | sizec2 |  |  | size | zec in t | time t | t 0.00000 | 00.00 |  | 0.0000 | 0 |
| $>7$ |  | sizec1 |  |  | sizec | in tim | me t-1 | 0.00000 | 00.00 |  | 0.0000 | 0 |
| $>8$ |  | repst2 |  | reproducti | ive statu | us in t | time t | 0.00000 | 0.00 |  | 0.0000 | 0 |
| $>9$ |  | repst1 |  | reproductive | e status | in tim | me t-1 | 0.00000 | 0.00 |  | 0.0000 | 0 |
| > 10 |  | age |  |  |  | age in t | time t | 0.00000 | 0.00 |  | 0.0000 | 0 |
| > 11 |  | density |  |  | densit | ity in t | time t | 0.00000 | 0.00 |  | 0.0000 | 0 |
| > 12 |  | indcova2 |  | ndividual cova | ovariate | a in t | time t | t 0.00000 | 0.00 |  | 0.0000 | 0 |
| $>13$ |  | indcova1 | 1 indi | ividual cova | ariate a | a in tim | me t-1 | 0.00000 | 00.00 |  | 0.0000 | 0 |
| > 14 |  | indcovb2 |  | ndividual cov | ovariate | b in t | time t | 0.00000 | 0.00 |  | 0.0000 | 0 |
| > 15 |  | indcovb1 | 1 indi | ividual cova | ariate b | b in time | me t-1 | 0.00000 | 00.00 |  | 0.0000 | 0 |
| > 16 |  | indcovc2 |  | ndividual cov | ovariate | c in t | time t | t 0.00000 | 0.00 |  | 0.0000 | 0 |
| > 17 |  | indcovc1 | 1 indi | ividual cova | ariate c | c in tim | me t-1 | 0.00000 | 00.00 |  | 0.0000 | 0 |
| > | sizec | repst | $f e c$ | jsurv jobs | jsizea | jsizeb | jsize | c jrepst | t jmat |  | sizea_zi |  |
| $>1$ | 0 | 01 | 1.517 | 00 | 0 | 0 |  | $0 \quad 0$ | 0 | 0 | 0 |  |
| $>2$ | 0 | 00 | 0.000 | 00 | 0 | 0 |  | 00 | 0 | 0 | 0 |  |
| $>3$ | 0 | 00 | 0.000 | 00 | 0 | 0 |  | 0 | 0 | 0 | 0 |  |
| $>4$ | 0 | 00 | 0.000 | 00 | 0 | 0 |  | 00 | 0 | 0 | 0 |  |
| $>5$ | 0 |  | 0.000 | 00 | 0 | 0 |  | 00 | 0 | 0 | 0 |  |
| $>6$ | 0 |  | 0.000 | 00 | 0 | 0 |  | 00 | 0 | 0 | 0 |  |
| $>7$ | 0 |  | 0.000 | 00 | 0 | 0 |  | 00 | 0 | 0 | 0 |  |
| $>8$ | 0 |  | 0.000 | 00 | 0 | 0 |  | 00 | 0 | 0 | 0 |  |
| > 9 | 0 |  | 0.000 | 00 | 0 | 0 |  | 00 | 0 | 0 | 0 |  |
| > 10 | 0 |  | 0.000 | 00 | 0 | 0 |  | 00 | 0 | 0 | 0 |  |
| > 11 | 0 |  | 0.000 | 00 | 0 | 0 |  | 00 | 0 | 0 | 0 |  |
| > 12 | 0 |  | 0.000 | 00 | 0 | 0 |  | 00 | 0 | 0 | 0 |  |
| $>13$ | 0 |  | 0.000 | 00 | 0 | 0 |  | 00 | 0 | 0 | 0 |  |
| > 14 | 0 |  | 0.000 | 00 | 0 | 0 |  | 00 | 0 | 0 | 0 |  |
| > 15 | 0 |  | 0.000 | 00 | 0 | 0 |  | 00 | 0 | 0 | 0 |  |
| > 16 | 0 |  | 0.000 | 00 | 0 | 0 |  | 00 | 0 | 0 | 0 |  |
| > 17 | 0 | 00 | 0.000 | 00 | 0 | 0 |  | 0 | 0 | 0 | 0 |  |
|  | sizeb_ | _zi size | ec_zi | fec_zi | jsizea_zi | zi jsize | eb_zi | jsizec_z |  |  |  |  |
| $>1$ |  | 0 | 0 | 6.252765 |  | 0 | 0 |  | 0 |  |  |  |
| $>2$ |  | 0 | 0 | -0.007313 |  | 0 | 0 |  | 0 |  |  |  |
| $>3$ |  | 0 | 0 | 0.000000 |  | 0 | 0 |  | 0 |  |  |  |
| $>4$ |  | 0 | 0 | 0.000000 |  | 0 | 0 |  | 0 |  |  |  |
| $>5$ |  | 0 | 0 | 0.000000 |  | 0 | 0 |  | 0 |  |  |  |
| $>6$ |  | 0 | 0 | 0.000000 |  | 0 | 0 |  | 0 |  |  |  |
| $>7$ |  | 0 | 0 | 0.000000 |  | 0 | 0 |  | 0 |  |  |  |
| $>8$ |  | 0 | 0 | 0.000000 |  | 0 | 0 |  | 0 |  |  |  |
| > 9 |  | 0 | 0 | 0.000000 |  | 0 | 0 |  | 0 |  |  |  |
| > 10 |  | 0 | 0 | 0.000000 |  | 0 | 0 |  | 0 |  |  |  |
| > 11 |  | 0 | 0 | 0.000000 |  | 0 | 0 |  | 0 |  |  |  |
| > 12 |  | 0 | 0 | 0.000000 |  | 0 | 0 |  | 0 |  |  |  |
| > 13 |  | 0 | 0 | 0.000000 |  | 0 | 0 |  | 0 |  |  |  |
| > 14 |  | 0 | 0 | 0.000000 |  | 0 | 0 |  | 0 |  |  |  |
| > 15 |  | 0 | 0 | 0.000000 |  | 0 | 0 |  | 0 |  |  |  |
| > 16 |  | 0 | 0 | 0.000000 |  | 0 | 0 |  | 0 |  |  |  |

```
llllllll
>
$year_frame
    years surv obs sizea sizeb sizec repst fec jsurv jobs jsizea
> 1 1988 0 0 9 96.3244 0 0 0 0
> 2 1989 0
> 3 1990 
> 1 0 0 0 0
> 2 0 0 0 0 0
> 3 0 0 0 0 0 0 0
jsizea_zi jsizeb_zi jsizec_zi
1 0 0 0
> 2 0
> 3 0 0
>
$patch_frame
    patches surv obs sizea sizeb sizec repst fec jsurv jobs jsizea jsizeb jsizec
1
    jrepst jmatst sizea_zi sizeb_zi sizec_zi fec_zi jsizea_zi jsizeb_zi jsizec_zi
1 0
$group2_frame
    groups surv obs sizea sizeb sizec repst fec jsurv jobs jsizea jsizeb jsizec
```



```
        jrepst jmatst sizea_zi sizeb_zi sizec_zi fec_zi jsizea_zi jsizeb_zi jsizec_zi
    1 [llllllllll
    $group1_frame
        groups surv obs sizea sizeb sizec repst fec jsurv jobs jsizea jsizeb jsizec
1 0}0
    jrepst jmatst sizea_zi sizeb_zi sizec_zi fec_zi jsizea_zi jsizeb_zi jsizec_zi
1 0
    $dist_frame
        response dist
    1 surv binom
2 obs binom
3 sizea gaussian
4 sizeb constant
5 sizec constant
6 repst constant
7 fec negbin
8 jsurv binom
9 jobs constant
10 jsizea gaussian
11 jsizeb constant
12 jsizec constant
> 13 jrepst constant
> 14 jmatst constant
>
```

```
\begin{tabular}{lrrrrr}
\(>\) & & & & & \\
\(>\) & Sst_frame & surv & obs & sizea & sizeb \\
\(>\) & 1.0000000 & 1.0000000 & 503.6167000 & 1.0000000 & 1.0000000 \\
\(>\) & fec & jsurv & jobs & jsizea & jsizeb \\
\(>\) & 0.2342114 & 1.0000000 & 1.0000000 & 1.0000000 & 1.0000000 \\
\(>\) & jrepst & jmatst & & 1.0000000 \\
\(>\) & 1.0000000 & 1.0000000 & & & \\
\(>\) & & & & & \\
\(>\) & & & & & \\
\(>\) & attr (,"class") & & & &
\end{tabular}
```

Next we will need to input the associated parameters for any other model included in the IPM. Particularly, we know that vital rate models were also estimated for seedlings, and that these models were incorporated as juvenile vital rate models. The juvenile vital rates include survival, sprouting, and size transition. The equations for these models are as follows.

$$
\begin{gather*}
\operatorname{logit}\left(s_{j u v}\left(x_{i}, t\right)\right)=1.03+\operatorname{year}(t)+\operatorname{indiv}(i)  \tag{13.9}\\
\operatorname{logit}\left(r_{j u v}\left(x_{j}, t+1\right)\right)=10.390+\operatorname{year}(t)+\operatorname{indiv}(i)  \tag{13.10}\\
E_{\text {juv }}\left(\operatorname{size}\left(x_{j}, t+1\right)\right)=3.0559+0.8482 \operatorname{size}(t)+\operatorname{year}(t)+\operatorname{indiv}(i) \tag{13.11}
\end{gather*}
$$

Together with the residual $\sigma$ for the size model and year terms for sprouting and size, let's add all of these terms to our vrm_input object.

```
lath_vrm$vrm_frame$jsurv[int.elem] <- 1.03
lath_vrm$vrm_frame$jobs[int.elem] <- 10.390
lath_vrm$vrm_frame$jsizea[int.elem] <- 3.0559
lath_vrm$vrm_frame$jsizea[size2.elem] <- 0.8482
lath_vrm$st_frame[10] <- 5.831
lath_vrm$year_frame$jobs <- c(-0.7459843, 0.6118826, -0.9468618)
lath_vrm$year_frame$jsizea <- c(0.5937962, 1.4551236, -2.0489198)
lath_vrm$dist_frame$dist[9] <- "binom"
lath_vrm
> $vrm_frame
> main_effect_1
> 1 intercept
> 2 size2
> 3 size1
> 4 sizeb2
> 5 sizeb1
> 6 sizec2
> 7 sizec1
> 8 repst2
> 9 repst1
    age
> 11 density
```

```
            main_1_defined surv obs sizea sizeb
```

            main_1_defined surv obs sizea sizeb
                                y-intercept 2.32571 2.23 164.0695 0
                                y-intercept 2.32571 2.23 164.0695 0
                            sizea in time t 0.00109 0.00 0.6211 0
                            sizea in time t 0.00109 0.00 0.6211 0
                            sizea in time t-1 0.00000 0.00 0.0000 0
                            sizea in time t-1 0.00000 0.00 0.0000 0
                            sizeb in time t 0.00000 0.00 0.0000 0
                            sizeb in time t 0.00000 0.00 0.0000 0
        sizeb in time t-1 0.00000 0.00 0.0000 0
        sizeb in time t-1 0.00000 0.00 0.0000 0
    sizec in time t 0.00000 0.00 0.0000 0
    sizec in time t 0.00000 0.00 0.0000 0
        sizec in time t-1 0.00000 0.00 0.0000 0
        sizec in time t-1 0.00000 0.00 0.0000 0
        reproductive status in time t 0.00000 0.00 0.0000 0
        reproductive status in time t 0.00000 0.00 0.0000 0
            reproductive status in time t-1 0.00000 0.00 0.0000 0
            reproductive status in time t-1 0.00000 0.00 0.0000 0
            age in time t 0.00000 0.00 0.0000 0
            age in time t 0.00000 0.00 0.0000 0
        density in time t 0.00000 0.00 0.0000 0
    ```
        density in time t 0.00000 0.00 0.0000 0
```



```
> 3-2.0489198 0-0 0 0 0
> fec_zi jsizea_zi jsizeb_zi jsizec_zi
> 1 3.741475e-07 0
> 2 -7.804715e-08 0 0 0
> 3-2.533755e-07 0 0
>
> $patch_frame
> patches surv obs sizea sizeb sizec repst fec jsurv jobs jsizea jsizeb jsizec
> 1 1 1 0 0 0 0rlllllllllllll
> jrepst jmatst sizea_zi sizeb_zi sizec_zi fec_zi jsizea_zi jsizeb_zi jsizec_zi
```



```
$group2_frame
groups surv obs sizea sizeb sizec repst fec jsurv jobs jsizea jsizeb jsizec
```



```
    jrepst jmatst sizea_zi sizeb_zi sizec_zi fec_zi jsizea_zi jsizeb_zi jsizec_zi
1 0
$group1_frame
        groups surv obs sizea sizeb sizec repst fec jsurv jobs jsizea jsizeb jsizec
1 0
        jrepst jmatst sizea_zi sizeb_zi sizec_zi fec_zi jsizea_zi jsizeb_zi jsizec_zi
1 0 0 0 0 0 0
$dist_frame
        response dist
1 surv binom
> 2 obs binom
> 3 sizea gaussian
> 4 sizeb constant
> 5 sizec constant
6 repst constant
> 7 fec negbin
> 8 jsurv binom
> 9 jobs binom
> 10 jsizea gaussian
> 11 jsizeb constant
> 12 jsizec constant
> 13 jrepst constant
> 14 jmatst constant
>
> $st_frame
> surv sizea sizeb sizec repst
> 1.0000000 1.0000000 503.6167000 1.0000000 1.0000000 1.0000000
> fec jrarv jobs jsizea jsizeb jsizec
> 0.2342114 1.0000000 1.0000000 5.8310000 1.0000000 1.0000000
> >
>
> attr(,"class")
> [1] "vrm_input"
```


## Voilà!

### 13.1.2 Fitting vital rate models to matrix estimation functions

At this point, we have created all of our main IPM vital rate model inputs. However, there are still a few missing pieces that we need to fit in order to create the IPM properly. Our next step is to include instructions that tell lefko3 how the vital rate models fit together. Let's see how this works.

By default, elements in survival-transition matrices created by functions flefko2(), flefko3(), aflefko2(), and fleslie() are estimated via the following two kernels, the first for adults and the second for juveniles.

$$
\begin{equation*}
e_{j, i}=\text { surv }_{\text {kernel }} \text { obs } k_{\text {kernel }} \text { sizea }_{\text {kernel }} \text { sizeb }_{\text {kernel }} \text { sizec }_{\text {kernel }} \text { repr }_{\text {kernel }} \tag{13.12}
\end{equation*}
$$

Here, $e_{j, i}$ refers to the element at the $j_{\text {th }}$ row and the $\mathrm{i}_{\text {th }}$ column. The terms to the right of the equal sign refer to the values developed by kernels representing different vital rates. So, surv kernel $^{\text {is }}$ the kernel developing the probability of survival, obs $s_{\text {kernel }}$ is the kernel developing the probability of observation (or its complement), sizea kernel is the probability of size transition in the primary size variable, sizeb $b_{\text {kernel }}$ is the probability of size transition in the secondary size variable, sizec ${ }_{\text {kernel }}$ is the probability of size transition in the tertiary size variable, and repst ${ }_{\text {kernel }}$ is the probability of becoming reproductive or its complement. The juvenile kernel is composed of similar vital rate kernels, but also includes jmatst ${ }_{\text {kernel }}$, which is the probability of becoming mature or its complement.

Package lefko3 uses the vital rate kernels above to produce values in essentially all cases, and so vital rate kernel values can shift to a constant value of 1 if a vital rate is not used. There are also circumstances in which vital rates may be fixed to 0 or even other constants, though these are relatively rare (this may occur is a juvenile size class is not utilized, because then juvenile vital rates should generally equal zero during matrix creation). The key is to fix unused vital rates to constant values of 1 . We do this by changing the y-intercept of unused vital rate models to exactly 1 , and changing the distribution listed for the vital rate in dist_frame as constant. Note that we do not need to bother doing this with zero-inflation components of unused vital rates. In our case, all unused vital rate models are already set to constant, so we will just change the intercepts to 1 .

```
lath_vrm$vrm_frame$sizeb[1] <- 1
lath_vrm$vrm_frame$sizec[1] <- 1
lath_vrm$vrm_frame$repst[1] <- 1
lath_vrm$vrm_frame$jsizeb[1] <- 1
lath_vrm$vrm_frame$jsizec[1] <- 1
lath_vrm$vrm_frame$jrepst[1] <- 1
lath_vrm$vrm_frame$jmatst[1] <- 1
lath_vrm
> $vrm_frame
> main_effect_1
> 1 intercept
> 2 size2
> 3 size1
> 4 sizeb2
> 5 sizeb1
> 6 sizec2
```

| main_1_defined | surv | obs | sizea | sizeb |
| ---: | ---: | ---: | ---: | ---: |
| y-intercept | 2.32571 | 2.23 | 164.0695 | 1 |
| sizea in time t | 0.00109 | 0.00 | 0.6211 | 0 |
| sizea in time t-1 | 0.00000 | 0.00 | 0.0000 | 0 |
| sizeb in time t | 0.00000 | 0.00 | 0.0000 | 0 |
| sizeb in time t-1 | 0.00000 | 0.00 | 0.0000 | 0 |
| sizec in time t | 0.00000 | 0.00 | 0.0000 | 0 |

602 CHAPTER 13. FURTHER ISSUES II: IMPORTING IPMS AND FUNCTION-BASED MPMS


```
> 2 1989 0 0 0-240.8036 0
> 3 1990 0 0 144.4792 0 0 0 0 -0.07964038 0-0.9468618
> jsizea jsizeb jsizec jrepst jmatst sizea_zi sizeb_zi sizec_zi
> 1 0.5937962 
> 2 1.4551236 
> 3-2.0489198 0
> fec_zi jsizea_zi jsizeb_zi jsizec_zi
> 1 3.741475e-07 0 0
> 2-7.804715e-08 0 0
> 3-2.533755e-07 0 0
> $patch_frame
> patches surv obs sizea sizeb sizec repst fec jsurv jobs jsizea jsizeb jsizec
> 1 1 1 0lllllllllllllll
> jrepst jmatst sizea_zi sizeb_zi sizec_zi fec_zi jsizea_zi jsizeb_zi jsizec_zi
1 0
$group2_frame
        groups surv obs sizea sizeb sizec repst fec jsurv jobs jsizea jsizeb jsizec
1 0
    jrepst jmatst sizea_zi sizeb_zi sizec_zi fec_zi jsizea_zi jsizeb_zi jsizec_zi
1 0
$group1_frame
    groups surv obs sizea sizeb sizec repst fec jsurv jobs jsizea jsizeb jsizec
```



```
> jrepst jmatst sizea_zi sizeb_zi sizec_zi fec_zi jsizea_zi jsizeb_zi jsizec_zi
1 0
$dist_frame
        response dist
1 surv binom
> 2 obs binom
> 3 sizea gaussian
> 4 sizeb constant
> 5 sizec constant
> 6 repst constant
> 7 fec negbin
> 8 jsurv binom
> 9 jobs binom
> 10 jsizea gaussian
> 11 jsizeb constant
> 12 jsizec constant
> 13 jrepst constant
> 14 jmatst constant
>
> $st_frame
> surv sizea sizeb sizec repst
> 1.0000000 1.0000000 503.6167000 1.0000000 1.0000000 1.0000000
> fec jsurv jobs jsizea jsizeb jsizec
> 0.2342114 1.0000000 1.0000000 5.8310000 1.0000000 1.0000000
```

```
> jrepst jmatst
> 1.0000000 1.0000000
>
> attr(,"class")
> [1] "vrm_input"
```

Next, we will add supplemental information not covered by IPM vital rate models. Let's add that portion now.

```
lathsupp2 <- supplemental(stage3 = c("Sd", "Sdl", "Sd", "Sdl"),
    stage2 = c("Sd", "Sd", "rep", "rep"),
    givenrate = c(0.345, 0.054, NA, NA),
    multiplier = c(NA, NA, 0.345, 0.054),
    type = c(1, 1, 3, 3), stageframe = lathframeipm, historical = FALSE)
lathsupp2
> stage3 stage2 stage1 age2 eststage3 eststage2 eststage1 estage2 givenrate
```



```
> Sdl Sd <NA> NA <NA> <NA> <NA> NA 0.054
> Sd rep <NA> NA <NA> <NA> <NA> NA NA
>4 Sdl rep <NA> NA <NA> <NA> <NA> NA NA
> multiplier convtype convtype_t12
> 1 NA 1 1
> 2 NA 1 1
> 3 0.345 3 1
>4 0.054 3
```

Finally, let's build our IPM using the flefko2() function.

```
lathmat2_importipm <- flefko2(stageframe = lathframeipm, modelsuite = lath_vrm,
    supplement = lathsupp2, reduce = FALSE)
summary(lathmat2_importipm)
>
> This ahistorical lefkoMat object contains 3 matrices.
>
> Each matrix is square with }103\mathrm{ rows and columns, and a total of 10609 elements.
> A total of 26926 survival transitions were estimated, with 8975.333 per matrix.
> A total of }600\mathrm{ fecundity transitions were estimated, with 200 per matrix.
> This lefkoMat object covers 1 population, 1 patch, and 3 time steps.
>
> The dataset contains a total of 0 unique individuals and 0 unique transitions.
>
> Survival probability sum check (each matrix represented by column in order):
> [,1] [,2] [,3]
> Min. 0.000 0.000 0.000
> 1st Qu. 0.979 0.956 0.980
> Median 0.998 0.998 0.998
> Mean 0.951 0.922 0.954
> 3rd Qu. 1.000 1.000 1.000
> Max. 1.000 1.000 1.000
```

Users exploring the output and comparing against the original published IPM will notice that the element values are almost the same, but differ by tiny amounts (often on the order of $10_{-5}$ or smaller).

These small differences occur because of the rounding that happens when authors publish their models, and are not a cause of concern nor will they cause dramatic differences in inference.

### 13.2 Importing complex IPMs and fbMPMs

Users may wish to import more complex structures using further terms or interactions. The approach above works in these cases, as well. For example, suppose that we wished to import the historical IPM for Lathyrus shown in the IPM chapter (Chapter 7). That IPM involves models with historical size and some two-way interactions. We can also produce imported IPMs for this more complicated case, but need to expand the vrm_input object by noting interactions = TRUE in the input to that function, as below (because this structure is quite large, we have suppressed the output for the final line).

| \$vrm_frame |  |  |  |
| :---: | :---: | :---: | :---: |
| > | main_effect_1 | main_1_defined | main_effect_2 |
| $>1$ | intercept | $y$-intercept |  |
| $>2$ | size2 | sizea in time t |  |
| $>3$ | size1 | sizea in time t-1 |  |
| $>4$ | sizeb2 | sizeb in time t |  |
| $>5$ | sizeb1 | sizeb in time t-1 |  |
| $>6$ | sizec2 | sizec in time t |  |
| $>7$ | sizec1 | sizec in time t-1 |  |
| $>8$ | repst2 | reproductive status in time t |  |
| $>9$ | repst1 | reproductive status in time t-1 |  |
| > 10 | age | age in time t |  |
| > 11 | density | density in time t |  |
| > 12 | indcova2 | individual covariate a in time t |  |
| > 13 | indcova1 | individual covariate a in time t-1 |  |
| > 14 | indcovb2 | individual covariate b in time t |  |
| > 15 | indcovb1 | individual covariate b in time t-1 |  |
| > 16 | indcovc2 | individual covariate c in time t |  |
| > 17 | indcovc1 | individual covariate c in time t-1 |  |
| > 18 | repst1 | reproductive status in time t-1 | repst2 |
| > 19 | size1 | sizea in time t-1 | size2 |
| > 20 | size1 | sizea in time t-1 | repst1 |
| > 21 | size2 | sizea in time t | repst2 |
| > 22 | size2 | sizea in time t | repst1 |
| > 23 | size1 | sizea in time t-1 | repst2 |
| > 24 | age | age in time t | size1 |
| > 25 | age | age in time t | size2 |
| > 26 | age | age in time t | repst1 |
| > 27 | age | age in time t | repst2 |
| > 28 | indcova2 | individual covariate a in time t | size2 |
| > 29 | indcovb2 | individual covariate $b$ in time $t$ | size2 |
| > 30 | indcovc2 | individual covariate c in time t | size2 |
| > 31 | indcova2 | individual covariate a in time t | repst2 |
| > 32 | indcovb2 | individual covariate b in time t | repst2 |
| > 33 | indcovc2 | individual covariate c in time t | repst2 |
| > 34 | indcova1 | individual covariate a in time t-1 | size1 |
| > 35 | indcovb1 | individual covariate b in time $\mathrm{t}-1$ | size1 |
| > 36 | indcovc1 | individual covariate c in time t-1 | size1 |
| > 37 | indcova1 | individual covariate a in time t-1 | repst1 |



| > 8 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| $>9$ | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| > 10 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| > 11 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| > 12 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| > 13 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| > 14 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| > 15 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| > 16 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| > 17 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| > 18 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| > 19 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| > 20 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| > 21 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| > 22 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| > 23 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| > 24 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| > 25 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| > 26 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| > 27 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| > 28 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| > 29 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| > 30 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| > 31 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| > 32 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| > 33 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| > 34 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| > 35 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| > 36 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| > 37 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| > 38 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| > 39 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| > 40 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| $>$ |  |  |  |  |  |  |  |  |  |  |
| > 1 |  |  | 0 |  |  |  |  |  |  |  |
| $>2$ |  |  | 0 |  |  |  |  |  |  |  |
| $>3$ |  |  | 0 |  |  |  |  |  |  |  |
| $>4$ |  |  | 0 |  |  |  |  |  |  |  |
| $>5$ |  |  | 0 |  |  |  |  |  |  |  |
| $>6$ |  |  | 0 |  |  |  |  |  |  |  |
| > 7 |  |  | 0 |  |  |  |  |  |  |  |
| $>8$ |  |  | 0 |  |  |  |  |  |  |  |
| > 9 |  |  | 0 |  |  |  |  |  |  |  |
| > 10 |  |  | 0 |  |  |  |  |  |  |  |
| > 11 |  |  | 0 |  |  |  |  |  |  |  |
| > 12 |  |  | 0 |  |  |  |  |  |  |  |
| > 13 |  |  | 0 |  |  |  |  |  |  |  |
| > 14 |  |  | 0 |  |  |  |  |  |  |  |
| > 15 |  |  | 0 |  |  |  |  |  |  |  |
| > 16 |  |  | 0 |  |  |  |  |  |  |  |
| > 17 |  |  | 0 |  |  |  |  |  |  |  |
| > 18 |  |  | 0 |  |  |  |  |  |  |  |

```
> 19 0 0 0
> 20 0 0
> 21 0 0 0
>22 0 0
>23 0 0
> 24
> 25
> 26
> 27
> 28
> 29
> 30
> 31
> 32
> 33
> 34
> 35
> 36
> 37
> 38 0
> 39 >
>40 cccc
> [ reached 'max' / getOption("max.print") -- omitted 88 rows ]
>
> $year_frame
> years surv obs sizea sizeb sizec repst fec jsurv jobs jsizea jsizeb jsizec
\begin{tabular}{llllllllllllll}
\(>\) & 1 & 1988 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{tabular}
\begin{tabular}{llllllllllllll}
\(>\) & 2 & 1989 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\(>\) & 3 & 1990 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0
\end{tabular}
> jrepst jmatst sizea_zi sizeb_zi sizec_zi fec_zi jsizea_zi jsizeb_zi jsizec_zi
> 1 0
> 2 
>
> $patch_frame
> patches surv obs sizea sizeb sizec repst fec jsurv jobs jsizea jsizeb jsizec
l 1 1 1 0rllllllllllllll
> jrepst jmatst sizea_zi sizeb_zi sizec_zi fec_zi jsizea_zi jsizeb_zi jsizec_zi
>1 [rlllllllll
> $group2_frame
> groups surv obs sizea sizeb sizec repst fec jsurv jobs jsizea jsizeb jsizec
> 1 0
> jrepst jmatst sizea_zi sizeb_zi sizec_zi fec_zi jsizea_zi jsizeb_zi jsizec_zi
> 1 0
>
> $group1_frame
> groups surv obs sizea sizeb sizec repst fec jsurv jobs jsizea jsizeb jsizec
> 1 0
> jrepst jmatst sizea_zi sizeb_zi sizec_zi fec_zi jsizea_zi jsizeb_zi jsizec_zi
> 1 [rlllllllll
```

```
> $dist_framerr
> 1 surv binom
> 2 obs constant
> 3 sizea gaussian
> 4 sizeb constant
> 5 sizec constant
>6 repst constant
> 7 fec negbin
> 8 jsurv binom
> 9 jobs constant
> 10 jsizea gaussian
> 11 jsizeb constant
> 12 jsizec constant
> 13 jrepst constant
> 14 jmatst constant
>
> $st_frame
> surv 
```



```
> jsizec jrepst jmatst
> 1 1 1 1
>
> attr(,"class")
> [1] "vrm_input"
```

Our new vrm_input object is bigger. The difference in size is a direct result of an increased number of rows in the vrm_frame element: previously, that data frame held 17 rows, but now it holds 128. The first 17 correspond to the main effects, as before, while the remaining 111 rows correspond to the two-way interactions.

Let's start off by changing the distributions, as before.

```
lath3_vrm$dist_frame$dist[2] <- "binom"
lath3_vrm$dist_frame$dist[9] <- "binom"
```

Now let's take a look at our vital rate models. The equations are as follows.

$$
\begin{equation*}
\operatorname{logit}\left(s\left(x_{i}, t\right)\right)=2.060+0.0009891 \operatorname{size}(t)+0.001531 \operatorname{size}(t-1)-0.0000004125 \operatorname{size}(t) \operatorname{size}(t-1)+y e a r(t)+\operatorname{indiv}(i) \tag{13.14}
\end{equation*}
$$

$$
\begin{equation*}
\operatorname{logit}\left(r\left(x_{j}, t+1\right)\right)=2.230+\operatorname{year}(t)+\operatorname{indiv}(i) \tag{13.15}
\end{equation*}
$$

$$
\begin{equation*}
E\left(\operatorname{size}\left(x_{j}, t+1\right)\right)=89.98+0.5954 \operatorname{size}(t)+0.3119 \operatorname{size}(t-1)-0.00009417 \operatorname{size}(t) \operatorname{size}(t-1)+y e a r(t)+i n d i v(i) \tag{13.16}
\end{equation*}
$$

$$
\begin{equation*}
\operatorname{logit}\left(f\left(x_{i}\right)=0\right)=6.252765-0.007313 \operatorname{size}(t)+\operatorname{year}(t)+\operatorname{indiv}(i) \tag{13.17}
\end{equation*}
$$

$$
\begin{align*}
& \log \left(f\left(x_{i}\right)>0\right)=1.517+\operatorname{year}(t)+\operatorname{indiv}(i)  \tag{13.18}\\
& \operatorname{logit}\left(s_{j u v}\left(x_{i}, t\right)\right)=1.03+\operatorname{year}(t)+\operatorname{indiv}(i) \tag{13.19}
\end{align*}
$$

$$
\begin{gather*}
\operatorname{logit}\left(r_{j u v}\left(x_{j}, t+1\right)\right)=10.390+\operatorname{year}(t)+\operatorname{indiv}(i)  \tag{13.20}\\
E_{j u v}\left(\operatorname{size}\left(x_{j}, t+1\right)\right)=3.0559+0.8482 \operatorname{size}(t)+\operatorname{year}(t)+\operatorname{indiv}(i) \tag{13.21}
\end{gather*}
$$

Some of these models are the same as in the ahistorical case, but a number are quite different. We will add these coefficients to the vrm_frame as before, but paying special attention to interaction terms. We will also add constant values of 1 for the intercepts of unused vital rate models. Note that we are suppressing the final output frmo the code below to prevent many pages of the book being taken up by a single object.

|  |  |  |  |
| :--- | ---: | ---: | ---: |
| $>$ | Svrm_frame |  |  |
| $>$ | main_effect_1 | main_1_defined main_effect_2 |  |
| $>$ | intercept | y-intercept |  |
| $>$ | 2 | size2 | sizea in time t |



| > 10 | 0 | 0.000 | 0.00 | 0.00 | 0.0000 | 0 | 0 | 0 | 0 | 0 | 0 |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| > 11 | 0 | 0.000 | 0.00 | 0.00 | 0.0000 | 0 | 0 | 0 | 0 | 0 | 0 |
| > 12 | 0 | 0.000 | 0.00 | 0.00 | 0.0000 | 0 | 0 | 0 | 0 | 0 | 0 |
| > 13 | 0 | 0.000 | 0.00 | 0.00 | 0.0000 | 0 | 0 | 0 | 0 | 0 | 0 |
| > 14 | 0 | 0.000 | 0.00 | 0.00 | 0.0000 | 0 | 0 | 0 | 0 | 0 | 0 |
| > 15 | 0 | 0.000 | 0.00 | 0.00 | 0.0000 | 0 | 0 | 0 | 0 | 0 | 0 |
| > 16 | 0 | 0.000 | 0.00 | 0.00 | 0.0000 | 0 | 0 | 0 | 0 | 0 | 0 |
| $>17$ | 0 | 0.000 | 0.00 | 0.00 | 0.0000 | 0 | 0 | 0 | 0 | 0 | 0 |
| > 18 | 0 | 0.000 | 0.00 | 0.00 | 0.0000 | 0 | 0 | 0 | 0 | 0 | 0 |
| > 19 | 0 | 0.000 | 0.00 | 0.00 | 0.0000 | 0 | 0 | 0 | 0 | 0 | 0 |
| > 20 | 0 | 0.000 | 0.00 | 0.00 | 0.0000 | 0 | 0 | 0 | 0 | 0 | 0 |
| > 21 | 0 | 0.000 | 0.00 | 0.00 | 0.0000 | 0 | 0 | 0 | 0 | 0 | 0 |
| > 22 | 0 | 0.000 | 0.00 | 0.00 | 0.0000 | 0 | 0 | 0 | 0 | 0 | 0 |
| > 23 | 0 | 0.000 | 0.00 | 0.00 | 0.0000 | 0 | 0 | 0 | 0 | 0 | 0 |
| > 24 | 0 | 0.000 | 0.00 | 0.00 | 0.0000 | 0 | 0 | 0 | 0 | 0 | 0 |
| > 25 | 0 | 0.000 | 0.00 | 0.00 | 0.0000 | 0 | 0 | 0 | 0 | 0 | 0 |
| > 26 | 0 | 0.000 | 0.00 | 0.00 | 0.0000 | 0 | 0 | 0 | 0 | 0 | 0 |
| > 27 | 0 | 0.000 | 0.00 | 0.00 | 0.0000 | 0 | 0 | 0 | 0 | 0 | 0 |
| > 28 | 0 | 0.000 | 0.00 | 0.00 | 0.0000 | 0 | 0 | 0 | 0 | 0 | 0 |
| > 29 | 0 | 0.000 | 0.00 | 0.00 | 0.0000 | 0 | 0 | 0 | 0 | 0 | 0 |
| > 30 | 0 | 0.000 | 0.00 | 0.00 | 0.0000 | 0 | 0 | 0 | 0 | 0 | 0 |
| > 31 | 0 | 0.000 | 0.00 | 0.00 | 0.0000 | 0 | 0 | 0 | 0 | 0 | 0 |
| > 32 | 0 | 0.000 | 0.00 | 0.00 | 0.0000 | 0 | 0 | 0 | 0 | 0 | 0 |
| > 33 | 0 | 0.000 | 0.00 | 0.00 | 0.0000 | 0 | 0 | 0 | 0 | 0 | 0 |
| > 34 | 0 | 0.000 | 0.00 | 0.00 | 0.0000 | 0 | 0 | 0 | 0 | 0 | 0 |
| > 35 | 0 | 0.000 | 0.00 | 0.00 | 0.0000 | 0 | 0 | 0 | 0 | 0 | 0 |
| > 36 | 0 | 0.000 | 0.00 | 0.00 | 0.0000 | 0 | 0 | 0 | 0 | 0 | 0 |
| > 37 | 0 | 0.000 | 0.00 | 0.00 | 0.0000 | 0 | 0 | 0 | 0 | 0 | 0 |
| > 38 | 0 | 0.000 | 0.00 | 0.00 | 0.0000 | 0 | 0 | 0 | 0 | 0 | 0 |
| > 39 | 0 | 0.000 | 0.00 | 0.00 | 0.0000 | 0 | 0 | 0 | 0 | 0 | 0 |
| > 40 | 0 | 0.000 | 0.00 | 0.00 | 0.0000 | 0 | 0 | 0 | 0 | 0 | 0 |

> sizec_zi fec_zi jsizea_zi jsizeb_zi jsizec_zi

| $>1$ | 0 | 6.252765 | 0 | 0 | 0 |
| :---: | :---: | :---: | :---: | :---: | :---: |
| $>2$ | 0 | -0.007313 | 0 | 0 | 0 |
| $>3$ | 0 | 0.000000 | 0 | 0 | 0 |
| $>4$ | 0 | 0.000000 | 0 | 0 | 0 |
| $>5$ | 0 | 0.000000 | 0 | 0 | 0 |
| $>6$ | 0 | 0.000000 | 0 | 0 | 0 |
| $>7$ | 0 | 0.000000 | 0 | 0 | 0 |
| $>8$ | 0 | 0.000000 | 0 | 0 | 0 |
| $>9$ | 0 | 0.000000 | 0 | 0 | 0 |
| > 10 | 0 | 0.000000 | 0 | 0 | 0 |
| > 11 | 0 | 0.000000 | 0 | 0 | 0 |
| > 12 | 0 | 0.000000 | 0 | 0 | 0 |
| $>13$ | 0 | 0.000000 | 0 | 0 | 0 |
| > 14 | 0 | 0.000000 | 0 | 0 | 0 |
| > 15 | 0 | 0.000000 | 0 | 0 | 0 |
| > 16 | 0 | 0.000000 | 0 | 0 | 0 |
| > 17 | 0 | 0.000000 | 0 | 0 | 0 |
| > 18 | 0 | 0.000000 | 0 | 0 | 0 |
| > 19 | 0 | 0.000000 | 0 | 0 | 0 |
| > 20 | 0 | 0.000000 | 0 | 0 | 0 |



```
> 1 surv binom
> 2 obs binom
> 3 sizea gaussian
>4 sizeb constant
> 5 sizec constant
> 6 repst constant
> 7 fec negbin
> 8 jsurv binom
> 9 jobs binom
> 10 jsizea gaussian
> 11 jsizeb constant
> 12 jsizec constant
> 13 jrepst constant
> 14 jmatst constant
>
> $st_frame
> surv obs sizea sizeb sizec repst fec jsurv jobs jsizea jsizeb
```



```
> jsizec jrepst jmatst
> 1-1 1
>
> attr(,"class")
> [1] "vrm_input"
```

Now we will add the appropriate year terms and the values of $\sigma$ and $\theta$, and the year coefficients.

```
lath3_vrm$st_frame[3] <- 480.4092
lath3_vrm$st_frame[7] <- 0.2342114
lath3_vrm$st_frame[10] <- 5.831241
lath3_vrm$year_frame$sizea <- c(193.44474, -277.14757, 83.70283)
lath3_vrm$year_frame$fec <- c(-0.41749627, 0.51421684, -0.07964038)
lath3_vrm$year_frame$fec_zi <- c(3.741475e-07, -7.804715e-08, -2.533755e-07)
lath3_vrm$year_frame$sizea <- c(193.44474, -277.14757, 83.70283)
lath3_vrm$year_frame$jobs <- c(-0.7459843, 0.6118826, -0.9468618)
lath_vrm$year_frame$jsizea <- c(0.5937962, 1.4551236, -2.0489198)
```

As before, we still need to supply supplemental information. We'll do that below.

```
lathsupp3 <- supplemental(stage3 = c("Sd","Sd","Sdl","Sdl","npr","Sd","Sdl"),
    stage2 = c("Sd", "Sd", "Sd", "Sd", "Sdl", "rep", "rep"),
    stage1 = c("Sd", "rep", "Sd", "rep", "Sd", "mat", "mat"),
    eststage3 = c(NA, NA, NA, NA, "npr", NA, NA),
    eststage2 = c(NA, NA, NA, NA, "Sdl", NA, NA),
    eststage1 = c(NA, NA, NA, NA, "Sdl", NA, NA),
    givenrate = c(0.345, 0.345, 0.054, 0.054, NA, NA, NA),
    multiplier = c(NA, NA, NA, NA, NA, 0.345, 0.054),
    type = c(1, 1, 1, 1, 1, 3, 3), type_t12 = c(1, 2, 1, 2, 1, 1, 1),
    stageframe = lathframeipm, historical = TRUE)
```

Finally we can produce our historical IPMs. Because they will be huge, we will create them in sparse format (standard matrix format will lead to an object around 7 gigabytes in size).

```
lathmat3_importipm <- flefko3(stageframe = lathframeipm, modelsuite = lath3_vrm,
    supplement = lathsupp3, reduce = FALSE, sparse_output = TRUE)
summary(lathmat3_importipm)
>
> This historical lefkoMat object contains 3 matrices.
>
> Each matrix is square with }10609\mathrm{ rows and columns, and a total of 112550881 elements.
> A total of 2684746 survival transitions were estimated, with 894915.333 per matrix.
> A total of }60600\mathrm{ fecundity transitions were estimated, with }20200\mathrm{ per matrix.
> This lefkoMat object covers 1 population, 1 patch, and 3 time steps.
>
> The dataset contains a total of 0 unique individuals and 0 unique transitions.
>
> Survival probability sum check (each matrix represented by column in order):
> [,1] [,2] [,3]
> Min. 0.000 0.000 0.000
> 1st Qu. 0.989 0.981 0.988
> Median 0.997 0.996 0.997
> Mean 0.955 0.945 0.954
> 3rd Qu. 0.999 0.998 0.999
> Max. 1.000 1.000 1.000
```

As before, we can use this IPM in any way that might use a normal fbMPM in lefko3. The general approach shown above can also be used to develop age-based (Leslie) or age-by-stage function-based MPMs.

### 13.3 Running projections from imported IPMs and fbMPMs

The vrm_input approach can also be used to run custom function-based projections.

```
lath_proj <- f_projection3(format = 3, stageframe = lathframeipm,
    supplement = lathsupp2, modelsuite = lath_vrm, nreps = 10, times = 10,
    stochastic = TRUE)
> Warning: Option patch not set, so will set to first patch/population.
summary(lath_proj)
>
> The input lefkoProj object covers 1 population-patches.
> It includes }10\mathrm{ projected steps per replicate and }10\mathrm{ replicates.
> The number of replicates with population size above the threshold size of 1 is as in
> the following matrix, with pop-patches given by column and milepost times given by row:
> $milepost_sums
            11
> 1 10
> 3 10
> 6 10
> 8 10
> 11 10
```

```
>
> $extinction_times
> [1] NA
```

Let's take a look at what our projection replicates look like.

```
plot(lath_proj, bty = "n")
```



Figure 13.1: Replicated projection resulting from imported Lathyrus IPM

All other kinds of projections - ordered, cyclical, and stochastic - are also possible, as are density dependent projections in all cases. We encourage users to experiment with these methods.

### 13.4 Points to remember

1. Users may import IPMs and fbMPMs into lefko3 if they are presented as kernels of linear models, with function vrm_import ().
2. Imported IPMs and fbMPMs may also be loaded into projections, including function-based projections that create new matrices at each time step with function f_projection3().

## Chapter 14

## Further Issues III: Editing Matrices in MPMs

You'll soon be hearing the chime Close to midnight If I could turn back the time I'd make it alright

- Ghost, Dance Macabre (2018)

In the last two chapters, we saw how to import MPMs and discretized IPMs as lefkoMat objects. Let's now explore how we can edit these structures. Editing functions included the ability to add and delete matrices, to subset MPMs into smaller MPMs, and to edit matrices within MPMs.

Let's start by importing a lefkoMat object to work with. We will use the anthyllis MPM that is already packaged within lefko3 for now, although later on we will use other MPMs to illustrate other points.

```
data(anthyllis)
summary(anthyllis)
>
> This ahistorical lefkoMat object contains 27 matrices.
>
> Each matrix is square with 4 rows and columns, and a total of 16 elements.
> A total of }167\mathrm{ survival transitions were estimated, with 6.185 per matrix.
> A total of 48 fecundity transitions were estimated, with 1.778 per matrix.
> This lefkoMat object covers 1 population, 9 patches, and 3 time steps.
>
> This lefkoMat object appears to have been imported. Number of unique individuals and transitions no
>
> Survival probability sum check (each matrix represented by column in order):
> [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11]
>Min. 0.0667 0.500 0.0625 0.0769 0.0909 0.000 0.0000 0.333 0.171 0.000 0.0000
> 1st Qu. 0.0810 0.575 0.1708 0.1192 0.1334 0.075 0.0375 0.405 0.268 0.170 0.0000
> Median 0.1198 0.657 0.2106 0.3077 0.2276 0.100 0.1706 0.453 0.350 0.239 0.0000
> Mean 0.1599 0.637 0.2209 0.4231 0.2303 0.175 0.1895 0.469 0.320 0.203 0.0556
> 3rd Qu. 0.1987 0.720 0.2607 0.6116 0.3245 0.200 0.3225 0.517 0.402 0.271 0.0556
> Max. 0.3333 0.736 0.4000 1.0000 0.3750 0.500 0.4167 0.636 0.409 0.333 0.2222
> [,12] [,13] [,14] [,15] [,16] [,17] [,18] [,19] [,20] [,21] [,22]
>Min. 0.000 0.0000 0.000 0.0370 0.333 0.000 0.238 0.00 0.000 0.000 0.000
> 1st Qu. 0.375 0.0000 0.193 0.0408 0.394 0.381 0.310 0.00 0.500 0.000 0.125
> Median 0.500 0.0163 0.313 0.0657 0.564 0.518 0.410 0.00 0.833 0.250 0.271
```

```
> Mean 0.409 0.0510 0.281 0.0705 0.565 0.420}00.388 0.25 0.667 0.304 0.260
> 3rd Qu. 0.534 0.0673 0.401 0.0954 0.736 0.557 0.488 0.25 1.000 0.554 0.406
>Max. 0.636 0.1714 0.500 0.1136 0.800 0.645 0.493 1.00 1.000 0.714 0.500
> [,23] [,24] [,25] [,26] [,27]
> Min. 0.000 0.000 0.0000 0.000 0.0000
> 1st Qu. 0.500 0.250 0.0000 0.375 0.0833
> Median 0.762 0.333 0.0833 0.583 0.2556
> Mean 0.631 0.323 0.0833 0.542 0.3153
> 3rd Qu. 0.893 0.406 0.1667 0.750 0.4875
> Max. 1.000 0.625 0.1667 1.000 0.7500
```

Quickly looking over the structure, we see that we have an object that is composed of 9 populations (organized as patches in the lefkoMat object), with four years of data yielding three time steps for each population. The lefkoMat object holds an imported ahistorical MPM. Let's move on to some editing.

### 14.1 Subsetting a lefkoMat object

Situations may arise in which a user will wish to create a new lefkoMat object that is itself a subset of another. Such situations can arise, for example, when a lefkoMat object covers several patches or populations and the user wishes to create a single object to hold a single patch or population. In these situations, we can use the subset_1M() function.

Let's see an example of this function at work. Here, we will create a subset of the anthyllis object focused only on patch S.

```
anth_S <- subset_1M(anthyllis, patch = "S")
summary(anth_S)
>
> This ahistorical lefkoMat object contains 3 matrices.
>
> Each matrix is square with 4 rows and columns, and a total of 16 elements.
> A total of 10 survival transitions were estimated, with 3.333 per matrix.
> A total of 4 fecundity transitions were estimated, with 1.333 per matrix.
> This lefkoMat object covers 1 population, 1 patch, and 3 time steps.
>
> This lefkoMat object appears to have been imported. Number of unique individuals and transitions nc
>
> Survival probability sum check (each matrix represented by column in order):
> [,1] [,2] [,3]
> Min. 0.0000 0.000 0.0000
> 1st Qu. 0.0000 0.375 0.0833
> Median 0.0833 0.583 0.2556
> Mean 0.0833 0.542 0.3153
> 3rd Qu. 0.1667 0.750 0.4875
> Max. 0.1667 1.000 0.7500
```

And so we see here that we have created a much smaller MPM that includes only one of the original populations. The result includes three matrices only. We can see this within the labels object.

```
anth_S$labels
> pop patch year2
```

| $>1$ | 1 | $S$ | 2003 |  |
| :--- | :--- | :--- | :--- | :--- |
| $>2$ | 1 | $S$ | 2004 |  |
| $>$ | 3 | 1 | $S$ | 2005 |

Now let's try another subset, but this time focused on matrices in which the year in time $t$ is either 2004 or 2005.

```
anth_20042005 <- subset_lM(anthyllis, year = c(2004, 2005))
summary(anth_20042005)
>
> This ahistorical lefkoMat object contains 18 matrices.
>
> Each matrix is square with 4 rows and columns, and a total of 16 elements.
> A total of 117 survival transitions were estimated, with 6.5 per matrix.
> A total of 30 fecundity transitions were estimated, with 1.667 per matrix.
> This lefkoMat object covers 1 population, 9 patches, and 2 time steps.
>
> This lefkoMat object appears to have been imported. Number of unique individuals and transitions no
>
> Survival probability sum check (each matrix represented by column in order):
> [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11]
>Min. 0.500 0.0625 0.0909 0.000 0.333 0.171 0.0000 0.000 0.000 0.0370 0.000
> 1st Qu. 0.575 0.1708 0.1334 0.075 0.405 0.268 0.0000 0.375 0.193 0.0408 0.381
> Median 0.657 0.2106 0.2276 0.100 0.453 0.350 0.0000 0.500 0.313 0.0657 0.518
> Mean 0.637 0.2209 0.2303 0.175 0.469 0.320 0.0556 0.409 0.281 0.0705 0.420
> 3rd Qu. 0.720 0.2607 0.3245 0.200 0.517 0.402 0.0556 0.534 0.401 0.0954 0.557
> Max. 0.736 0.4000 0.3750 0.500 0.636 0.409 0.2222 0.636 0.500 0.1136 0.645
> [,12] [,13] [,14] [,15] [,16] [,17] [,18]
> Min. 0.238 0.000 0.000 0.000 0.000 0.000 0.0000
> 1st Qu. 0.310 0.500 0.000 0.500 0.250 0.375 0.0833
> Median 0.410 0.833 0.250 0.762 0.333 0.583 0.2556
>Mean 0.388 0.667 0.304 0.631 0.323 0.542 0.3153
> 3rd Qu. 0.488 1.000 0.554 0.893 0.406 0.750 0.4875
> Max. 0.493 1.000 0.714 1.000 0.625 1.000 0.7500
```

Now we have a subset that includes 18 matrices, since nine of the original 27 matrices were for year 2003 and so are not included in the new object.

We can make even more complicated subsets by using the matrix number from the A list, or the row number from the labels data frame, as below.

```
anth_complex <- subset_lM(anthyllis, mat_num = c(1, 5, 20, 21))
summary(anth_complex)
>
> This ahistorical lefkoMat object contains 4 matrices.
>
> Each matrix is square with 4 rows and columns, and a total of 16 elements.
> A total of 24 survival transitions were estimated, with 6 per matrix.
> A total of 6 fecundity transitions were estimated, with 1.5 per matrix.
> This lefkoMat object covers 1 population, 3 patches, and 3 time steps.
>
> This lefkoMat object appears to have been imported. Number of unique individuals and transitions no
>
```

```
> Survival probability sum check (each matrix represented by column in order):
> [,1] [,2] [,3] [,4]
> Min. 0.0667 0.0909 0.000 0.000
> 1st Qu. 0.0810 0.1334 0.500 0.000
> Median 0.1198 0.2276 0.833 0.250
> Mean 0.1599 0.2303 0.667 0.304
> 3rd Qu. 0.1987 0.3245 1.000 0.554
> Max. 0.3333 0.3750 1.000 0.714
```

Here we made a new lefkoMat object composed of four matrices, and chose those directly with the mat_num option. Let's see the metadata associated with these matrices.

```
anth_complex$labels
> pop patch year2
> 1 1 C 2003
>2 1 E 2004
> 3 1 Q 2004
>4 1 Q 2005
```


### 14.2 Adding and deleting matrices

All lefkoMat objects may be edited. One primary way in which we might wish to edit these objects is through the addition or deletion of matrices. Let's take a look at deleting matrices first.

The delete_lM() function is the primary means of removing matrices from a lefkoMat object. It takes the same inputs as subset_1M() and uses them in similar ways, but to remove matrices rather than subset them. It removes the matrix and edits the associated labels and quality control objects to reflect the change. Let's try one such situation, in which we create a new lefkoMat object by creating a copy of anthyllis in which we have removed patch S .

```
anth_noS <- delete_lM(anthyllis, patch = "S")
summary(anth_noS)
>
> This ahistorical lefkoMat object contains 24 matrices.
>
> Each matrix is square with 4 rows and columns, and a total of 16 elements.
> A total of }157\mathrm{ survival transitions were estimated, with 6.542 per matrix.
> A total of 44 fecundity transitions were estimated, with 1.833 per matrix.
> This lefkoMat object covers 1 population, 8 patches, and 3 time steps.
>
> This lefkoMat object appears to have been imported. Number of unique individuals and transitions nc
>
> Survival probability sum check (each matrix represented by column in order):
\(>\quad[, 1][, 2] \quad[, 3] \quad[, 4] \quad[, 5][, 6] \quad[, 7] \quad[, 8] \quad[, 9][, 10] \quad[, 11]\)
> Min. 0.0667 0.500 0.0625 0.0769 0.0909 0.000 0.0000 0.333 0.171 0.000 0.0000
> 1st Qu. 0.0810 0.575 0.1708 0.1192 0.1334 0.075 0.0375 0.405 0.268 0.170 0.0000
> Median 0.1198 0.657 0.2106 0.3077 0.2276 0.100}00.1706 0.453 0.350 0.239 0.0000
> Mean 0.1599 0.637 0.2209 0.4231 0.2303 0.175 0.1895 0.469 0.320 0.203 0.0556
> 3rd Qu. 0.1987 0.720 0.2607 0.6116 0.3245 0.200 0.3225 0.517 0.402 0.271 0.0556
> Max. 0.3333 0.736 0.4000 1.0000 0.3750 0.500 0.4167 0.636 0.409 0.333 0.2222
> [,12] [,13] [,14] [,15] [,16] [,17] [,18] [,19] [,20] [,21] [,22]
>Min. 0.000 0.0000 0.000 0.0370 0.333 0.000 0.238 0.00 0.000 0.000 0.000
```

```
> 1st Qu. 0.375 0.0000 0.193 0.0408 0.394 0.381 0.310 0.00 0.500 0.000 0.125
> Median 0.500 0.0163 0.313 0.0657 0.564 0.518}00.410 0.00 0.833 0.250 0.271
> Mean 0.409 0.0510 0.281 0.0705 0.565 0.420}00.388 0.25 0.667 0.304 0.260
> 3rd Qu. 0.534 0.0673 0.401 0.0954 0.736 0.557 0.488 0.25 1.000 0.554 0.406
> Max. 0.636 0.1714 0.500 0.1136 0.800 0.645 0.493 1.00 1.000 0.714 0.500
> [,23] [,24]
> Min. 0.000 0.000
> 1st Qu. 0.500 0.250
> Median 0.762 0.333
> Mean 0.631 0.323
> 3rd Qu. 0.893 0.406
> Max. 1.000 0.625
```

We can see that we have a new object with three fewer matrices, and one fewer patch. Let's see the labels object to make sure.


The last patch is clearly patch $R$. So, patch $S$ does not exist in the new edited lefkoMat object.
We may also add matrices, using the add_lM() function. In this case, we usually need to add a little information to make sure that the labels object is updated properly. We provide the new matrices as single matrices or as a list of matrices, via the Amats option with a logical setting for UFdecomp, or via the Umats and Fmats option if we have survival-transition matrices separated from fecundity matrices. We provide the extra information identifying these matrices via the entrystage, pop, patch, and year options.

Let's see how this function works by adding the $S$ patches back to our new lefkoMat object.

```
# POPN S 2003-2004
XS3 <- matrix(c(0, 0, 2.1, 0.816666667,
    0.166666667, 0, 0, 0,
    0, 0, 0, 0,
    0, 0, 0, 0.166666667), 4, 4, byrow = TRUE)
# POPN S 2004-2005
XS4 <- matrix(c(0, 0, 0, 7,
    0.333333333, 0.5, 0, 0,
    0, 0, 0, 0,
    0.333333333, 0, 0, 1), 4, 4, byrow = TRUE)
# POPN S 2005-2006
XS5 <- matrix(c(0, 0, 0, 1.4,
    0, 0, 0, 0,
    0, 0, 0, 0.2,
    0.111111111, 0.75, 0, 0.2), 4, 4, byrow = TRUE)
mats_list_S <- list(XS3, XS4, XS5)
anth_noS_addedS <- add_1M(anth_noS, Amats = mats_list_S, UFdecomp = TRUE,
    patch = c("S", "S", "S"), year = c(2003, 2004, 2005))
summary(anth_noS_addedS)
>
> This ahistorical lefkoMat object contains 27 matrices.
>
> Each matrix is square with 4 rows and columns, and a total of 16 elements.
> A total of 167 survival transitions were estimated, with 6.185 per matrix.
> A total of 48 fecundity transitions were estimated, with 1.778 per matrix.
> This lefkoMat object covers 1 population, }9\mathrm{ patches, and 3 time steps.
>
> This lefkoMat object appears to have been imported. Number of unique individuals and transitions nc
> Survival probability sum check (each matrix represented by column in order):
> [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11]
>Min. 0.0667 0.500 0.0625 0.0769 0.0909 0.000 0.0000 0.333 0.171 0.000 0.0000
> 1st Qu. 0.0810 0.575 0.1708 0.1192 0.1334 0.075 0.0375 0.405 0.268 0.170 0.0000
> Median 0.1198 0.657 0.2106 0.3077 0.2276 0.100 0.1706 0.453 0.350 0.239 0.0000
> Mean 0.1599 0.637 0.2209 0.4231 0.2303 0.175 0.1895 0.469 0.320 0.203 0.0556
> 3rd Qu. 0.1987 0.720 0.2607 0.6116 0.3245 0.200 0.3225 0.517 0.402 0.271 0.0556
> Max. 0.3333 0.736 0.4000 1.0000 0.3750 0.500 0.4167 0.636 0.409 0.333 0.2222
> [,12] [,13] [,14] [,15] [,16] [,17] [,18] [,19] [,20] [,21] [,22]
> Min. 0.000 0.0000 0.000 0.0370 0.333 0.000 0.238 0.00 0.000 0.000 0.000
> 1st Qu. 0.375 0.0000 0.193 0.0408 0.394 0.381 0.310 0.00 0.500 0.000 0.125
> Median 0.500 0.0163 0.313 0.0657 0.564 0.518 0.410 0.00 0.833 0.250 0.271
> Mean 0.409 0.0510 0.281 0.0705 0.565 0.420 0.388}00.25 0.667 0.304 0.260
> 3rd Qu. 0.534 0.0673 0.401 0.0954 0.736 0.557 0.488 0.25 1.000 0.554 0.406
>Max. 0.636 0.1714 0.500 0.1136 0.800 0.645 0.493 1.00 1.000 0.714 0.500
> [,23] [,24] [,25] [,26] [,27]
> Min. 0.000 0.000 0.0000 0.000 0.0000
> 1st Qu. 0.500 0.250 0.0000 0.375 0.0833
```

```
> Median 0.762 0.333 0.0833 0.583 0.2556
> Mean 0.631 0.323 0.0833 0.542 0.3153
> 3rd Qu. 0.893 0.406 0.1667 0.750 0.4875
> Max. 1.000 0.625 0.1667 1.000 0.7500
```

Users can compare this summary to that of anthyllis to see that they are the same.
In the options above, we set UFdecomp = TRUE. This is an important setting, because it allows R to use the stageframe in the existing lefkoMat object to infer which elements correspond to fecundity transitions. The function create_1M() also has this setting, but the default for that function is TRUE. In add_1M(), the default is set to FALSE, meaning that fecundity terms would not be set to the F matrices instead of the $U$ matrices without the user explicitly setting it to TRUE. Conversely, if matrix elements holding fecundity rates also mix those rates with survival probabilities, then the user may wish to set this option to FALSE across the board and edit the $U$ and $F$ matrices by hand to set the matrices properly.

### 14.3 Editing elements within matrices

Users may occasionally need to edit matrices within lefkoMat objects. If the user merely wishes to change a single element in a single matrix and knows the element's position, then it is easy to manipulate that element manually. However, the issue becomes more complicated if the matrix is particularly large, or if the elements that need adjusting occur in several matrices. To deal with these issues, lefko3 contains the function edit_lM(), which can change elements associated with specific transitions or groups of transitions, and in not just a single matrix but even in groups of matrices.

Function edit_lM() takes inputs similar to function supplemental(), with the addition of specific input options for population (pop), patch (patch), and time (year2). It uses stage names in the associated stageframe for the input lefkoMat object, and can utilize the same stage name shorthand that supplemental() uses.

As an example, let's say that we wish to create a lefkoMat object based on anthyllis, but in which the transition from seedling to small flowering adult is equal to 0.10 in population S . Let's take a look at the stageframe to see the location of the element in question.

```
anthyllis$ahstages[,c(1:3)]
> stage_id stage_id.1 stage
> 1 1 1 Sdl
>2 2 2 Veg
> 3 3 SmFlo
>4 4 4 LFlo
```

The survival transition element in question would be in column 1 and row 3 . Let's now investigate the value of this transition in the current MPM. Instead of targeting just the element, let's see all of the survival transition matrices (we would not likely do this if the matrices were large).

```
anthyllis$U[which(anthyllis$labels$patch == "S")]
> [[1]]
> [,1] [,2] [,3] [,4]
> [1,] 0.0000000 0 0 0.0000000
> [2,] 0.1666667 0 0 0.0000000
> [3,] 0.0000000 0 0 0.0000000
> [4,] 0.0000000 0 0 0.1666667
>
> [[2]]
```

```
> [,1] [,2] [,3] [,4]
> [1,] 0.0000000 0.0 0 0
> [2,] 0.3333333 0.5 0
> [3,] 0.0000000 0.0 0}
> [4,] 0.3333333
>
> [[3]]
> [,1] [,2] [,3] [,4]
> [1,] 0.0000000 0.00 0 0.0
> [2,] 0.0000000 0.00 0 0.0
> [3,] 0.0000000 0.00 0 0.2
> [4,] 0.11111111}0.75000.
```

We see that in all cases, this transition equals 0 . This may be due to a biologically impossible transition, or to no individuals taking that particular transition within a small dataset.

Let's create our new lefkoMat object, using anthyllis as a template and the edit_1M() function to hold the editing instructions. In addition to changing the survival transitions noted above, let's also double fecundity across the board. Note the similarity to how such information would be structured in function supplemental(), even to the point of using the "rep" shorthand to code for reproductive stages.

```
anth_editexample <- edit_lM(anthyllis, patch = "S",
    stage3 = c("SmFlo", "Sdl"), stage2 = c("Sdl", "rep"), givenrate = c(0.10, NA),
    multiplier = c(1.0, 2.0), type =c(1, 2))
```

Now let's compare the survival transitions in the old vs. new MPMs.

```
anthyllis$U[which(anthyllis$labels$patch == "S")]
> [[1]]
> [,1] [,2] [,3] [,4]
> [1,] 0.0000000 0 0 0.0000000
> [2,] 0.1666667 0 0 0.0000000
> [3,] 0.0000000 0 0 0.0000000
> [4,] 0.0000000 0 0 0.1666667
>
> [[2]]
> [,1] [,2] [,3] [,4]
> [1,] 0.0000000 0.0 0}
> [2,] 0.3333333 0.5 0}
> [3,] 0.0000000 0.0 0}
> [4,] 0.3333333 0.0 0
>
> [[3]]
> [,1] [,2] [,3] [,4]
> [1,] 0.0000000 0.00 0 0.0
> [2,] 0.0000000 0.00 0}00.
> [3,] 0.0000000 0.00 0}00.
> [4,] 0.11111111 0.75 0}00.
anth_editexample$U[which(anth_editexample$labels$patch == "S")]
> [[1]]
> [,1] [,2] [,3] [,4]
> [1,] 0.0000000 0 0 0.0000000
```

```
> [2,] 0.1666667 0 0 0.0000000
> [3,] 0.1000000 0 0 0.0000000
> [4,] 0.0000000 0 0 0.1666667
>
> [[2]]
> [,1] [,2] [,3] [,4]
> [1,] 0.0000000 0.0 0 0
> [2,] 0.3333333 0.5 0 0
> [3,] 0.1000000 0.0 0 0
> [4,] 0.3333333 0.0 0 1
>
> [[3]]
> [,1] [,2] [,3] [,4]
> [1,] 0.0000000 0.00 0 0.0
> [2,] 0.0000000 0.00 0 0.0
> [3,] 0.1000000 0.00 0}00.
> [4,] 0.1111111 0.75 0}00.
```

The survival transition from seedling to small flowering adult has been altered. Now let's see the fecundity matrices in the old vs. new MPMs.

```
anthyllis$F[which(anthyllis$labels$patch == "S")]
> [[1]]
> [,1] [,2] [,3] [,4]
> [1,] 0 0 2.1 0.8166667
> [2,] 0
> [3,] 0
> [4,] 0 0 0.0 0.0000000
>
> [[2]]
> [,1] [,2] [,3] [,4]
> [1,] 0 0 0 % 7
> [2,] 0
> [3,] 0
> [4,] 0 0 0 0
>
> [[3]]
> [,1] [,2] [,3] [,4]
> [1,] 0 0 0 1.4
> [2,] 0 0 0 0.0
> [3,] 0}0000
> [4,] 0 0 0 0.0
anth_editexample$F[which(anth_editexample$labels$patch == "S")]
> [[1]]
> [,1] [,2] [,3] [,4]
> [1,] 0 0 4.2 1.633333
> [2,] 0 0 0.0 0.000000
> [3,] 0 0}00.0 0.00000
> [4,] 0}0
>
> [[2]]
> [,1] [,2] [,3] [,4]
```

|  | $>[1]$, | 0 | 0 | 0 | 14 |
| ---: | :--- | ---: | ---: | ---: | ---: |
|  | $>[2]$, | 0 | 0 | 0 | 0 |
|  | $>[3]$, | 0 | 0 | 0 | 0 |
|  | $>[4]$, | 0 | 0 | 0 | 0 |
|  | $>$ |  |  |  |  |
|  | $>[[3]]$ |  |  |  |  |
|  | $>$ | $[, 1]$ | $[, 2]$ | $[, 3]$ | $[, 4]$ |
|  | $>[1]$, | 0 | 0 | 0 | 2.8 |
|  | $>[2]$, | 0 | 0 | 0 | 0.0 |
|  | $>[3]$, | 0 | 0 | 0 | 0.0 |
|  | $>[4]$, | 0 | 0 | 0 | 0.0 |

Clearly the fecundity has been doubled.

### 14.4 Dealing with age-specific issues

The function edit_1M() can be used to edit Leslie and age-by-stage MPMs, as well as stage-based ahistorical and historical MPMs. Let's load an age-by-stage MPM, this time focused on the lathyrus dastaset, and look at a summary.

```
data(lathyrus)
lathyrus$indiv_id <- paste(lathyrus$SUBPLOT, lathyrus$GENET)
sizevector <- c(0, 100, 13, 127, 3730, 3800, 0)
stagevector <- c("Sd", "Sdl", "VSm", "Sm", "VLa", "Flo", "Dorm")
repvector <- c(0, 0, 0, 0, 0, 1, 0)
obsvector <- c(0, 1, 1, 1, 1, 1, 0)
matvector <- c(0, 0, 1, 1, 1, 1, 1)
immvector <- c(1, 1, 0, 0, 0, 0, 0)
propvector <- c(1, 0, 0, 0, 0, 0, 0)
indataset <- c(0, 1, 1, 1, 1, 1, 1)
binvec <- c(0, 100, 11, 103, 3500, 3800, 0.5)
lathframe_raw <- sf_create(sizes = sizevector, stagenames = stagevector,
    repstatus = repvector, obsstatus = obsvector, matstatus = matvector,
    immstatus = immvector, indataset = indataset, binhalfwidth = binvec,
    propstatus = propvector)
lathvert_raw <- verticalize3(lathyrus, noyears = 4, firstyear = 1988,
    patchidcol = "SUBPLOT", individcol = "indiv_id", blocksize = 9,
    juvcol = "Seedling1988", sizeacol = "Volume88", repstracol = "FCODE88",
    fecacol = "Intactseed88", deadacol = "Dead1988", nonobsacol = "Dormant1988",
    stageassign = lathframe_raw, stagesize = "sizea", censorcol = "Missing1988",
    censorkeep = NA, censorRepeat = TRUE, censor = TRUE)
lathvert_raw_small <- subset(lathvert_raw, firstseen > 1988)
lathsupp2 <- supplemental(stage3 = c("Sd", "Sdl", "Sd", "Sdl"),
    stage2 = c("Sd", "Sd", "rep", "rep"), givenrate = c(0.345, 0.054, NA, NA),
    multiplier = c(NA, NA, 0.345, 0.054), type = c(1, 1, 3, 3),
```

```
    stageframe = lathframe_raw, historical = FALSE, agebased = TRUE)
lathmat2p_raw <- arlefko2(data = lathvert_raw_small, stageframe = lathframe_raw,
    supplement = lathsupp2, stages = c("stage3", "stage2", "stage1"),
    patch = "all", patchcol = "patchid", yearcol = "year2", agecol = "obsage",
    indivcol = "individ")
summary(lathmat2p_raw)
>
> This ahistorical lefkoMat object contains 12 matrices.
>
> Each matrix is square with 21 rows and columns, and a total of 441 elements.
> A total of 209 survival transitions were estimated, with 17.417 per matrix.
> A total of 12 fecundity transitions were estimated, with 1 per matrix.
> This lefkoMat object covers 1 population, 6 patches, and 2 time steps.
>
> The dataset contains a total of 345 unique individuals and 531 unique transitions.
>
> Survival probability sum check (each matrix represented by column in order):
> [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11] [,12]
> Min. 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000
> 1st Qu. 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000
> Median 0.000 0.000 0.000 0.399 0.000 0.399 0.000 0.399 0.000 0.399 0.000 0.000
> Mean 0.281 0.349 0.261 0.385 0.274 0.463 0.268 0.423 0.223 0.388 0.234 0.279
> 3rd Qu. 0.399 0.778 0.399 0.750 0.399 1.000 0.399 0.778 0.399 0.818 0.399 0.399
>Max. 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000
```

These are not huge matrices. They contain seven stages, and $3+$ ages (meaning that we have three ages recognized, in which the final age incorporates ages greater than the final age). Further, we have already used the supplemental() function to make general edits to all matrices. Let's take a look at the first two A matrices, corresponding to patch 1.

| lathmat2p_raw\$A[c(1:2)] |  |  |  |  |  |  |  |  |  |  |  |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| > [[1]] |  |  |  |  |  |  |  |  |  |  |  |
| > | [,1] | [,2] | [,3] | [,4] | [,5] | [,6] | [,7] | [,8] | [,9] | [,10] | [,11] |
| > [1,] | 0.000 | 0.0000000 | 0.0000000 | 0.0000 | 0.000 | 0 | 1.035 | 0.000 | 0 | 0 | 0 |
| $>$ [2,] | 0.000 | 0.0000000 | 0.0000000 | 0.0000 | 0.000 | 0 | 0.162 | 0.000 | 0 | 0 | 0 |
| $>$ [3,] | 0.000 | 0.0000000 | 0.0000000 | 0.0000 | 0.000 | 0 | 0.000 | 0.000 | 0 | 0 | 0 |
| $>$ [4,] | 0.000 | 0.0000000 | 0.0000000 | 0.0000 | 0.000 | 0 | 0.000 | 0.000 | 0 | 0 | 0 |
| $>$ [5,] | 0.000 | 0.0000000 | 0.0000000 | 0.0000 | 0.000 | 0 | 0.000 | 0.000 | 0 | 0 | 0 |
| $>$ [6,] | 0.000 | 0.0000000 | 0.0000000 | 0.0000 | 0.000 | 0 | 0.000 | 0.000 | 0 | 0 | 0 |
| $>$ [7,] | 0.000 | 0.0000000 | 0.0000000 | 0.0000 | 0.000 | 0 | 0.000 | 0.000 | 0 | 0 | 0 |
| > [8,] | 0.345 | 0.0000000 | 0.0000000 | 0.0000 | 0.000 | 0 | 0.000 | 0.000 | 0 | 0 | 0 |
| $>$ [9,] | 0.054 | 0.0000000 | 0.0000000 | 0.0000 | 0.000 | 0 | 0.000 | 0.000 | 0 | 0 | 0 |
| > [10,] | 0.000 | 0.9047619 | 0.7142857 | 0.1875 | 0.250 | 0 | 0.000 | 0.000 | 0 | 0 | 0 |
| > [11,] | 0.000 | 0.0000000 | 0.1428571 | 0.6875 | 0.500 | 0 | 1.000 | 0.000 | 0 | 0 | 0 |
| > [12,] | 0.000 | 0.0000000 | 0.0000000 | 0.0000 | 0.000 | 0 | 0.000 | 0.000 | 0 | 0 | 0 |
| $>$ [13,] | 0.000 | 0.0000000 | 0.0000000 | 0.0625 | 0.125 | 0 | 0.000 | 0.000 | 0 | 0 | 0 |
| > [14,] | 0.000 | 0.0000000 | 0.0000000 | 0.0000 | 0.125 | 0 | 0.000 | 0.000 | 0 | 0 | 0 |
| > [15,] | 0.000 | 0.0000000 | 0.0000000 | 0.0000 | 0.000 | 0 | 0.000 | 0.345 | 0 | 0 | 0 |
| > [16,] | 0.000 | 0.0000000 | 0.0000000 | 0.0000 | 0.000 | 0 | 0.000 | 0.054 | 0 | 0 | 0 |
| > [17, ] | 0.000 | 0.0000000 | 0.0000000 | 0.0000 | 0.000 | 0 | 0.000 | 0.000 | 0 | 0 | 0 |
| > [18, ] | 0.000 | 0.0000000 | 0.0000000 | 0.0000 | 0.000 | 0 | 0.000 | 0.000 | 0 | 0 | 0 |



| $>$ | [2,] | 0.00000000 | 0 | 0 | 0 | 0.000 | 0 | 0 | 0 | 0 | 0 | 0 |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| > | [3,] | 0.00000000 | 0 | 0 | 0 | 0.000 | 0 | 0 | 0 | 0 | 0 | 0 |
| $>$ | [4, ] | 0.00000000 | 0 | 0 | 0 | 0.000 | 0 | 0 | 0 | 0 | 0 | 0 |
| > | [5,] | 0.00000000 | 0 | 0 | 0 | 0.000 | 0 | 0 | 0 | 0 | 0 | 0 |
| > | [6,] | 0.00000000 | 0 | 0 | 0 | 0.000 | 0 | 0 | 0 | 0 | 0 | 0 |
| > | [7,] | 0.00000000 | 0 | 0 | 0 | 0.000 | 0 | 0 | 0 | 0 | 0 | 0 |
| > | [8,] | 0.00000000 | 0 | 0 | 0 | 0.000 | 0 | 0 | 0 | 0 | 0 | 0 |
| > | [9,] | 0.00000000 | 0 | 0 | 0 | 0.000 | 0 | 0 | 0 | 0 | 0 | 0 |
| > | [10,] | 0.00000000 | 0 | 0 | 0 | 0.000 | 0 | 0 | 0 | 0 | 0 | 0 |
| > | [11,] | 0.00000000 | 0 | 0 | 0 | 0.000 | 0 | 0 | 0 | 0 | 0 | 0 |
| > | [12,] | 0.00000000 | 0 | 0 | 0 | 0.000 | 0 | 0 | 0 | 0 | 0 | 0 |
| > | [13,] | 0.00000000 | 0 | 0 | 0 | 0.000 | 0 | 0 | 0 | 0 | 0 | 0 |
| > | [14,] | 0.00000000 | 0 | 0 | 0 | 0.000 | 0 | 0 | 0 | 0 | 0 | 0 |
| > | [15,] | 0.00000000 | 0 | 0 | 0 | 0.345 | 0 | 0 | 0 | 0 | 0 | 0 |
| > | [16,] | 0.00000000 | 0 | 0 | 0 | 0.054 | 0 | 0 | 0 | 0 | 0 | 0 |
| > | [17,] | 0.05555556 | 0 | 0 | 0 | 0.000 | 0 | 0 | 0 | 0 | 0 | 0 |
| > | [18,] | 0.55555556 | 0 | 1 | 0 | 0.000 | 0 | 0 | 0 | 0 | 0 | 0 |
| > | [19,] | 0.11111111 | 0 | 0 | 0 | 0.000 | 0 | 0 | 0 | 0 | 0 | 0 |
| > | [20,] | 0.05555556 | 0 | 0 | 1 | 0.000 | 0 | 0 | 0 | 0 | 0 | 0 |
| > | [21,] | 0.2222222 | 0 | 0 | 0 | 0.000 | 0 | 0 | 0 | 0 | 0 | 0 |

The matrix structure is as a supermatrix in which each age is composed of seven rows and seven columns, corresponding to the stages in the stageframe. One interesting observation about these matrices is that the third age is generally empty. This is of course due to the fact that the dataset itself only contains four years, and we subset this to reduce it by one year. So, one edit that we might wish to make is to copy the transitions from age 2 over to age 3 . Let's try to do that here.

```
lathmat2p_raw_ageedit <- edit_lM(lathmat2p_raw, stage3 = c("all", "all"),
    stage2 = c("all", "all"), eststage3 = c("all", "all"),
    eststage2 = c("all", "all"), age2 = c(3, 3), estage2 = c(2, 2),
    multiplier = c(1, 1), type = c(1, 2))
summary(lathmat2p_raw_ageedit)
>
> This ahistorical lefkoMat object contains 12 matrices.
>
> Each matrix is square with 21 rows and columns, and a total of 441 elements.
> A total of 209 survival transitions were estimated, with 17.417 per matrix.
> A total of 12 fecundity transitions were estimated, with 1 per matrix.
> This lefkoMat object covers 1 population, 6 patches, and 2 time steps.
>
> The dataset contains a total of 345 unique individuals and 531 unique transitions.
>
> Survival probability sum check (each matrix represented by column in order):
> [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11] [,12]
> Min. 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000
> 1st Qu. 0.000 0.000 0.000 0.143 0.000 0.399 0.000 0.399 0.000 0.399 0.000 0.000
> Median 0.000 0.500 0.000 0.667 0.000 0.846 0.000 0.700 0.000 0.625 0.000 0.399
> Mean 0.281 0.533 0.261 0.564 0.274 0.693 0.268 0.624 0.223 0.608 0.234 0.417
> 3rd Qu. 0.399 1.000 0.399 1.000 0.399 1.000 0.399 1.000 0.399 1.000 0.399 0.889
> Max. 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000
```

Perusing the summary output, we might spot a few differences in the survival probability sum check.

For example, the mean value given for matrix 2 is higher than previously. Let's now take a look at the first two matrices.

```
lathmat2p_raw_ageedit$A[c(1:2)]
> [[1]]
> [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11]
> [1,] 0.000 0.0000000 0.0000000 0.0000 0.000 0 1.035 0.000 0 0 0 0
> [2,] 0.000 0.0000000 0.0000000 0.0000 0.000 0 0.162 0.000 0 0 0 0
> [3,] 0.000 0.0000000 0.0000000 0.0000 0.000 0 0.000 0.000
> [4,] 0.000 0.0000000 0.0000000 0.0000 0.000
> [5,] 0.000 0.0000000 0.0000000 0.0000 0.000
> [6,] 0.000 0.0000000 0.0000000 0.0000 0.000
> [7,] 0.000 0.0000000 0.0000000 0.0000 0.000
> [8,] 0.345 0.0000000 0.0000000 0.0000 0.000
> [9,] 0.054 0.0000000 0.0000000 0.0000 0.000
> [10,] 0.000 0.9047619}00.7142857 0.1875 0.250
> [11,] 0.000 0.0000000 0.1428571 0.6875 0.500
> [12,] 0.000 0.0000000 0.0000000 0.0000 0.000
> >[13,] 0.000 0.0000000 0.0000000 0.0625 0.125
> [14,] 0.000 0.0000000 0.0000000 0.0000 0.125
> [15,] 0.000 0.0000000 0.0000000 0.0000 0.000
> [17,] 0.000 0.0000000 0.0000000 0.0000 0.000
> [18,] 0.000 0.0000000 0.0000000 0.0000 0.000
> [19,] 0.000 0.0000000 0.0000000 0.0000 0.000
> [20,] 0.000 0.0000000 0.0000000 0.0000 0.000 
> [21,] 0.000 0.0000000 0.0000000 0.0000 0.000 0 0.000 0.000 0 0 [, [,13] [,14] [,15] [,16] [,17] [,18] [,19] [,20] [,21]
\begin{tabular}{llllllllllll}
\(>\) & {\([1]\),} & 0 & 0 & 0 & 0.000 & 0 & 0 & 0 & 0 & 0 & 0 \\
\(>\) & {\([2]\),} & 0 & 0 & 0 & 0.000 & 0 & 0 & 0 & 0 & 0 & 0 \\
\(>\) & {\([3]\),} & 0 & 0 & 0 & 0.000 & 0 & 0 & 0 & 0 & 0 & 0 \\
\(>\) & {\([4]\),} & 0 & 0 & 0 & 0.000 & 0 & 0 & 0 & 0 & 0 & 0 \\
\(>\) & {\([5]\),} & 0 & 0 & 0 & 0.000 & 0 & 0 & 0 & 0 & 0 & 0 \\
\(>[6]\), & 0 & 0 & 0 & 0.000 & 0 & 0 & 0 & 0 & 0 & 0 \\
\(>[7]\), & 0 & 0 & 0 & 0.000 & 0 & 0 & 0 & 0 & 0 & 0 \\
\(>\) & {\([8]\),} & 0 & 0 & 0 & 0.000 & 0 & 0 & 0 & 0 & 0 & 0 \\
\(>[9]\), & 0 & 0 & 0 & 0.000 & 0 & 0 & 0 & 0 & 0 & 0 \\
\(>[10]\), & 0 & 0 & 0 & 0.000 & 0 & 0 & 0 & 0 & 0 & 0 \\
\(>[11]\), & 0 & 0 & 0 & 0.000 & 0 & 0 & 0 & 0 & 0 & 0 \\
\(>[12]\), & 0 & 0 & 0 & 0.000 & 0 & 0 & 0 & 0 & 0 & 0 \\
\(>[13]\), & 0 & 0 & 0 & 0.000 & 0 & 0 & 0 & 0 & 0 & 0 \\
\(>[14]\), & 0 & 0 & 0 & 0.000 & 0 & 0 & 0 & 0 & 0 & 0 \\
\(>[15]\), & 0 & 0 & 0 & 0.345 & 0 & 0 & 0 & 0 & 0 & 0 \\
\(>[16]\), & 0 & 0 & 0 & 0.054 & 0 & 0 & 0 & 0 & 0 & 0 \\
\(>[17]\), & 0 & 0 & 0 & 0.000 & 0 & 0 & 0 & 0 & 0 & 0 \\
\(>[18]\), & 0 & 0 & 0 & 0.000 & 0 & 0 & 0 & 0 & 0 & 0 \\
\(>[19]\), & 0 & 0 & 0 & 0.000 & 0 & 0 & 0 & 0 & 0 & 0 \\
\(>[20]\), & 0 & 0 & 0 & 0.000 & 0 & 0 & 0 & 0 & 0 & 0 \\
\(>[21]\), & 0 & 0 & 0 & 0.000 & 0 & 0 & 0 & 0 & 0 & 0
\end{tabular}
>
> [[2]]
> [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
```

| > | [1,] | 0.0000 .0000000 | 0.0000000 | 0.0 | 0 |  | 00.000 | 00.0000 | 00000 |  |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| > | [2,] | 0.0000 .0000000 | 0.0000000 | 0.0 | 0 |  | 00.000 | 00.000 | 00000 |  |
| > | [3,] | 0.0000 .0000000 | 0.0000000 | 0.0 | 0 |  | 00.000 | 00.000 | 00000 |  |
| > | [4, ] | 0.0000 .0000000 | 0.0000000 | 0.0 | 0 |  | 00.000 | 00.000 | 00000 |  |
| > | [5,] | 0.0000 .0000000 | 0.0000000 | 0.0 | 0 |  | 00.000 | 00.000 | 00000 |  |
| > | [6,] | 0.0000 .0000000 | 0.0000000 | 0.0 | 0 |  | 00.000 | 00.000 | 00000 |  |
| $>$ | [7, ] | 0.0000 .0000000 | 0.0000000 | 0.0 | 0 |  | 00.000 | 00.000 | 00000 |  |
| $>$ | [8,] | 0.3450 .0000000 | 0.0000000 | 0.0 | 0 |  | 00.000 | 00.000 | 00000 |  |
| > | [9,] | 0.0540 .0000000 | 0.0000000 | 0.0 | 0 |  | 00.000 | 00.000 | 00000 |  |
| > | [10,] | 0.0000 .6666667 | 0.5714286 | 0.0 | 0 |  | 00.000 | 00.000 | 00000 |  |
| > | [11,] | 0.0000 .0000000 | 0.4285714 | 0.5 | 0 |  | 00.000 | 00.000 | 00000 |  |
| > | [12,] | 0.0000 .0000000 | 0.0000000 | 0.0 | 0 |  | 00.000 | 00.0000 | 00000 |  |
| > | [13,] | 0.0000 .1111111 | 0.0000000 | 0.0 | 0 |  | 00.000 | 00.000 | 00000 |  |
| > | [14,] | 0.0000 .0000000 | 0.0000000 | 0.0 | 0 |  | 00.000 | 00.000 | 00000 |  |
| > | [15,] | 0.0000 .0000000 | 0.0000000 | 0.0 | 0 |  | 00.345 | 00.0000 | 00000 |  |
| > | [16,] | 0.0000 .0000000 | 0.0000000 | 0.0 | 0 |  | 00.054 | 00.000 | 00000 |  |
| > | [17,] | 0.0000 .0000000 | 0.0000000 | 0.0 | 0 |  | 00.000 | 00.72 | 41379 |  |
| > | [18,] | 0.0000 .0000000 | 0.0000000 | 0.0 | 0 |  | 00.000 | 00.137 | 79310 |  |
| > | [19,] | 0.0000 .0000000 | 0.0000000 | 0.0 | 0 |  | 00.000 | 00.000 | 00000 |  |
| > | [20,] | 0.0000 .0000000 | 0.0000000 | 0.0 | 0 |  | 00.000 | 00.000 | 00000 |  |
| > | [21,] | 0.0000 .0000000 | 0.0000000 | 0.0 | 0 |  | 00.000 | 00.000 | 0000 |  |
| > |  | [,11] [,12] | [,13] [,1 | 4] [,15] | [,16] |  | [,17] | [,18] | [,19] | [,20] |
| > | [1,] | 0.00000000 | 0 | 00.000 | 0 |  | 0.0000000 | 0.00000000 | 0 | 0 |
| $>$ | [2,] | 0.00000000 | 0 | 00.000 | 0 |  | 0.0000000 | 0.00000000 | 0 | 0 |
| > | [3,] | 0.00000000 | 0 | 00.000 | 0 |  | 0.0000000 | 0.00000000 | 0 | 0 |
| $>$ | [4, ] | 0.00000000 | 0 | 00.000 | 0 |  | 0.0000000 | 0.00000000 | 0 | 0 |
| $>$ | [5, ] | 0.00000000 | 0 | 00.000 | 0 |  | 0.0000000 | 0.00000000 | 0 | 0 |
| > | [6, ] | 0.00000000 | 0 | 00.000 | 0 |  | 0.0000000 | 0.00000000 | 0 | 0 |
| > | [7, ] | 0.00000000 | 0 | 00.000 | 0 |  | 0.0000000 | 0.00000000 | 0 | 0 |
| > | [8,] | 0.00000000 | 0 | 00.000 | 0 |  | 0.0000000 | 0.00000000 | 0 | 0 |
| > | [9,] | 0.00000000 | 0 | 00.000 |  |  | 0.0000000 | 0.00000000 | 0 | 0 |
| > | [10,] | 0.00000000 | 0 | 00.000 |  |  | 0.0000000 | 0.00000000 | 0 | 0 |
| > | [11,] | 0.00000000 | 0 | 00.000 |  |  | 0.0000000 | 0.00000000 | 0 | 0 |
| > | [12,] | 0.00000000 | 0 | 00.000 |  |  | 0.0000000 | 0.00000000 | 0 | 0 |
| > | [13,] | 0.00000000 | 0 | 00.000 |  |  | 0.0000000 | 0.00000000 | 0 | 0 |
| > | [14,] | 0.00000000 | 0 | 00.000 |  |  | 0.0000000 | 0.00000000 | 0 | 0 |
| > | [15,] | 0.00000000 | 0 | 00.345 |  |  | 0.0000000 | 0.00000000 | 0 | 0 |
| > | [16,] | 0.00000000 | 0 | 00.054 |  |  | 0.0000000 | 0.00000000 | 0 | 0 |
| > | [17,] | 0.05555556 | 0 | 00.000 |  | 0 | 0.7241379 | 0.05555556 | 0 | 0 |
| > | [18,] | 0.55555556 | - 1 | 00.000 |  |  | 0.1379310 | 0.55555556 | 0 | 1 |
| > | [19,] | 0.111111110 | 0 | 00.000 |  | 0 | 0.0000000 | 0.11111111 | 0 | 0 |
| > | [20,] | 0.05555556 | 0 | 10.000 |  | 0 | 0.0000000 | 0.05555556 | 0 | 0 |
| > | [21,] | 0.222222220 | 0 | 00.000 | - | 0 | 0.0000000 | 0.22222222 | 0 | 0 |
| > |  | [,21] |  |  |  |  |  |  |  |  |
| > | [1,] | 0 |  |  |  |  |  |  |  |  |
| > | [2,] | 0 |  |  |  |  |  |  |  |  |
| > | [3,] | 0 |  |  |  |  |  |  |  |  |
| > | [4, ] | 0 |  |  |  |  |  |  |  |  |
| > | [5,] | 0 |  |  |  |  |  |  |  |  |
| > | [6,] | 0 |  |  |  |  |  |  |  |  |
|  | [7, ] | 0 |  |  |  |  |  |  |  |  |


| $>$ | $[8]$, | 0 |
| ---: | ---: | ---: |
| $>$ | $[9]$, | 0 |
| $>$ | $[10]$, | 0 |
| $>[11]$, | 0 |  |
| $>[12]$, | 0 |  |
| $>[13]$, | 0 |  |
| $>[14]$, | 0 |  |
| $>[15]$, | 0 |  |
| $>[16]$, | 0 |  |
| $>[17]$, | 0 |  |
| $>[18]$, | 0 |  |
| $>[19]$, | 0 |  |
| $>[20]$, | 1 |  |
| $>[21]$, | 0 |  |

We see that the first matrix appears the same, but this is because there are essentially no mature transitions for age 2 individuals. The second matrix shows age 2 transitions have been copied over to age 3 .

Similar strategies can be used with Leslie MPMs, and also to change specific transitions in single matrices, or groups of matrices.

### 14.5 Adding stages to MPMs

Situations may arise in which users wish to alter an MPM by adding a stage, or perhaps even several stages. One such situation is in the creation of age-hybrid MPMs, in which a Leslie MPM is altered with the addition of a non-age-specific stage. We describe such a situation in section 6.6 of chapter 6 , but repeat the code here to show how this can be done.

```
lathvert_base <- verticalize3(lathyrus, noyears = 4, firstyear = 1988,
    patchidcol = "SUBPLOT", individcol = "indiv_id", blocksize = 9,
    sizeacol = "Volume88", repstracol = "FCODE88", fecacol = "Intactseed88",
    deadacol = "Dead1988", censorcol = "Missing1988", censorkeep = NA,
    censorRepeat= TRUE, censor = TRUE, NAas0 = TRUE, NRasRep = TRUE,
    NOasObs = TRUE)
lathvert_age <- subset(lathvert_base, firstseen > 1988)
lathmodels2_age <- modelsearch(lathvert_age, historical = FALSE,
    approach = "mixed", suite = "cons", bestfit = "AICc&k", age = "obsage",
    vitalrates = c("surv", "fec"), fecdist = "negbin", indiv = "individ",
    year = "year2", year.as.random = TRUE, patch.as.random = TRUE,
    show.model.tables = TRUE, fec.zero = TRUE, global.only = TRUE,
    test.age = TRUE, quiet = "partial")
>
> Developing global model of survival probability...
>
> Global model of survival probability developed. Proceeding with model dredge...
>
> Observation probability will be treated as constant.
>
> Primary size will be treated as constant.
>
```

```
> Reproduction probability will be treated as constant.
>
> Developing global model of fecundity...
>
> Global model of fecundity developed. Proceeding with model dredge...
lathsupp2_ageonly <- supplemental(age2 = c(2, 3), multiplier = c(0.345, 0.345),
    type = c(3, 3), historical = FALSE, stagebased = FALSE, agebased = TRUE)
lathmat2ageonly_func <- fleslie(year = "all", data = lathvert_age,
    supplement = lathsupp2_ageonly, modelsuite = lathmodels2_age)
lathmat2_agehybrid <- add_stage(lathmat2ageonly_func, add_before = 1,
    stage_name = "DormSeed")
lathmat2_agehybrid$ahstages$propstatus[1] <- 1 # Dormant seeds are propagules
lathmat2_agehybrid <- edit_lM(lathmat2_agehybrid,
    stage3 = c("DormSeed", "Age1", "Age1", "DormSeed"),
    stage2 = c("DormSeed", "DormSeed", "Age3", "Age3"),
    eststage3 = c(NA, NA, NA, "Age1"),
    eststage2 = c(NA, NA, NA, "Age3"),
    givenrate = c(0.5, 0.2, NA, NA),
    multiplier = c(NA, NA, 0.5, 1),
    type = c(1, 1, 3, 2))
summary(lathmat2_agehybrid)
>
> This ahistorical lefkoMat object contains 2 matrices.
>
> Each matrix is square with 4 rows and columns, and a total of 16 elements.
> A total of 10 survival transitions were estimated, with 5 per matrix.
> A total of 6 fecundity transitions were estimated, with 3 per matrix.
> This lefkoMat object covers 1 population, 1 patch, and 2 time steps.
> The dataset contains a total of 345 unique individuals and 531 unique transitions.
>
Vital rate modeling quality control:
>
Survival estimated with 345 individuals and 531 individual transitions.
Observation probability not estimated.
Primary size transition not estimated.
Secondary size transition not estimated.
Tertiary size transition not estimated.
Reproduction probability not estimated.
Fecundity estimated with 16 individuals and 19 individual transitions.
Juvenile survival not estimated.
Juvenile observation probability not estimated.
Juvenile primary size transition not estimated.
Juvenile secondary size transition not estimated.
Juvenile tertiary size transition not estimated.
```

```
> Juvenile reproduction probability not estimated.
> Juvenile maturity transition probability not estimated.
>
> Survival probability sum check (each matrix represented by column in order):
> [,1] [,2]
> Min. 0.700 0.700
> 1st Qu. 0.782 0.710
> Median 0.816 0.722
> Mean 0.792 0.723
> 3rd Qu. 0.826 0.736
> Max. 0.835 0.749
```

Basically, we begin by creating a purely age-based MPM, called lathmat2ageonly_func. We then create a new MPM using the add_stage () function, telling that function to copy the previous MPM but to include a new stage called DormSeed just before the first stage in the ahstages object of the Leslie MPM. The function creates this stage, adding it to the ahstages object (also the hstages or agestages objects, if appropriate) and to all of the matrices. These new rows and columns are zero filled with zeroes, and the default description of the stage in the ahstages object details the stage as immature and observable. So, we need to edit the propstatus term associated with it to reflect this stage's status as a propagule (a dormant seed), and then we still need to use the edit_1M() function to add in the appropriate new transition values in the new rows and columns of the $A, U$, and $F$ matrices. Here is the full MPM.

```
lathmat2_agehybrid
> $A
> $A[[1]]
> [,1] [,2] [,3] [,4]
> [1,] 0.5 0.0000000 0.0000000 0.1696108
> [2,] 0.2 0.2318637 0.2787757 0.1696108
> [3,] 0.0 0.8089559 0.0000000 0.0000000
> [4,] 0.0 0.0000000 0.8226009 0.8354695
>
> $A[[2]]
> [,1] [,2] [,3] [,4]
> [1,] 0.5 0.0000000 0.0000000 0.1695883
> [2,] 0.2 0.2304600 0.2783520}0.169588
> [3,] 0.0 0.7130813 0.0000000 0.0000000
> [4,] 0.0 0.0000000 0.7312994 0.7487683
>
>
> $U
> $U[[1]]
> [,1] [,2] [,3] [,4]
> [1,] 0.5 0.0000000 0.0000000 0.0000000
> [2,] 0.2 0.0000000 0.0000000 0.0000000
> [3,] 0.0 0.8089559 0.0000000 0.0000000
> [4,] 0.0 0.0000000 0.8226009 0.8354695
>
> $U[[2]]
> [,1] [,2] [,3] [,4]
> [1,] 0.5 0.0000000 0.0000000 0.0000000
> [2,] 0.2 0.0000000 0.0000000 0.0000000
```

```
[3,] 0.0 0.7130813 0.0000000 0.0000000
[4,] 0.0 0.0000000 0.7312994 0.7487683
>
>
$F
$F[[1]]
[,1] [,2] [,3] [,4]
[1,] 0 0.0000000 0.0000000 0.1696108
[2,] 0 0.2318637 0.2787757 0.1696108
[3,] 0 0.0000000 0.0000000 0.0000000
[4,] 0 0.0000000 0.0000000 0.0000000
```



```
$F[[2]]
            [,1] [,2] [,3] [,4]
[1,] 0 0.00000 0.000000 0.1695883
[2,] 0}0.23046 0.278352 0.1695883
> [3,] 0 0.00000 0.000000 0.0000000
[4,] 0 0.00000 0.000000 0.0000000
>
>
$ahstages
    stage_id stage original_size original_size_b original_size_c min_age
> 1 4 DormSeed 0 0 0
>2 1 Age1 NA NA NA N
> 3 2 Age2 NA NA NA N
4 3 Age3 NA NA NA 3
> max_age repstatus obsstatus propstatus immstatus matstatus entrystage
```



```
> 2 1rrrrrll
> 3 2 
> 4 NA 
    indataset binhalfwidth_raw sizebin_min sizebin_max sizebin_center
1 0 1 0.5 0.5
2 1 NA NA NA NA
3 1 NA NA NA NA
4 1 NA NA NA NA
    sizebin_width binhalfwidthb_raw sizebinb_min sizebinb_max sizebinb_center
l 1 0.5 1 1 0.5 0
>2 NA NA NA NA
> 3 NA NA NA NA
N NA NA NA NA NA
    sizebinb_width binhalfwidthc_raw sizebinc_min sizebinc_max sizebinc_center
\begin{tabular}{llllll}
1 & 0.5 & 1 & 0.5 & 1.5 & 1
\end{tabular}
2 NA NA NA NA NA
3 NA NA NA NA
4 NA NA NA NA NA
    sizebinc_width group comments alive almostborn
> 1 0.5 0 new stage 1 0
> NA 0 No description 1 0
> NA 0 No description 1 0
>4 NA 0 No description 1 0
```

```
>
$hstages
    X1
    1 NA
$agestages
    X1
1 NA
>
$labels
pop patch year2
1 1 1 1989
2 1 1 1990
>
$dataqc
    [1] 345 531
$matrixqc
[1] 10 6 2
$modelqc
vital_rate individuals transitions distribution accuracy
1 survival 345 531 binomial 0.7608286
> observation 0 0 NA
> size 0 0 gaussian NA
4 sizeb 0 0 NA
5 sizec 0 0 NA
6 reproduction 0 0 binomial N
7 fecundity 16 negbin 0.8343949
8 juvenile_survival 0
9 juvnile_observation
                juvenile_size
                    juvenile_sizeb
> 13 juvenile_reproduction 
>
> attr(,"class")
> [1] "lefkoMat"
```

The add_stage() function can be used to add new stages anywhere in the matrix, and can be used in stage-based ahistorical, historical, or age-by-stage MPMs, as well. For example, here we add a new juvenile stage to the age-by-stage Lathyrus MPM, lathmat2p_raw_ageedit. We will make this the new third stage, this time using the add_after option (this option can be used to add stages to the end of a matrix, as well).

```
lath_new_juv_mpm <- add_stage(lathmat2p_raw_ageedit, add_after = 2,
    stage_name = "NewJuv")
summary(lath_new_juv_mpm)
>
> This ahistorical lefkoMat object contains 12 matrices.
>
```

```
> Each matrix is square with 24 rows and columns, and a total of 576 elements.
> A total of 209 survival transitions were estimated, with 17.417 per matrix.
> A total of }12\mathrm{ fecundity transitions were estimated, with 1 per matrix.
> This lefkoMat object covers 1 population, }6\mathrm{ patches, and 2 time steps.
>
> The dataset contains a total of 345 unique individuals and 531 unique transitions.
>
> Survival probability sum check (each matrix represented by column in order):
> [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11] [,12]
> Min. 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000
> 1st Qu. 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000
> Median 0.000 0.399 0.000 0.450 0.000 0.842 0.000 0.599 0.000 0.562 0.000 0.000
> Mean 0.246 0.467 0.228 0.493 0.240 0.607 0.235 0.546 0.195 0.532 0.205 0.365
> 3rd Qu. 0.399 1.000 0.399 1.000 0.399 1.000 0.399 1.000 0.399 1.000 0.399 0.889
> Max. 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000
```

This summary suggests that three new rows and columns have been added. This reflects the fact that the stage structure of the stageframe is repeated in every age in an age-by-stage MPM. Since this object has three ages, that means three rows and columns have been added.

Let's see the ahstages object first.

| lath_new_juv_mpm\$ahstages |  |  |  |  |  |  |  |  |  |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
|  | stage_id | stage | original_size | original_si | ize_b | original_s | size_c | min_age | max_age |
| $>1$ | 1 | Sd | 0 |  | NA |  | NA | 0 | NA |
| $>2$ | 2 | Sdl | 100 |  | NA |  | NA | 0 | NA |
| $>3$ | 8 | NewJuv | 0 |  | 0 |  | 0 | 0 | 0 |
| $>4$ | 3 | VSm | 13 |  | NA |  | NA | 0 | NA |
| $>5$ | 4 | Sm | 127 |  | NA |  | NA | 0 | NA |
| $>6$ | 5 | VLa | 3730 |  | NA |  | NA | 0 | NA |
| $>7$ | 7 | Dorm | 0 |  | NA |  | NA | 0 | NA |
| $>8$ | 6 | Flo | 3800 |  | NA |  | NA | 0 | NA |
| repstatus obsstatus propstatus immstatus matstatus entrystage indataset |  |  |  |  |  |  |  |  |  |
| > 1 |  | 0 | $0 \quad 1$ | 1 |  | 0 | 1 |  | 0 |
| $>2$ |  | 0 | 10 | 1 |  | 0 | 1 |  | 1 |
| $>3$ |  | 1 | 10 | 0 |  | 1 | 0 |  | 0 |
| $>4$ |  | 0 | 10 | 0 |  | 1 | 0 |  | 1 |
| $>5$ |  | 0 | 10 | 0 |  | 1 | 0 |  | 1 |
| $>6$ |  | 0 | 10 | 0 |  | 1 | 0 |  | 1 |
| $>7$ |  | 0 | $0 \quad 0$ | 0 |  | 1 | 0 |  | 1 |
| $>8$ |  | 1 | 10 | 0 |  | 1 | 0 |  | 1 |
| > binhalfwidth_raw sizebin_min sizebin_max sizebin_center sizebin_width |  |  |  |  |  |  |  |  |  |
| > 1 |  | 0.0 | 0.0 | 0.0 |  | 0 |  | 0.0 |  |
| $>2$ |  | 100.0 | 0.0 | 200.0 |  | 100 |  | 200.0 |  |
| $>3$ |  | 1.0 | 0.5 | 1.5 |  | 1 |  | 0.5 |  |
| $>4$ |  | 11.0 | 2.0 | 24.0 |  | 13 |  | 22.0 |  |
| $>5$ |  | 103.0 | 24.0 | 230.0 |  | 127 |  | 206.0 |  |
| $>6$ |  | 3500.0 | 230.0 | 7230.0 |  | 3730 |  | 7000.0 |  |
| $>7$ |  | 0.5 | -0.5 | 0.5 |  | 0 |  | 1.0 |  |
| $>8$ |  | 3800.0 | 0.0 | 7600.0 |  | 3800 |  | 7600.0 |  |
| > binhalfwidthb_raw sizebinb_min sizebinb_max sizebinb_center sizebinb_width |  |  |  |  |  |  |  |  |  |
| > 1 |  |  | JA NA |  | NA |  | NA |  | NA |
| > 2 |  | N | NA NA |  | NA |  | NA |  | NA |


| > 3 | 1 | 0.5 | 1.5 | 1 | 0.5 |
| :---: | :---: | :---: | :---: | :---: | :---: |
| > 4 | NA | NA | NA | NA | NA |
| > 5 | NA | NA | NA | NA | NA |
| $>6$ | NA | NA | NA | NA | NA |
| $>7$ | NA | NA | NA | NA | NA |
| > 8 | NA | NA | NA | NA | NA |
| binhalfwidthc_raw sizebinc_min sizebinc_max sizebinc_center sizebinc_width |  |  |  |  |  |
| > 1 | NA | NA | NA | NA | NA |
| > 2 | NA | NA | NA | NA | NA |
| > 3 | 1 | 0.5 | 1.5 | 1 | 0.5 |
| > 4 | NA | NA | NA | NA | NA |
| $>5$ | NA | NA | NA | NA | NA |
| > 6 | NA | NA | NA | NA | NA |
| > 7 | NA | NA | NA | NA | NA |
| > 8 | NA | NA | NA | NA | NA |
|  | group comments | e almo |  |  |  |
|  | 0 No description | 1 |  |  |  |
| > 2 | 0 No description | 1 |  |  |  |
| > 3 | 0 new stage | 1 |  |  |  |
| $>4$ | 0 No description | 1 |  |  |  |
| $>5$ | 0 No description | 1 |  |  |  |
| > 6 | 0 No description | 1 |  |  |  |
| > 7 | 0 No description | 1 |  |  |  |
| > 8 | 0 No description | 1 |  |  |  |

We see that we have a new stage, named NewJuv and marked as stage 8, in the third row of this stageframe.

Let's now see the agestages object.
$l$
$l$ lath_new_juv_mpm\$agestages

| $>22$ | 5 | VLa | 3 |
| ---: | ---: | ---: | ---: |
| $>23$ | 7 | Dorm | 3 |
| $>24$ | 6 | Flo | 3 |

Our new stage appears in rows 3,11 , and 19 , corresponding to the rows and columns for that stage in the matrices.

Finally, let's take a look at a single matrix.

| lath_new_juv_mpm\$A[[1] ] |  |  |  |  |  |  |  |  |  |  |  |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| > | [,1] | [,2] | [,3] | [,4] | [,5] | ] [,6] | [,7] | [,8] | [,9] | [,10] | [,11] |
| $>$ [1,] | 0.000 | 0.0000000 | 0 | 0.0000000 | 0.0000 | 00.000 | 0 | 1.035 | 0.000 | 0 | 0 |
| $>$ [2,] | 0.000 | 0.0000000 | 0 | 0.0000000 | 0.0000 | 00.000 | 0 | 0.162 | 0.000 | 0 | 0 |
| $>$ [3,] | 0.000 | 0.0000000 | 0 | 0.0000000 | 0.0000 | 00.000 | 0 | 0.000 | 0.000 | 0 | 0 |
| $>$ [4,] | 0.000 | 0.0000000 | 0 | 0.0000000 | 0.0000 | 00.000 | 0 | 0.000 | 0.000 | 0 | 0 |
| $>$ [5,] | 0.000 | 0.0000000 | 0 | 0.0000000 | 0.0000 | 00.000 | 0 | 0.000 | 0.000 | 0 | 0 |
| $>$ [6,] | 0.000 | 0.0000000 | 0 | 0.0000000 | 0.0000 | 00.000 | 0 | 0.000 | 0.000 | 0 | 0 |
| $>$ [7,] | 0.000 | 0.0000000 | 0 | 0.0000000 | 0.0000 | 00.000 | 0 | 0.000 | 0.000 | 0 | 0 |
| > [8,] | 0.000 | 0.0000000 | 0 | 0.0000000 | 0.0000 | 00.000 | 0 | 0.000 | 0.000 | 0 | 0 |
| $>$ [9,] | 0.345 | 0.0000000 | 0 | 0.0000000 | 0.0000 | 00.000 | 0 | 0.000 | 0.000 | 0 | 0 |
| > [10,] | 0.000 | 0.0000000 | 0 | 0.0000000 | 0.0000 | 00.000 | 0 | 0.000 | 0.000 | 0 | 0 |
| > [11,] | 0.054 | 0.0000000 | 0 | 0.0000000 | 0.0000 | 00.000 | 0 | 0.000 | 0.000 | 0 | 0 |
| > [12,] | 0.000 | 0.9047619 | 0 | 0.7142857 | 0.1875 | 50.250 | 0 | 0.000 | 0.000 | 0 | 0 |
| > [13,] | 0.000 | 0.0000000 | 0 | 0.1428571 | 0.6875 | 50.500 | 0 | 1.000 | 0.000 | 0 | 0 |
| > [14,] | 0.000 | 0.0000000 | 0 | 0.0000000 | 0.0000 | 00.000 | 0 | 0.000 | 0.000 | 0 | 0 |
| $>$ [15,] | 0.000 | 0.0000000 | 0 | 0.0000000 | 0.0625 | 50.125 | 50 | 0.000 | 0.000 | 0 | 0 |
| > [16,] | 0.000 | 0.0000000 | 0 | 0.0000000 | 0.0000 | 00.125 | 50 | 0.000 | 0.000 | 0 | 0 |
| > [17,] | 0.000 | 0.0000000 | 0 | 0.0000000 | 0.0000 | 00.000 | 0 | 0.000 | 0.000 | 0 | 0 |
| > [18,] | 0.000 | 0.0000000 | 0 | 0.0000000 | 0.0000 | 00.000 | 0 | 0.000 | 0.345 | 0 | 0 |
| > [19,] | 0.000 | 0.0000000 | 0 | 0.0000000 | 0.0000 | 00.000 | 0 | 0.000 | 0.054 | 0 | 0 |
| > [20,] | 0.000 | 0.0000000 | 0 | 0.0000000 | 0.0000 | 00.000 | 0 | 0.000 | 0.000 | 0 | 0 |
| > [21,] | 0.000 | 0.0000000 | 0 | 0.0000000 | 0.0000 | 00.000 | 0 | 0.000 | 0.000 | 0 | 0 |
| > [22,] | 0.000 | 0.0000000 | 0 | 0.0000000 | 0.0000 | 00.000 | 0 | 0.000 | 0.000 | 0 | 0 |
| > [23,] | 0.000 | 0.0000000 | 0 | 0.0000000 | 0.0000 | 0.000 | 0 | 0.000 | 0.000 | 0 | 0 |
| > [24,] | 0.000 | 0.0000000 | 0 | 0.0000000 | 0.0000 | 00.000 | 0 | 0.000 | 0.000 | 0 | 0 |
| $>$ | [,12] | [,13] [,14] | ] [,1 | 5] [,16] | [,17] [ | [,18] | [,19] | [,20] | [,21] | [,22] | [,23] |
| > [1,] | 0 | 0 | 0 | 0 0 | 0 | 0.000 | 0 | 0 | 0 | 0 | 0 |
| $>$ [2,] | 0 | 0 | 0 | 00 | 00 | 0.000 | 0 | 0 | 0 | 0 | 0 |
| $>$ [3,] | 0 | 0 | 0 | 00 | 00 | 0.000 | 0 | 0 | 0 | 0 | 0 |
| $>$ [4,] | 0 | 0 | 0 | 00 | 00 | 0.000 | 0 | 0 | 0 | 0 | 0 |
| $>$ [5,] | 0 | 0 | 0 | 00 | 00 | 0.000 | 0 | 0 | 0 | 0 | 0 |
| $>$ [6,] | 0 | 0 | 0 | $0 \quad 0$ | 00 | 0.000 | 0 | 0 | 0 | 0 | 0 |
| > [7,] | 0 | 0 | 0 | 00 | 00 | 0.000 | 0 | 0 | 0 | 0 | 0 |
| > [8,] | 0 | 0 | 0 | 00 | 00 | 0.000 | 0 | 0 | 0 | 0 | 0 |
| $>$ [9,] | 0 | 0 | 0 | 00 | 00 | 0.000 | 0 | 0 | 0 | 0 | 0 |
| > [10,] | 0 | 0 | 0 | 00 | 00 | 0.000 | 0 | 0 | 0 | 0 | 0 |
| > [11,] | 0 | 0 | 0 | 00 | 00 | 0.000 | 0 | 0 | 0 | 0 | 0 |
| > [12,] | 0 | 0 | 0 | 00 | 00 | 0.000 | 0 | 0 | 0 | 0 | 0 |
| > [13,] | 0 | 0 | 0 | 00 | 00 | 0.000 | 0 | 0 | 0 | 0 | 0 |
| > [14,] | 0 | 0 | 0 | 00 | 00 | 0.000 | 0 | 0 | 0 | 0 | 0 |
| > [15,] | 0 | 0 | 0 | 00 | 00 | 0.000 | 0 | 0 | 0 | 0 | 0 |
| > [16,] | 0 | 0 | 0 | 00 | 00 | 0.000 | 0 | 0 | 0 | 0 | 0 |
| > [17,] | 0 | 0 | 0 | 00 | 00 | 0.000 | 0 | 0 | 0 | 0 | 0 |


| $>$ | $[18]$, | 0 | 0 | 0 | 0 | 0 | 0 | 0.345 | 0 | 0 | 0 | 0 |
| :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- |
| $>[19]$, | 0 | 0 | 0 | 0 | 0 | 0 | 0.054 | 0 | 0 | 0 | 0 | 0 |
| $>[20]$, | 0 | 0 | 0 | 0 | 0 | 0 | 0.000 | 0 | 0 | 0 | 0 | 0 |
| $>[21]$, | 0 | 0 | 0 | 0 | 0 | 0 | 0.000 | 0 | 0 | 0 | 0 | 0 |
| $>[22]$, | 0 | 0 | 0 | 0 | 0 | 0 | 0.000 | 0 | 0 | 0 | 0 | 0 |
| $>[23]$, | 0 | 0 | 0 | 0 | 0 | 0 | 0.000 | 0 | 0 | 0 | 0 | 0 |
| $>[24]$, | 0 | 0 | 0 | 0 | 0 | 0 | 0.000 | 0 | 0 | 0 | 0 | 0 |

$>\quad[, 24]$
$>\quad[1] \quad$,
$>\quad[2] \quad$,
$>\quad[3] \quad$,
$>[4] \quad$,
$>[5] \quad$,
$>\quad[6] \quad$,
$>\quad[7] \quad$,
$>\quad[8] \quad$,
$>\quad[9] \quad$,
$>$ [10,] 0
$>$ [11,] 0
$>[12] \quad$,
$>[13] \quad$,
$>$ [14,] 0
$>[15] \quad$,
$>[16] \quad$,
$>[17] \quad$,
$>[18] \quad$,
$>[19] \quad$,
$>[20] \quad$,
$>$ [21,] 0
$>$ [22,] 0
$>[23] \quad$,
$>[24] \quad$,

Close inspection will reveal that rows and columns 3, 11, and 19 are empty, filled only with zeros. Happy editing!

### 14.6 Large matrices

Let's try one more situation, this time dealing with large, sparse matrices. We will deal with a historical version of the Cypripedium MPM. First, let's create the MPM itself.

```
data(cypdata)
sizevector <- c(0, 0, 0, 0, 0, 0, 1, 3, 6, 11, 19.5)
stagevector <- c("SD", "P1", "P2", "P3", "SL", "D", "XSm", "Sm", "Md", "Lg",
    "XLg")
repvector <- c(0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1)
obsvector <- c(0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1)
matvector <- c(0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1)
immvector <- c(0, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0)
propvector <- c(1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
```

```
indataset <- c(0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1)
binvec <- c(0, 0, 0, 0, 0, 0.5, 0.5, 1.5, 1.5, 3.5, 5)
comments <- c("Dormant seed", "1st yr protocorm", "2nd yr protocorm",
    "3rd yr protocorm", "Seedling", "Dormant adult",
    "Extra small adult (1 shoot)", "Small adult (2-4 shoots)",
    "Medium adult (5-7 shoots)", "Large adult (8-14 shoots)",
    "Extra large adult (>14 shoots)")
cypframe_raw <- sf_create(sizes = sizevector, stagenames = stagevector,
    repstatus = repvector, obsstatus = obsvector, matstatus = matvector,
    propstatus = propvector, immstatus = immvector, indataset = indataset,
    binhalfwidth = binvec, comments = comments)
cypraw_v1 <- verticalize3(data = cypdata, noyears = 6, firstyear = 2004,
    patchidcol = "patch", individcol = "plantid", blocksize = 4,
    sizeacol = "Inf2.04", sizebcol = "Inf.04", sizeccol = "Veg.04",
    repstracol = "Inf.04", repstrbcol = "Inf2.04", fecacol = "Pod.04",
    stageassign = cypframe_raw, stagesize = "sizeadded", NAas0 = TRUE,
    NRasRep = TRUE, age_offset = 4)
seeds_per_fruit <- 5000
sl_mult <- 0.7
cypsupp3_raw <- supplemental(stage3 = c("SD", "SD", "P1", "P1", "P2", "P2",
        "P3", "SL", "SL", "SL", "D", "XSm", "Sm", "D", "XSm", "Sm", "mat", "mat",
        "mat", "SD", "P1"),
    stage2 = c("SD", "SD", "SD", "SD", "P1", "P1", "P2", "P3", "SL", "SL", "SL",
        "SL", "SL", "SL", "SL", "SL", "D", "XSm", "Sm", "rep", "rep"),
    stage1 = c("SD", "rep", "SD", "rep", "SD", "rep", "P1", "P2", "P3", "SL",
        "SL", "SL", "SL", "P3", "P3", "P3", "SL", "SL", "SL", "mat", "mat"),
    eststage3 = c(NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, "D", "XSm", "Sm", "D",
        "XSm", "Sm", "mat", "mat", "mat", NA, NA),
    eststage2 = c(NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, "XSm", "XSm", "XSm",
        "XSm", "XSm", "XSm", "D", "XSm", "Sm", NA, NA),
    eststage1 = c(NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, "XSm", "XSm", "XSm",
        "XSm", "XSm", "XSm", "XSm", "XSm", "XSm", NA, NA),
    givenrate = c(0.08, 0.10, 0.10, 0.10, 0.10, 0.10, 0.10, 0.05, 0.05, 0.05, NA,
        NA, NA, NA, NA, NA, NA, NA, NA, NA, NA),
    multiplier = c(NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, sl_mult, sl_mult,
        sl_mult, sl_mult, sl_mult, sl_mult, 1, 1, 1, 0.5 * seeds_per_fruit,
        0.5 * seeds_per_fruit),
    type =c(1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 3, 3),
    stageframe = cypframe_raw, historical = TRUE)
cypmatrix3rp <- rlefko3(data = cypraw_v1, stageframe = cypframe_raw,
    year = "all", patch = "all", stages = c("stage3", "stage2", "stage1"),
    size = c("size3added", "size2added", "size1added"), supplement = cypsupp3_raw,
    yearcol = "year2", patchcol = "patchid", indivcol = "individ")
summary(cypmatrix3rp)
>
> This historical lefkoMat object contains 12 matrices.
>
```

```
> Each matrix is square with }121\mathrm{ rows and columns, and a total of 14641 elements.
> A total of 516 survival transitions were estimated, with 43 per matrix.
> A total of }70\mathrm{ fecundity transitions were estimated, with 5.833 per matrix.
> This lefkoMat object covers 1 population, 3 patches, and 4 time steps.
>
> The dataset contains a total of 74 unique individuals and 320 unique transitions.
>
> Survival probability sum check (each matrix represented by column in order):
> [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11]
> Min. 0.000 0.0000 0.0000 0.000 0.000 0.000 0.00 0.000 0.000 0.0000 0.000
> 1st Qu. 0.000 0.0000 0.0000 0.000 0.000 0.000 0.00 0.000 0.000 0.0000 0.000
>Median 0.000 0.0000 0.0000 0.000 0.000 0.000 0.00 0.000 0.000 0.0000 0.000
> Mean 0. 0.107 0.0945 0.0851 0.101 0.158 0.158 0.14 0.169 0.119 0.0851 0.119
> 3rd Qu. 0.000 0.0000 0.0000 0.000 0.100 0.100 0.05 0.100 0.000 0.0000 0.000
> Max. 1.000 1.0000 1.0000 1.000 1.000 1.000 1.00 1.000 1.000 1.0000 1.000
> [,12]
> Min. 0.000
> 1st Qu. 0.000
> Median 0.000
> Mean 0.144
> 3rd Qu. 0.000
> Max. 1.000
```

We have 12 matrices, and each matrix has 121 rows and columns. On average, approximately 49 elements in each matrix are non-zero, making these matrices extremely sparse. Let's also take a look at the labels element to see the order of the matrices.

| cypmatrix3rp\$labels |  |  |  |
| :--- | ---: | ---: | ---: |
| $>$ | pop patch | year2 |  |
| $>$ | p | 1 | A |
| $\gg 2$ | 2005 |  |  |
| $>$ | 1 | A | 2006 |
| $>$ | 1 | A | 2007 |
| $>4$ | 1 | A | 2008 |
| $>5$ | 1 | B | 2005 |
| $>6$ | 1 | B | 2006 |
| $>7$ | 1 | B | 2007 |
| $>8$ | 1 | B | 2008 |
| $>9$ | 1 | C | 2005 |
| $>10$ | 1 | C | 2006 |
| $>11$ | 1 | C | 2007 |
| $>12$ | 1 | C | 2008 |

Let's suppose that we have some extra information that germination is higher in patch B than the other patches. We assumed that germination occurred at a rate of 0.10 typically, but let's say that we have data suggesting that in patch B , it is twice that value. In this case, we can use function edit_lM() to make this change, as below.

```
cypmatrix3rp_boostB <- edit_1M(cypmatrix3rp, patch = "B", stage3 = c("P1", "P1"),
    stage2 = c("rep", "SD"), stage1 = c("mat", "rep"), givenrate = c(NA, 0.2),
    multiplier = c(2, NA), type = c(2, 1))
summary(cypmatrix3rp_boostB)
>
```

```
> This historical lefkoMat object contains 12 matrices.
>
> Each matrix is square with }121\mathrm{ rows and columns, and a total of 14641 elements.
> A total of 516 survival transitions were estimated, with 43 per matrix.
> A total of 70 fecundity transitions were estimated, with 5.833 per matrix.
> This lefkoMat object covers 1 population, 3 patches, and 4 time steps.
>
> The dataset contains a total of }74\mathrm{ unique individuals and 320 unique transitions.
>
> Survival probability sum check (each matrix represented by column in order):
> [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11]
> Min. 0.000 0.0000 0.0000 0.000 0.000 0.000 0.000 0.000 0.000 0.0000 0.000
> 1st Qu. 0.000 0.0000 0.0000 0.000 0.000 0.000 0.000 0.000 0.000 0.0000 0.000
> Median 0.000 0.0000 0.0000 0.000 0.000 0.000 0.000 0.000 0.000 0.0000 0.000
> Mean 0.107 0.0945 0.0851 0.101 0.162 0.162 0.144 0.173 0.119 0.0851 0.119
> 3rd Qu. 0.000 0.0000 0.0000 0.000 0.100 0.100 0.050 0.100 0.000 0.0000 0.000
> Max. 1.000 1.0000 1.0000 1.000 1.000 1.000 1.000 1.000 1.000 1.0000 1.000
> [,12]
> Min. 0.000
> 1st Qu. 0.000
> Median 0.000
> Mean 0.144
> 3rd Qu. 0.000
> Max. 1.000
```

This summary shows a shift in the mean values for survival probabilities associated with the fifth through eighth matrices, suggesting that we have, indeed, edited the matrices. To quickly assess this, we might try code like the following, which simply looks for differences and summarizies them.

```
# Patch A
summary(as.vector(cypmatrix3rp_boostB$A[[1]] - cypmatrix3rp$A[[1]]))
> Min. 1st Qu. Median Mean 3rd Qu. Max.
> 0
summary(as.vector(cypmatrix3rp_boostB$A[[2]] - cypmatrix3rp$A[[2]]))
> Min. 1st Qu. Median Mean 3rd Qu. Max.
> 0
summary(as.vector(cypmatrix3rp_boostB$A[[3]] - cypmatrix3rp$A[[3]]))
> Min. 1st Qu. Median Mean 3rd Qu. Max.
> 0
summary(as.vector(cypmatrix3rp_boostB$A[[4]] - cypmatrix3rp$A[[4]]))
> Min. 1st Qu. Median Mean 3rd Qu. Max.
> 0
# Patch B
summary(as.vector(cypmatrix3rp_boostB$A[[5]] - cypmatrix3rp$A[[5]]))
> Min. 1st Qu. Median Mean 3rd Qu. Max.
> 0.000 0.000 0.000 3.311 0.000 17500.000
summary(as.vector(cypmatrix3rp_boostB$A[[6]] - cypmatrix3rp$A[[6]]))
> Min. 1st Qu. Median Mean 3rd Qu. Max.
lllllll
summary(as.vector(cypmatrix3rp_boostB$A[[7]] - cypmatrix3rp$A[[7]]))
> Min. 1st Qu. Median Mean 3rd Qu. Max.
```

```
> 0.00 0.00 0.00 0.37 0.00 2500.00
summary(as.vector(cypmatrix3rp_boostB$A[[8]] - cypmatrix3rp$A[[8]]))
> Min. 1st Qu. Median Mean 3rd Qu. Max.
> 0.0000 0.0000 0.0000 0.4269 0.0000 2500.0000
# Patch C
summary(as.vector(cypmatrix3rp_boostB$A[[9]] - cypmatrix3rp$A[[9]]))
> Min. 1st Qu. Median Mean 3rd Qu. Max.
> 0
summary(as.vector(cypmatrix3rp_boostB$A[[10]] - cypmatrix3rp$A[[10]]))
> Min. 1st Qu. Median Mean 3rd Qu. Max.
> 0
summary(as.vector(cypmatrix3rp_boostB$A[[11]] - cypmatrix3rp$A[[11]]))
> Min. 1st Qu. Median Mean 3rd Qu. Max.
> 0
summary(as.vector(cypmatrix3rp_boostB$A[[12]] - cypmatrix3rp$A[[12]]))
> Min. 1st Qu. Median Mean 3rd Qu. Max.
> 0
```

All looks good! The extremely large values in the maxima for the fifth through eighth matrices correspond to the doubling of fecundity there, which is a result of doubled germination.

Users may use further strengths in this approach, including the use of proxy stages and stage groups, just as in supplemental().

### 14.7 Points to remember

1. MPMs may be subset from larger lefkoMat objects using the subset_1M() function.
2. Matrices may be added or deleted with add_1M() and delete_1M().
3. Function edit_lM () allows matrices to be edited easily within lefkoMat objects, and these edits may be automated across groups of matrices or single matrices.

## Chapter 15

## Further Issues IV: Quality Control

When someone is honestly $55 \%$ right, that's very good and there's no use wrangling. And if someone is $60 \%$ right, it's wonderful, it's great luck, and let him thank God. But what's to be said about 75\% right? Wise people say this is suspicious. Well, and what about $100 \%$ right? Whoever says he's $100 \%$ right is a fanatic, a thug, and the worst kind of rascal.

- Quote assigned to an elder Jewish man from Galicia, The Captive Mind by Czeslaw Miłosz (1953), translation by Jane Zielonko

Stand up You've got to manage I won't sympathize any more. And if you complain once more You'll meet an army of me.
— Björk, Army of Me (1995)
Quality control is essential to making good inferences from matric projection analysis. Package lefko3 was made with quality control as a top priority. In this chapter, we will look at a number of the quality control tools and options that are included in this package.

### 15.1 Quality control in life history models and vertical datasets

Vertical datasets are standardized using the functions verticalize3() and historicalize3(). These functions also offer quality control options, particularly to assess whether the standardized datasets and the life history models match properly.

Let's try an example of using th quality control features of these functions. In the code below, we set up a stageframe for the Cypripedium dataset. However, we have deliberately assigned a smaller binwidth for the Small adult class. This will cause function verticalize3() to fail in stage assignment for some portion of the data. Let's see this in action.

```
data(cypdata)
sizevector <- c(0, 0, 0, 0, 0, 0, 1, 3, 6, 11, 19.5)
stagevector <- c("SD", "P1", "P2", "P3", "SL", "D", "XSm", "Sm", "Md", "Lg",
    "XLg")
repvector <- c(0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1)
obsvector <- c(0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1)
matvector <- c(0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1)
immvector <- c(0, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0)
propvector <- c(1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
indataset <- c(0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1)
binvec <- c(0, 0, 0, 0, 0, 0.5, 0.5, 0.5, 1.5, 3.5, 5) # 8th entry originally 1.5
```

```
comments <- c("Dormant seed", "1st yr protocorm", "2nd yr protocorm",
    "3rd yr protocorm", "Seedling", "Dormant adult",
    "Extra small adult (1 shoot)", "Small adult (2-4 shoots)",
    "Medium adult (5-7 shoots)", "Large adult (8-14 shoots)",
    "Extra large adult (>14 shoots)")
cypframe_raw <- sf_create(sizes = sizevector, stagenames = stagevector,
    repstatus = repvector, obsstatus = obsvector, matstatus = matvector,
    propstatus = propvector, immstatus = immvector, indataset = indataset,
    binhalfwidth = binvec, comments = comments)
cypraw_v1 <- verticalize3(data = cypdata, noyears = 6, firstyear = 2004,
    patchidcol = "patch", individcol = "plantid", blocksize = 4,
    sizeacol = "Inf2.04", sizebcol = "Inf.04", sizeccol = "Veg.04",
    repstracol = "Inf.04", repstrbcol = "Inf2.04", fecacol = "Pod.04",
    stageassign = cypframe_raw, stagesize = "sizeadded", NAas0 = TRUE,
    NRasRep = TRUE, age_offset = 4)
> Warning: Some stages occurring in the dataset do not match any characteristics
> in the input stageframe.
> Warning: Some stages occurring in the dataset do not match any characteristics
> in the input stageframe.
    Warning: Some stages occurring in the dataset do not match any characteristics
    in the input stageframe.
    Warning: Some stages occurring in the dataset do not match any characteristics
    in the input stageframe.
    Warning: Some stages occurring in the dataset do not match any characteristics
    in the input stageframe.
    Warning: Some stages occurring in the dataset do not match any characteristics
    in the input stageframe.
    Warning: Some stages occurring in the dataset do not match any characteristics
    in the input stageframe.
    Warning: Some stages occurring in the dataset do not match any characteristics
    in the input stageframe.
    Warning: Some stages occurring in the dataset do not match any characteristics
    in the input stageframe.
    Warning: Some stages occurring in the dataset do not match any characteristics
    in the input stageframe.
    Warning: Some stages occurring in the dataset do not match any characteristics
    in the input stageframe.
    Warning: Some stages occurring in the dataset do not match any characteristics
    in the input stageframe.
```

```
> Warning: Some stages occurring in the dataset do not match any characteristics
> in the input stageframe.
> Warning: Some stages occurring in the dataset do not match any characteristics
> in the input stageframe.
> Warning: Some stages occurring in the dataset do not match any characteristics
> in the input stageframe.
> Warning: Some stages occurring in the dataset do not match any characteristics
> in the input stageframe.
> Warning: Some stages occurring in the dataset do not match any characteristics
> in the input stageframe.
> Warning: Some stages occurring in the dataset do not match any characteristics
> in the input stageframe.
> Warning: Some stages occurring in the dataset do not match any characteristics
> in the input stageframe.
> Warning: Some stages occurring in the dataset do not match any characteristics
> in the input stageframe.
> Warning: Some stages occurring in the dataset do not match any characteristics
> in the input stageframe.
> Warning: Some stages occurring in the dataset do not match any characteristics
> in the input stageframe.
Warning: Some stages occurring in the dataset do not match any characteristics
in the input stageframe.
Warning: Some stages occurring in the dataset do not match any characteristics
in the input stageframe.
Warning: Some stages occurring in the dataset do not match any characteristics
in the input stageframe.
Warning: Some stages occurring in the dataset do not match any characteristics
in the input stageframe.
Warning: Some stages occurring in the dataset do not match any characteristics
in the input stageframe.
Warning: Some stages occurring in the dataset do not match any characteristics
in the input stageframe.
Warning: Some stages occurring in the dataset do not match any characteristics
in the input stageframe.
```

```
> Warning: Some stages occurring in the dataset do not match any characteristics
> in the input stageframe.
> Warning: Some stages occurring in the dataset do not match any characteristics
> in the input stageframe.
> Warning: Some stages occurring in the dataset do not match any characteristics
in the input stageframe.
Warning: Some stages occurring in the dataset do not match any characteristics
in the input stageframe.
Warning: Some stages occurring in the dataset do not match any characteristics
in the input stageframe.
Warning: Some stages occurring in the dataset do not match any characteristics
in the input stageframe.
Warning: Some stages occurring in the dataset do not match any characteristics
in the input stageframe.
Warning: Some stages occurring in the dataset do not match any characteristics
in the input stageframe.
Warning: Some stages occurring in the dataset do not match any characteristics
in the input stageframe.
Warning: Some stages occurring in the dataset do not match any characteristics
in the input stageframe.
Warning: Some stages occurring in the dataset do not match any characteristics
in the input stageframe.
Warning: Some stages occurring in the dataset do not match any characteristics
in the input stageframe.
Warning: Some stages occurring in the dataset do not match any characteristics
in the input stageframe.
Warning: Some stages occurring in the dataset do not match any characteristics
in the input stageframe.
Warning: Some stages occurring in the dataset do not match any characteristics
in the input stageframe.
Warning: Some stages occurring in the dataset do not match any characteristics
in the input stageframe.
Warning: Some stages occurring in the dataset do not match any characteristics
in the input stageframe.
```

```
> Warning: Some stages occurring in the dataset do not match any characteristics
> in the input stageframe.
> Warning: Some stages occurring in the dataset do not match any characteristics
> in the input stageframe.
> Warning: Some stages occurring in the dataset do not match any characteristics
in the input stageframe.
Warning: Some stages occurring in the dataset do not match any characteristics
in the input stageframe.
Warning: Some stages occurring in the dataset do not match any characteristics
in the input stageframe.
> Warning: Some stages occurring in the dataset do not match any characteristics
> in the input stageframe.
Warning: Some stages occurring in the dataset do not match any characteristics
in the input stageframe.
Warning: Some stages occurring in the dataset do not match any characteristics
in the input stageframe.
Warning: Some stages occurring in the dataset do not match any characteristics
in the input stageframe.
Warning: Some stages occurring in the dataset do not match any characteristics
in the input stageframe.
Warning: Some stages occurring in the dataset do not match any characteristics
in the input stageframe.
Warning: Some stages occurring in the dataset do not match any characteristics
in the input stageframe.
Warning: Some stages occurring in the dataset do not match any characteristics
in the input stageframe.
Warning: Some stages occurring in the dataset do not match any characteristics
in the input stageframe.
Warning: Some stages occurring in the dataset do not match any characteristics
in the input stageframe.
Warning: Some stages occurring in the dataset do not match any characteristics
in the input stageframe.
Warning: Some stages occurring in the dataset do not match any characteristics
in the input stageframe.
```

```
> Warning: Some stages occurring in the dataset do not match any characteristics
> in the input stageframe.
> Warning: Some stages occurring in the dataset do not match any characteristics
> in the input stageframe.
Warning: Some stages occurring in the dataset do not match any characteristics
in the input stageframe.
Warning: Some stages occurring in the dataset do not match any characteristics
in the input stageframe.
Warning: Some stages occurring in the dataset do not match any characteristics
in the input stageframe.
Warning: Some stages occurring in the dataset do not match any characteristics
in the input stageframe.
Warning: Some stages occurring in the dataset do not match any characteristics
in the input stageframe.
Warning: Some stages occurring in the dataset do not match any characteristics
in the input stageframe.
Warning: Some stages occurring in the dataset do not match any characteristics
in the input stageframe.
Warning: Some stages occurring in the dataset do not match any characteristics
in the input stageframe.
Warning: Some stages occurring in the dataset do not match any characteristics
in the input stageframe.
Warning: Some stages occurring in the dataset do not match any characteristics
in the input stageframe.
Warning: Some stages occurring in the dataset do not match any characteristics
in the input stageframe.
Warning: Some stages occurring in the dataset do not match any characteristics
in the input stageframe.
Warning: Some stages occurring in the dataset do not match any characteristics
in the input stageframe.
Warning: Some stages occurring in the dataset do not match any characteristics
in the input stageframe.
Warning: Some stages occurring in the dataset do not match any characteristics
in the input stageframe.
```

```
> Warning: Some stages occurring in the dataset do not match any characteristics
> in the input stageframe.
> Warning: Some stages occurring in the dataset do not match any characteristics
> in the input stageframe.
> Warning: Some stages occurring in the dataset do not match any characteristics
> in the input stageframe.
> Warning: Some stages occurring in the dataset do not match any characteristics
> in the input stageframe.
> Warning: Some stages occurring in the dataset do not match any characteristics
> in the input stageframe.
> Warning: Some stages occurring in the dataset do not match any characteristics
> in the input stageframe.
> Warning: Some stages occurring in the dataset do not match any characteristics
> in the input stageframe.
```

The output shows quite a few repeated warnings about some stages in the dataset not matching the input life history model as programmed in the stageframe. Let's take a look more closely, using the summary_hfv() function. We will suppress the output as the data frame of errors is quite large.
$>$
> This hfv dataset contains 320 rows, 57 variables, 1 population,
> 3 patches, 74 individuals, and 5 time steps.
> Problems in stage assignment identified in rows:
$>$
$>\begin{array}{ccccccccccccccccccc} & {[1]} & 2 & 5 & 9 & 10 & 11 & 12 & 14 & 16 & 19 & 21 & 24 & 26 & 27 & 28 & 29 & 33 & 34 \\ 38\end{array}$

| $>$ | $[19]$ | 39 | 40 | 42 | 46 | 48 | 49 | 52 | 53 | 54 | 59 | 64 | 66 | 69 | 73 | 74 | 75 | 76 | 78 |
| :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- |


| $>$ | $[37]$ | 80 | 83 | 85 | 88 | 90 | 91 | 92 | 93 | 94 | 98 | 99 | 101 | 102 | 103 | 104 | 105 | 106 |
| :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | 107

> [55] 1111 | 112 | 113 | 114 | 115 | 116 | 117 | 118 | 119 | 120 | 125 | 129 | 131 | 134 | 135 | 136 | 140 | 141 |
| :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- |


> $\left[\begin{array}{llllllllllllllllllll}181] & 172 & 173 & 174 & 178 & 179 & 180 & 181 & 182 & 183 & 184 & 185 & 188 & 190 & 194 & 195 & 199 & 200 & 204\end{array}\right.$
> [109] 205209212213216220221222223224229232233234236237239241

> [145] 272274275276283285286291294295296298300303304306309310
> [163] 312316317318319320
> rowid popid patchid individ year2
$>$ Min. : 1.00 Length:320 A: 93 Min. : 164.0 Min. :200
$>$ 1st Qu.:21.00 Class :character B:154 1st Qu.: 391.0 1st Qu.:2005
$>$ Median :37.50 Mode :character C: 73 Median : 453.0 Median :2006
$>$ Mean $: 38.45$ Mean : 651.5 Mean :2006
$>$ 3rd Qu.:56.00
$>$ Max. :77.00
$>$ firsteen
> Min. :2004 Min. :2004
> 1st Qu.:2004 1st Qu.:2009
> Median :2004 Median :2009
> Mean :2004 Mean :2009 Mean :6.853 Mean :4.556






The output above shows, from the third line down a few lines, rows in the standardized dataset that have problems in stage assignment. We can use this output to take a look at some of these rows and try to determine where our mistake is, as below.



In the output above, we find that the five rows we have chosen to investigate show NoMatch under the stage 3 column, meaning that R could not assign stages in time $t+1$ here. In these five cases, the individuals were observable and mature, though they could be reproductive or not. The size seems to be the common feature, which is 2 for all (see the size3added column).

Let's see if we can get to the bottom of the problem by looking at the stageframe.

| cypframe_raw |  |  |  |  |  |  |  |  |  |  |  |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| > | stage | size | size_b | size_c | min_age | max_age | repsta | tus | obsstatu |  | propstatus |
| $>1$ | SD | 0.0 | NA | NA | NA | NA |  | 0 |  | 0 | 1 |
| $>2$ | P1 | 0.0 | NA | NA | NA | NA |  | 0 |  | 0 | 0 |
| $>3$ | P2 | 0.0 | NA | NA | NA | NA |  | 0 |  | 0 | 0 |
| $>4$ | P3 | 0.0 | NA | NA | NA | NA |  | 0 |  | 0 | 0 |
| $>5$ | SL | 0.0 | NA | NA | NA | NA |  | 0 |  | 0 | 0 |
| $>6$ | D | 0.0 | NA | NA | NA | NA |  | 0 |  | 0 | 0 |
| $>7$ | XSm | 1.0 | NA | NA | NA | NA |  | 1 |  | 1 | 0 |
| $>8$ | Sm | 3.0 | NA | NA | NA | NA |  | 1 |  | 1 | 0 |
| $>9$ | Md | 6.0 | NA | NA | NA | NA |  | 1 |  | 1 | 0 |
| > 10 |  | 11.0 | NA | NA | NA | NA |  | 1 |  | 1 | 0 |
| > 11 | XLg | 19.5 | NA | NA | NA | NA |  | 1 |  | 1 | 0 |
| > | immstatus matstatus indataset binhalfwidth_raw sizebin_min sizebin_max |  |  |  |  |  |  |  |  |  |  |
| > 1 |  | 0 | 0 | 0 | 0 |  | 0.0 |  | 0.0 |  | 0.0 |
| $>2$ |  | 1 | 0 | 0 | 0 |  | 0.0 |  | 0.0 |  | 0.0 |
| $>3$ |  | 1 | 0 | 0 | 0 |  | 0.0 |  | 0.0 |  | 0.0 |
| $>4$ |  | 1 | 0 | 0 | 0 |  | 0.0 |  | 0.0 |  | 0.0 |
| $>5$ |  | 1 | 0 | 0 | 0 |  | 0.0 |  | 0.0 |  | 0.0 |
| > 6 |  | 0 | 1 | 1 | 1 |  | 0.5 |  | -0.5 |  | 0.5 |



The key to assessing where the problem lies is in assessing what is missing from the size bins here. To assess this, we can look at the sizebin_min and sizebin_max columns. Doing so shows us that, in the adult stages, stage XSm ranges in size from 0.5 to 1.5 , and the next bigger stage ranges in size from 2.5 to 3.5 . In fact, looking further, we also see that stage Md ranges in size from 4.5 to 7.5 , meaning that a size of 4 is also not included in any stage. With this knowledge in hand, we can revise our stageframe to expand the bin width of stage Sm an extra 2 sprouts, as below.

```
binvec <- c(0, 0, 0, 0, 0, 0.5, 0.5, 1.5, 1.5, 3.5, 5)
cypframe_raw <- sf_create(sizes = sizevector, stagenames = stagevector,
    repstatus = repvector, obsstatus = obsvector, matstatus = matvector,
```

```
propstatus = propvector, immstatus = immvector, indataset = indataset,
binhalfwidth = binvec, comments = comments)
cypraw_v2 <- verticalize3(data = cypdata, noyears = 6, firstyear = 2004,
    patchidcol = "patch", individcol = "plantid", blocksize = 4,
    sizeacol = "Inf2.04", sizebcol = "Inf.04", sizeccol = "Veg.04",
    repstracol = "Inf.04", repstrbcol = "Inf2.04", fecacol = "Pod.04",
    stageassign = cypframe_raw, stagesize = "sizeadded", NAas0 = TRUE,
    NRasRep = TRUE, age_offset = 4)
summary_hfv(cypraw_v2)
>
> This hfv dataset contains }320\mathrm{ rows, 57 variables, 1 population,
> 3 patches, }74\mathrm{ individuals, and 5 time steps.
> rowid popid patchid individ year2
> Min. : 1.00 Length:320 A: 93 Min. : 164.0 Min. :2004
> 1st Qu.:21.00 Class :character B:154 1st Qu.: 391.0 1st Qu.:2005
> Median :37.50 Mode :character C: 73 Median : 453.0 Median :2006
> Mean :38.45 Mean : 651.5 Mean :2006
> 3rd Qu.:56.00 3rd Qu.: 476.0 3rd Qu.:2007
> Max. :77.00 Max. :1560.0 Max. :2008
> firstseen lastseen obsage obslifespan
> Min. :2004 Min. :2004 Min. :5.000 Min. :0.000
> 1st Qu.:2004 1st Qu.:2009 1st Qu.:6.000 1st Qu.:5.000
> Median :2004 Median :2009 Median :7.000 Median :5.000
> Mean :2004 Mean :2009 Mean :6.853 Mean :4.556
> 3rd Qu.:2004 3rd Qu.:2009 3rd Qu.:8.000 3rd Qu.:5.000
> Max. :2008 Max. :2009 Max. :9.000 Max. :5.000
> sizea1 sizeb1 sizec1 size1added
>Min. :0.000000 Min. : 0.0000 Min. : 0.0 Min. : 0.000
> 1st Qu.:0.000000 1st Qu.: 0.0000 1st Qu.: 0.0 1st Qu.: 0.000
> Median :0.000000 Median : 0.0000 Median : 1.0 Median : 2.000
> Mean :0.009375 Mean : 0.7469 Mean : 1.9 Mean : 2.656
> 3rd Qu.:0.000000 3rd Qu.: 1.0000 3rd Qu.: 3.0 3rd Qu.: 4.000
> Max. :1.000000 Max. :18.0000 Max. :13.0 Max. :21.000
repstra1 repstrb1 repstr1added feca1
Min. : 0.0000 Min. :0.000000 Min. : 0.0000 Min. :0.0000
1st Qu.: 0.0000 1st Qu.:0.000000 1st Qu.: 0.0000 1st Qu.:0.0000
Median : 0.0000 Median :0.000000 Median : 0.0000 Median :0.0000
Mean : 0.7469 Mean :0.009375 Mean : 0.7562 Mean :0.2656
3rd Qu.: 1.0000 3rd Qu.:0.000000 3rd Qu.: 1.0000 3rd Qu.:0.0000
Max. :18.0000 Max. :1.000000 Max. :18.0000 Max. :7.0000
        fec1added obsstatus1 repstatus1 fecstatus1
    Min. :0.0000 Min. :0.0000 Min. :0.0000 Min. :0.0000
    1st Qu.:0.0000 1st Qu.:0.0000 1st Qu.:0.0000 1st Qu.:0.0000
    Median :0.0000 Median :1.0000 Median :0.0000 Median :0.0000
    Mean :0.2656 Mean :0.7469 Mean :0.2875 Mean :0.1344
    3rd Qu.:0.0000 3rd Qu.:1.0000 3rd Qu.:1.0000 3rd Qu.:0.0000
    Max. :7.0000 Max. :1.0000 Max. :1.0000 Max. :1.0000
        matstatus1 alive1
    Min. :0.0000 Min. :0.0000
    1st Qu.:1.0000 1st Qu.:1.0000 Class :character 1st Qu.: 6.000
```



| > | 3rd Qu.:0.0000 | 3rd Qu.:1.0 3rd | Qu.:1.0 | 3rd Qu | :0.0000 | 3rd Qu. : 1 |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| > | Max. :8.0000 | Max. :1.0 Max. | :1.0 | Max. | $: 1.0000$ | Max. 1 |
| > | alive3 | stage3 | stage3index |  |  |  |
| > | Min. 0.0000 | Length:320 | Min. | 0.000 |  |  |
| > | 1st Qu.:1.0000 | Class : character | 1st Qu | 7.000 |  |  |
| > | Median :1.0000 | Mode :character | Median | 8.000 |  |  |
| > | Mean :0.9469 |  | Mean | 7.544 |  |  |
| > | 3rd Qu.:1.0000 |  | 3rd Qu | 8.000 |  |  |
| > | Max. 1.0000 |  | Max. | 11.000 |  |  |

We no longer see any issues popping up in the summary_hfv() output.
In addition to the above, the function $h f v_{\mathrm{z}} \mathrm{qc}()$ is extremely useful in assessing the quality of our data. Let's use this function to explore our vertical dataset.

```
hfv_qc(cypraw_v2)
> Survival:
>
> Data subset has 58 variables and 320 transitions.
>
> Variable alive3 has O missing values.
> Variable alive3 is a binomial variable.
>
>
> Primary size:
>
> Data subset has 58 variables and 303 transitions.
>
> Variable sizea3 has 0 missing values.
> Variable sizea3 appears to be an integer variable.
>
> Variable sizea3 is fully non-negative.
>
> Overdispersion test:
> Mean sizea3 is 0.009901
> The variance in sizea3 is 0.009835
> The probability of this dispersion level by chance assuming that
> the true mean sizea3 = variance in sizea3,
> and an alternative hypothesis of overdispersion, is 1
> Dispersion level in sizea3 matches expectation.
>
> Zero-inflation and truncation tests:
> Mean lambda in sizea3 is 0.9901
> The actual number of 0s in sizea3 is 300
> The expected number of 0s in sizea3 under the null hypothesis is 300
> The probability of this deviation in Os from expectation by chance is 0.9025
> Variable sizea3 is not significantly zero-inflated.
>
>
> Fecundity:
>
> Data subset has 58 variables and 320 transitions.
```

```
> Variable feca2 has 0 missing values.
> Variable feca2 appears to be an integer variable.
>
> Variable feca2 is fully non-negative.
>
> Overdispersion test:
> Mean feca2 is 0.2906
> The variance in feca2 is 0.7084
> The probability of this dispersion level by chance assuming that
> the true mean feca2 = variance in feca2,
> and an alternative hypothesis of overdispersion, is 1
> Dispersion level in feca2 matches expectation.
>
> Zero-inflation and truncation tests:
> Mean lambda in feca2 is 0.7478
> The actual number of 0s in feca2 is 270
> The expected number of 0s in feca2 under the null hypothesis is 239.3
> The probability of this deviation in Os from expectation by chance is 2.189e-26
> Variable feca2 is significantly zero-inflated.
```

The output gives us quite a lot to work with. All of the variables that we might be interested in assessing as vital rates are examined. Naturally, variables coding for probabilities need to be binomial, and so we see that we have variables tested for whether they fit the characteristics of a binomial variable. We see that size and fecundity are explored to assess whether they fit the characteristics required of the associated distribution. So, they are examined for whether they are count variables or continuous, and they are also assessed to see whether they match the characteristics of distributions such as the Gaussian, the Poisson, and the negative binomial. Lastly, we see that the output includes information about the data subsets that will be used to assess the various vital rates, including the numbers of individuals and the the numbers of transitions (standardized dataset rows) to parameterize the vital rate models.

### 15.2 Quality control in vital rate models

The function modelsearch (), and its associated summary () function for lefkoMod objects, both provide critical quality control for vital rate models used to develop function-based MPMs, including discretized MPMs. The two key processes are actually conducted by function modelsearch() itself, but summary () provides easy access to the results. In particular, modelsearch() assesses the numbers of individuals and transitions used to develop each vital rate model, and the overall accuracy of each model.

Let's take a look at how this works using the function-based version of the Cypripedium analysis, as given in Chapter 5. Here, we load all of the preliminaries for the historical analysis.

```
data(cypdata)
stagevector <- c("SD", "P1", "P2", "P3", "SL", "D", "V1", "V2", "V3", "V4",
    "V5", "V6", "V7", "V8", "V9", "V10", "V11", "V12", "V13", "V14", "V15", "V16",
    "V17", "V18", "V19", "V20", "V21", "V22", "V23", "V24", "F1", "F2", "F3",
    "F4", "F5", "F6", "F7", "F8", "F9", "F10", "F11", "F12", "F13", "F14", "F15",
    "F16", "F17", "F18", "F19", "F20", "F21", "F22", "F23", "F24")
indataset <- c(0, 0, 0, 0, 0, rep(1, 49))
sizevector <- c(0, 0, 0, 0, 0, seq(from = 0, t = 24), seq(from = 1, to = 24))
```

```
repvector <- c(0, 0, 0, 0, 0, rep(0, 25), rep(1, 24))
obsvector <- c(0, 0, 0, 0, 0, 0, rep(1, 48))
matvector <- c(0, 0, 0, 0, 0, rep(1, 49))
immvector <- c(0, 1, 1, 1, 1, rep(0, 49))
propvector <- c(1, rep(0, 53))
comments <- c("Dormant seed", "Yr1 protocorm", "Yr2 protocorm", "Yr3 protocorm",
    "Seedling", "Veg dorm", "Veg adult 1 stem", "Veg adult 2 stems",
    "Veg adult 3 stems", "Veg adult 4 stems", "Veg adult 5 stems",
    "Veg adult 6 stems", "Veg adult 7 stems", "Veg adult 8 stems",
    "Veg adult 9 stems", "Veg adult 10 stems", "Veg adult 11 stems",
    "Veg adult 12 stems", "Veg adult 13 stems", "Veg adult 14 stems",
    "Veg adult 15 stems", "Veg adult 16 stems", "Veg adult 17 stems",
    "Veg adult 18 stems", "Veg adult 19 stems", "Veg adult 20 stems",
    "Veg adult 21 stems", "Veg adult 22 stems", "Veg adult 23 stems",
    "Veg adult }24\mathrm{ stems", "Flo adult 1 stem", "Flo adult 2 stems",
    "Flo adult 3 stems", "Flo adult 4 stems", "Flo adult 5 stems",
    "Flo adult 6 stems", "Flo adult 7 stems", "Flo adult 8 stems",
    "Flo adult }9\mathrm{ stems", "Flo adult }10\mathrm{ stems", "Flo adult 11 stems",
    "Flo adult 12 stems", "Flo adult 13 stems", "Flo adult 14 stems",
    "Flo adult 15 stems", "Flo adult 16 stems", "Flo adult 17 stems",
    "Flo adult }18\mathrm{ stems", "Flo adult 19 stems", "Flo adult 20 stems",
    "Flo adult 21 stems", "Flo adult 22 stems", "Flo adult 23 stems",
    "Flo adult 24 stems")
cypframe_fb <- sf_create(sizes = sizevector, stagenames = stagevector,
    repstatus = repvector, obsstatus = obsvector, matstatus = matvector,
    propstatus = propvector, immstatus = immvector, indataset = indataset,
    comments = comments)
cypfb_v1 <- verticalize3(data = cypdata, noyears = 6, firstyear = 2004,
    patchidcol = "patch", individcol = "plantid", blocksize = 4,
    sizeacol = "Inf2.04", sizebcol = "Inf.04", sizeccol = "Veg.04",
    repstracol = "Inf.04", repstrbcol = "Inf2.04", fecacol = "Pod.04",
    stageassign = cypframe_fb, stagesize = "sizeadded", NAas0 = TRUE,
    age_offset = 4)
seeds_per_fruit <- 5000
sl_mult <- 0.7
cypsupp3_fb <- supplemental(stage3 = c("SD", "SD", "P1", "P1", "P2", "P3", "SL",
        "SL", "SL", "D", "V1", "V2", "V3", "D", "V1", "V2", "V3", "mat", "mat",
        "mat", "mat", "SD", "P1"),
    stage2 = c("SD", "SD", "SD", "SD", "P1", "P2", "P3", "SL", "SL", "SL", "SL",
        "SL", "SL", "SL", "SL", "SL", "SL", "D", "V1", "V2", "V3", "rep", "rep"),
    stage1 = c("SD", "rep", "SD", "rep", "SD", "P1", "P2", "P3", "SL", "P3", "P3",
        "P3", "P3", "SL", "SL", "SL", "SL", "SL", "SL", "SL", "SL", "mat", "mat"),
    eststage3 = c(NA, NA, NA, NA, NA, NA, NA, NA, NA, "D", "V1", "V2", "V3", "D",
        "V1", "V2", "V3", "mat", "mat", "mat", "mat", NA, NA),
    eststage2 = c(NA, NA, NA, NA, NA, NA, NA, NA, NA, "D", "D", "D", "D", "D",
        "D", "D", "D", "D", "V1", "V2", "V3", NA, NA),
    eststage1 = c(NA, NA, NA, NA, NA, NA, NA, NA, NA, "D", "D", "D", "D", "D",
```

```
    "D", "D", "D", "V1", "V1", "V1", "V1", NA, NA),
givenrate = c(0.08, 0.08, 0.1, 0.1, 0.1, 0.1, 0.1, 0.05, 0.05, NA, NA, NA, NA,
    NA, NA, NA, NA, NA, NA, NA, NA, NA, NA),
multiplier = c(NA, NA, NA, NA, NA, NA, NA, NA, NA, sl_mult, sl_mult,
    sl_mult, sl_mult, sl_mult, sl_mult, sl_mult, sl_mult, 1, 1, 1, 1,
    0.5 * seeds_per_fruit, 0.5 * seeds_per_fruit),
type = c("S", "S", "S", "S", "S", "S", "S", "S", "S", "S", "S", "S", "S", "S",
    "S", "S", "S", "S", "S", "S", "S", "R", "R"),
type_t12 = c("S", "F", "S", "F", "S", "S", "S", "S", "S", "S", "S", "S", "S",
    "S", "S", "S", "S", "S", "S", "S", "S", "S", "S"),
stageframe = cypframe_fb, historical = TRUE)
```

Now let's run the vital rate models for a historical MPM.

```
cypmodels3p <- modelsearch(cypfb_v1, historical = TRUE, approach = "mixed",
    vitalrates = c("surv", "obs", "size", "repst", "fec"), patch = "patchid",
    sizedist = "negbin", size.trunc = TRUE, fecdist = "poisson", fec.zero = TRUE,
    suite = "main", size = c("size3added", "size2added", "size1added"),
    quiet = "partial")
```

Let's take a peek at the summary of the resulting lefkoMod object.

```
summary(cypmodels3p)
This LefkoMod object includes 5 linear models.
Best-fit model criterion used: aicc&k
>
>
>
Survival model:
Generalized linear mixed model fit by maximum likelihood (Laplace
    Approximation) [glmerMod]
    Family: binomial ( logit )
    Formula: alive3 ~ size2added + (1 | year2) + (1 | patchid) + (1 | individ)
        Data: subdata
            AIC BIC logLik deviance df.resid
    130.1321 148.9737-60.0660 120.1321 315
    Random effects:
    Groups Name Std.Dev.
    individ (Intercept) 1.199e+00
    year2 (Intercept) 5.117e-05
    patchid (Intercept) 1.172e-05
    Number of obs: 320, groups: individ, 74; year2, 5; patchid, 3
    Fixed Effects:
    (Intercept) size2added
        2.0356 0.6343
    optimizer (Nelder_Mead) convergence code: O (OK) ; 0 optimizer warnings; 1 lme4 warnings
    Observation model:
    Generalized linear mixed model fit by maximum likelihood (Laplace
```

```
Approximation) [glmerMod]
Family: binomial ( logit )
Formula: obsstatus3 ~ size2added + (1 | year2) + (1 | patchid) + (1 |
            individ)
        Data: subdata
            AIC BIC logLik deviance df.resid
120.2567 138.8254 -55.1284 110.2567 298
Random effects:
Groups Name Std.Dev.
individ (Intercept) 0.0000
year2 (Intercept) 0.8776
    patchid (Intercept) 0.0000
Number of obs: 303, groups: individ, 70; year2, 5; patchid, 3
Fixed Effects:
(Intercept) size2added
> 2.4904 0.3134
optimizer (Nelder_Mead) convergence code: O (OK) ; 0 optimizer warnings; 1 lme4 warnings
>
Size model:
Formula: size3added ~ (1 | year2) + (1 | patchid) + (1 | individ)
Data: subdata
    AIC BIC logLik df.resid
1009.9750 1028.2898-499.9875 283
Random-effects (co)variances:
Conditional model:
    Groups Name Std.Dev.
    year2 (Intercept) 0.1133
    patchid (Intercept) 0.2118
    individ (Intercept) 1.0320
Number of obs: 288 / Conditional model: year2, 5; patchid, 3; individ, 70
Dispersion parameter for truncated_nbinom2 family (): 2.73e+07
Fixed Effects:
Conditional model:
(Intercept)
        0.587
    Secondary size model:
    [1] 1
>
>
>
> Tertiary size model:
```

$>$
$>$

```
> [1] 1
>
>
>
Reproductive status model:
Generalized linear mixed model fit by maximum likelihood (Laplace
    Approximation) [glmerMod]
    Family: binomial ( logit )
Formula: repstatus3 ~ repstatus2 + size2added + (1 | year2) + (1 | patchid) +
            (1 | individ)
        Data: subdata
            AIC BIC logLik deviance df.resid
    333.4037 355.3815 -160.7019 321.4037 282
Random effects:
Groups Name Std.Dev.
individ (Intercept) 0.1776
year2 (Intercept) 0.6636
    patchid (Intercept) 0.3501
Number of obs: 288, groups: individ, 70; year2, 5; patchid, 3
Fixed Effects:
(Intercept) repstatus2 size2added
            -1.3836 1.5543 0.1788
```



```
>
>
> Fecundity model:
Formula:
feca2 ~ size2added + (1 | year2) + (1 | patchid) + (1 | individ)
Zero inflation:
~size2added + (1 | year2) + (1 | patchid) + (1 | individ)
Data: subdata
AIC BIC logLik df.resid
251.4551 279.1619-115.7275 108
Random-effects (co)variances:
```



```
Conditional model:
Groups Name Std.Dev.
year2 (Intercept) 5.610e-01
    patchid (Intercept) 2.283e-01
    individ (Intercept) 4.630e-08
```



```
Zero-inflation model:
Groups Name Std.Dev.
year2 (Intercept) 3.340e-07
    patchid (Intercept) 1.724e-12
    individ (Intercept) 2.057e-04
>
> Number of obs: 118 / Conditional model: year2, 5; patchid, 3; individ, 51 / Zero-inflation model: y
Fixed Effects:
>
```

```
Conditional model:
(Intercept) size2added
> -0.56501 0.06247
>
Zero-inflation model:
> (Intercept) size2added
> 3.840 -1.588
>
>
Juvenile survival model:
[1] 1
>
>
>
> Juvenile observation model:
> [1] 1
>
>
>
Juvenile size model:
[1] 1
>
>
>
Juvenile secondary size model:
> [1] 1
>
>
>
Juvenile tertiary size model:
[1] 1
>
>
>
> Juvenile reproduction model:
> [1] 1
>
>
>
> Juvenile maturity model:
> [1] 1
>
>
>
>
>
Number of models in survival table: 16
Number of models in observation table: 16
>
Number of models in size table: 16
```

```
>
Number of models in secondary size table: 1
>
Number of models in tertiary size table: 1
Number of models in reproduction status table: 16
Number of models in fecundity table: 241
Number of models in juvenile survival table: 1
Number of models in juvenile observation table: 1
Number of models in juvenile size table: 1
Number of models in juvenile secondary size table: 1
Number of models in juvenile tertiary size table: 1
Number of models in juvenile reproduction table: 1
Number of models in juvenile maturity table: 1
General model parameter names (column 1), and
specific names used in these models (column 2):
            parameter_names mainparams
                time t
                            year2
                    individual individ
                            patch patch
                    alive in time t+1 surv3
4
>
> 6
>}
sizec3
9 reproductive status in time t+1 repst3
10 fecundity in time t+1 fec3
11
12
13
14
15
16
17
>
        reproductive status in time t repst2
        reproductive status in time t-1 repst1
        maturity status in time t+1 matst3
            maturity status in time t matst2
```

```
>22 age in time t age
> 23 density in time t density
> 24 individual covariate a in time t indcova2
> 25 individual covariate a in time t-1 indcova1
> 26 individual covariate b in time t indcovb2
> 27 individual covariate b in time t-1 indcovb1
> 28 individual covariate c in time t indcovc2
> 29 individual covariate c in time t-1 indcovc1
> 30 stage group in time t group2
>31 stage group in time t-1 group1
>
>
>
>
> Quality control:
>
> Survival model estimated with }74\mathrm{ individuals and 320 individual transitions.
> Survival model accuracy is 0.947.
> Observation status model estimated with 70 individuals and 303 individual transitions.
> Observation status model accuracy is 0.95.
> Primary size model estimated with 70 individuals and 288 individual transitions.
> Primary size model R-squared is 0.82.
> Secondary size model not estimated.
> Tertiary size model not estimated.
> Reproductive status model estimated with 70 individuals and 288 individual transitions.
> Reproductive status model accuracy is 0.74.
> Fecundity model estimated with 51 individuals and 118 individual transitions.
> Fecundity model R-squared is 0.535.
> Juvenile survival model not estimated.
> Juvenile observation status model not estimated.
> Juvenile primary size model not estimated.
> Juvenile secondary size model not estimated.
> Juvenile tertiary size model not estimated.
> Juvenile reproductive status model not estimated.
> Juvenile maturity status model not estimated.
```

In the summary output above, there is of course a section labeled Quality control, but there is more quality control in the output than just this section. Let's first explore some of the other parts of this output.

First, the best-fit model output is worth studying. The output is actually the output from whatever package and function was used to estimate the model. In this case, where the models were mixed models, the output comes from the packages 1 me 4 and glmmTMB. The most important quality control output comes in the form of the number of observations across the different random factors, and the overall variance or standard deviation of each random factor. If we have 118 observations in a best-fit model, and the random factors include a summed number of observations of around this number, then we likely cannot estimate random factors properly. For example, the conditional model of the fecundity model has 118 observations, and there are $5+3+51=59$ observations used up by random factors, so everything looks OK. While the models listed above look OK in general in this regard, the observation model does suggest some problems, since the standard deviations associated with individual and patch are equal to 0.0 .

Next, let's look over the section labeled Quality control. Here we see a good deal of information
that might be useful to us. First, we see the numbers of individuals and transitions (standardized dataset rows) used to develop each best-fit model. Generally, the higher the numbers of individuals and transitions, the better the overall quality of the model.

Second, the accuracy or $R^{2}$ of the best-fit model is shown. Accuracy is estimated as the number of predicted responses that are equal to the observed responses divided by the total number of responses in the dataset used to parameterize the model, and is applied to situations in which the response is binomial or a count. $R^{2}$ is a simple $R^{2}$ and is applied to all continuous response models. Accuracy works best with binomial models, because when applied to a count, accuracy does not distinguish models in which the predicted response is very wrong situations in which the predicted response is still very close. So, it may be worth exploring the predictions in count models a bit. However, regardless of this, we would argue that the best models have accuracy or $\mathrm{R}^{2}$ greater than or equal to 0.90 . Lower values make accurate prediction virtually impossible, and may impede inference.

Let's now look at quality control in MPMs themselves.

### 15.3 Quality control in MPMs and discretized IPMs

Let's first load some MPMs. Here, we will load some ahistorical Cypripedium MPMs from Chapter 4.

```
seeds_per_fruit <- 5000
sl_mult <- 0.7
cypsupp2_raw <- supplemental(stage3 = c("SD", "P1", "P2", "P3", "SL", "SL", "D",
    "XSm", "Sm", "SD", "P1"),
    stage2 = c("SD", "SD", "P1", "P2", "P3", "SL", "SL", "SL", "SL", "rep", "rep"),
    eststage3 = c(NA, NA, NA, NA, NA, NA, "D", "XSm", "Sm", NA, NA),
    eststage2 = c(NA, NA, NA, NA, NA, NA, "XSm", "XSm", "XSm", NA, NA),
    givenrate = c(0.08, 0.10, 0.10, 0.10, 0.05, 0.05, NA, NA, NA, NA, NA),
    multiplier = c(NA, NA, NA, NA, NA, NA, sl_mult, sl_mult, sl_mult,
        0.5 * seeds_per_fruit, 0.5 * seeds_per_fruit),
    type =c(1, 1, 1, 1, 1, 1, 1, 1, 1, 3, 3),
    stageframe = cypframe_raw, historical = FALSE)
cypmatrix2rp <- rlefko2(data = cypraw_v2, stageframe = cypframe_raw,
    year = "all", patch = "all", stages = c("stage3", "stage2"),
    size = c("size3added", "size2added"), supplement = cypsupp2_raw,
    yearcol = "year2", patchcol = "patchid", indivcol = "individ")
```

Now let's take a look at a summary of this lefkoMat object.

```
summary(cypmatrix2rp)
>
> This ahistorical lefkoMat object contains 15 matrices.
    Each matrix is square with }11\mathrm{ rows and columns, and a total of 121 elements.
    A total of 266 survival transitions were estimated, with 17.733 per matrix.
    A total of }70\mathrm{ fecundity transitions were estimated, with 4.667 per matrix.
    This lefkoMat object covers 1 population, 3 patches, and 5 time steps.
>
The dataset contains a total of 74 unique individuals and 320 unique transitions.
>
Survival probability sum check (each matrix represented by column in order):
```

| > | [,1] | [,2] | [,3] | [,4] | [,5] | [,6] | [,7] | [,8] | [,9] | [, 10] | [,11] | [,12] |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| > Min. | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 | 0.050 | 0.050 | 0.000 | 0.050 | 0.000 | 0.000 |
| > 1st Qu. | 0.075 | 0.025 | 0.075 | 0.025 | 0.075 | 0.075 | 0.140 | 0.140 | 0.100 | 0.140 | 0.100 | 0.100 |
| > Median | 0.180 | 0.100 | 0.180 | 0.100 | 0.180 | 0.180 | 0.909 | 0.778 | 0.686 | 0.857 | 0.750 | 0.575 |
| > Mean | 0.457 | 0.361 | 0.471 | 0.328 | 0.417 | 0.464 | 0.631 | 0.611 | 0.530 | 0.631 | 0.562 | 0.523 |
| > 3rd Qu. | 0.955 | 0.769 | 1.000 | 0.592 | 0.781 | 1.000 | 1.000 | 1.000 | 0.955 | 1.000 | 1.000 | 1.000 |
| > Max. | 1.000 | 1.000 | 1.000 | 1.000 | 1.000 | 1.000 | 1.000 | 1.000 | 1.000 | 1.000 | 1.000 | 1.000 |
| > | [,13] | [,14] | [,15] |  |  |  |  |  |  |  |  |  |
| > Min. | 0.000 | 0.000 | 0.000 |  |  |  |  |  |  |  |  |  |
| > 1st Qu. | 0.075 | 0.075 | 0.100 |  |  |  |  |  |  |  |  |  |
| > Median | 0.180 | 0.180 | 0.750 |  |  |  |  |  |  |  |  |  |
| > Mean | 0.432 | 0.450 | 0.562 |  |  |  |  |  |  |  |  |  |
| > 3rd Qu. | 0.875 | 1.000 | 1.000 |  |  |  |  |  |  |  |  |  |
| > Max. | 1.000 | 1.000 | 1.000 |  |  |  |  |  |  |  |  |  |

There are a few key portions of this output to look at, when assessing MPM quality. First, it is extremely useful to take a look at the number of individuals and transitions used to develop the MPM. The larger the numbers for both, the stronger the inference possible with an MPM. Here, we see that we have a small dataset, and so we need to bear that in mind when assessing our MPMs.

Next, let's look at the number of estimated transitions per matrix. We notice that this ahistorical MPM has $17.733+4.667=22.4$ estimated non-zero elements per matrix, but there are also 121 elements per matrix overall. So, our dataset is very sparse relative to our stageframe, the latter of which probably requires a larger dataset than we have access to.

Finally, the survival probability sum check gives us the quartile summary of column sums of the survival transition matrices in the MPM. This is very important, because the column sums give the survival probabilities of the stages in the stageframe (or the stage-pairs in a historical MPM, ages in a Leslie MPM, or age-stages in an age-by-stage MPM). So, the summaries should never show survival values greater than 1.0 or less than 0.0 . If they do, then there is an error in the MPM construction, and the user should most definitely go back to the drawing board (note that many of the matrices loaded into the COMPADRE and COMADRE matrices have this problem, and so will be flagged if imported into lefko3).

We plan to add further quality control protocols to package lefko3, and will update this manual as we do. Stay tuned!

### 15.4 Points to remember

1. Quality control in standardized datasets can be assessed with the summary_hfv() function
2. Errors in the development of life history models can be assessed when datasets are standardized via the functions verticalize3() and historicalize3().
3. Quality control in vital rate models can be assessed by looking at the numbers of observations utilized in the models and their random factors, as well by exploring the accuracy or $\mathrm{R}^{2}$ of each model.
4. MPM quality can be explored with the summary() function applied to a lefkoMat object of interest.

## Chapter 16

## Appendix I: Full Analysis Code Examples

This section includes full bkocs of code showing how to build and conduct basic analyses using lefko3 in a variety of situations.
16.0.1 Basic ahistorical stage-based (Lefkovitch), raw (empirical) model


Figure 16.1: Model 1. Basic stage-based model for Cypripedium candidum, a North American herbaceous plant species

This life history model includes 11 stages: 1) dormant seed 2) first-year protocorm 3) second-year protocorm 4) third-year protocorm 5) seedling 6) dormant adult (no sprouts) 7) extra-small adult (1 sprout) 8) small adult ( $2-3$ sprouts) 9) medium adult ( $4-5$ sprouts) 10) large adult ( $6-7$ sprouts) 11) extra-large adult ( $8+$ sprouts)

The stageframe for this life history model can be programmed as follows:

```
data(cypdata)
sizevector <- c(0, 0, 0, 0, 0, 0, 1, 3, 6, 11, 19.5)
stagevector <- c("SD", "P1", "P2", "P3", "SL", "D", "XSm", "Sm", "Md", "Lg",
    "XLg")
repvector <- c(0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1)
obsvector <- c(0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1)
matvector <- c(0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1)
immvector <- c(0, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0)
propvector <- c(1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
indataset <- c(0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1)
binvec <- c(0, 0, 0, 0, 0, 0.5, 0.5, 1.5, 1.5, 3.5, 5)
comments <- c("Dormant seed", "1st yr protocorm", "2nd yr protocorm",
    "3rd yr protocorm", "Seedling", "Dormant adult",
    "Extra small adult (1 shoot)", "Small adult (2-4 shoots)",
    "Medium adult (5-7 shoots)", "Large adult (8-14 shoots)",
    "Extra large adult (>14 shoots)")
cyp_lefk_raw_frame <- sf_create(sizes = sizevector, stagenames = stagevector,
    repstatus = repvector, obsstatus = obsvector, matstatus = matvector,
    propstatus = propvector, immstatus = immvector, indataset = indataset,
    binhalfwidth = binvec, comments = comments)
```

Now we will create the MPM, first developing the standardized dataset and the supplemental data table.

```
cyp_lefk_raw_data <- verticalize3(data = cypdata, noyears = 6, firstyear = 2004,
    patchidcol = "patch", individcol = "plantid", blocksize = 4,
    sizeacol = "Inf2.04", sizebcol = "Inf.04", sizeccol = "Veg.04",
    repstracol = "Inf.04", repstrbcol = "Inf2.04", fecacol = "Pod.04",
    stageassign = cyp_lefk_raw_frame, stagesize = "sizeadded", NAas0 = TRUE,
    NRasRep = TRUE, age_offset = 4)
seeds_per_fruit <- 5000
sl_mult <- 0.7
cyp_lefk_raw_supp <- supplemental(stage3 = c("SD", "P1", "P2", "P3", "SL", "SL",
        "D", "XSm", "Sm", "SD", "P1"),
    stage2 = c("SD", "SD", "P1", "P2", "P3", "SL", "SL", "SL", "SL", "rep","rep"),
    eststage3 = c(NA, NA, NA, NA, NA, NA, "D", "XSm", "Sm", NA, NA),
    eststage2 = c(NA, NA, NA, NA, NA, NA, "XSm", "XSm", "XSm", NA, NA),
    givenrate = c(0.08, 0.10, 0.10, 0.10, 0.05, 0.05, NA, NA, NA, NA, NA),
    multiplier = c(NA, NA, NA, NA, NA, NA, sl_mult, sl_mult, sl_mult,
        0.5 * seeds_per_fruit, 0.5 * seeds_per_fruit),
    type =c(1, 1, 1, 1, 1, 1, 1, 1, 1, 3, 3),
    stageframe = cyp_lefk_raw_frame, historical = FALSE)
cyp_lefk_raw_mpm <- rlefko2(data = cyp_lefk_raw_data, year = "all",
    patch = "all", stageframe = cyp_lefk_raw_frame, supplement = cyp_lefk_raw_supp,
```

```
    stages = c("stage3", "stage2"), size = c("size3added", "size2added"),
    yearcol = "year2", patchcol = "patchid", indivcol = "individ")
summary(cyp_lefk_raw_mpm)
>
> This ahistorical lefkoMat object contains 15 matrices.
>
> Each matrix is square with 11 rows and columns, and a total of 121 elements.
> A total of 266 survival transitions were estimated, with 17.733 per matrix.
> A total of 70 fecundity transitions were estimated, with 4.667 per matrix.
> This lefkoMat object covers 1 population, 3 patches, and 5 time steps.
>
> The dataset contains a total of 74 unique individuals and 320 unique transitions.
>
> Survival probability sum check (each matrix represented by column in order):
\(>\quad[, 1][, 2][, 3][, 4][, 5][, 6][, 7][, 8]\) [,9] [,10] [,11] [,12]
> Min. 0.000 0.000 0.000 0.000 0.000 0.000 0.050 0.050 0.000 0.050 0.000 0.000
> 1st Qu. 0.075 0.025 0.075 0.025 0.075 0.075 0.140 0.140 0.100 0.140 0.100 0.100
> Median 0.180 0.100 0.180 0.100 0.180 0.180 0.909 0.778 0.686 0.857 0.750 0.575
> Mean 0.457 0.361 0.471 0.328 0.417 0.464 0.631 0.611 0.530}00.631 0.562 0.523
> 3rd Qu. 0.955 0.769 1.000 0.592 0.781 1.000 1.000 1.000 0.955 1.000 1.000 1.000
> Max. 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000
> [,13] [,14] [,15]
> Min. 0.000 0.000 0.000
> 1st Qu. 0.075 0.075 0.100
> Median 0.180 0.180 0.750
> Mean 0.432 0.450 0.562
> 3rd Qu. 0.875 1.000 1.000
> Max. 1.000 1.000 1.000
```

Some basic analyses.

| lambda3 (cyp_lefk_raw_mpm) |  |  |  |  |
| :---: | :---: | :---: | :---: | :---: |
| > | pop | patch | year2 | lambda |
| > 1 | 1 | A | 2004 | 1.0830875 |
| 2 | 1 | A | 2005 | 1.0000000 |
| 3 | 1 | A | 2006 | 1.0000000 |
| 4 | 1 | A | 2007 | 0.5000000 |
| 5 | 1 | A | 2008 | 0.8917561 |
| 6 | 1 | B | 2004 | 1.0214506 |
| 7 | 1 | B | 2005 | 1.1285250 |
| 8 | 1 | B | 2006 | 1.1815829 |
| 9 | 1 | B | 2007 | 0.9604335 |
| > 10 | 1 | B | 2008 | 1.0893093 |
| > 11 | 1 | C | 2004 | 1.1124223 |
| > 12 | 1 | C | 2005 | 0.8333333 |
| $>13$ | 1 | C | 2006 | 1.1543634 |
| > 14 | 1 | C | 2007 | 0.8619361 |
| > 15 | 1 | C | 2008 | 1.0786352 |



Figure 16.2: Model 2. Complex stage-based model for Cypripedium candidum

### 16.0.2 Complex ahistorical stage-based (Lefkovitch), function-based model

This life history model includes 54 stages: 1) dormant seed 2) first-year protocorm (juvenile) 3) second-year protocorm (juvenile) 4) third-year protocorm (juvenile) 5) seedling (juvenile) 6) dormant adult 7-30) non-flowering but sprouting adults with 1-24 sprouts (1 stage per \# sprouts) 31-54) flowering adults with $1-24$ sprouts ( 1 stage per \# sprouts)

The stageframe for this life history model can be programmed as follows:

```
data(cypdata)
stagevector <- c("SD", "P1", "P2", "P3", "SL", "D", "V1", "V2", "V3", "V4",
    "V5", "V6", "V7", "V8", "V9", "V10", "V11", "V12", "V13", "V14", "V15", "V16",
    "V17", "V18", "V19", "V20", "V21", "V22", "V23", "V24", "F1", "F2", "F3",
    "F4", "F5", "F6", "F7", "F8", "F9", "F10", "F11", "F12", "F13", "F14", "F15",
    "F16", "F17", "F18", "F19", "F20", "F21", "F22", "F23", "F24")
indataset <- c(0, 0, 0, 0, 0, rep(1, 49))
sizevector <- c(0, 0, 0, 0, 0, seq(from = 0, t = 24), seq(from = 1, to = 24))
repvector <- c(0, 0, 0, 0, 0, rep(0, 25), rep(1, 24))
obsvector <- c(0, 0, 0, 0, 0, 0, rep(1, 48))
matvector <- c(0, 0, 0, 0, 0, rep(1, 49))
immvector <- c(0, 1, 1, 1, 1, rep(0, 49))
propvector <- c(1, rep(0, 53))
comments <- c("Dormant seed", "Yr1 protocorm", "Yr2 protocorm", "Yr3 protocorm",
    "Seedling", "Veg dorm", "Veg adult 1 stem", "Veg adult 2 stems",
    "Veg adult 3 stems", "Veg adult 4 stems", "Veg adult 5 stems",
    "Veg adult 6 stems", "Veg adult 7 stems", "Veg adult 8 stems",
```

```
"Veg adult 9 stems", "Veg adult 10 stems", "Veg adult 11 stems",
"Veg adult 12 stems", "Veg adult 13 stems", "Veg adult 14 stems",
"Veg adult 15 stems", "Veg adult 16 stems", "Veg adult 17 stems",
"Veg adult 18 stems", "Veg adult 19 stems", "Veg adult 20 stems",
"Veg adult 21 stems", "Veg adult 22 stems", "Veg adult 23 stems",
"Veg adult 24 stems", "Flo adult 1 stem", "Flo adult 2 stems",
"Flo adult 3 stems", "Flo adult 4 stems", "Flo adult 5 stems",
"Flo adult 6 stems", "Flo adult 7 stems", "Flo adult 8 stems",
"Flo adult 9 stems", "Flo adult 10 stems", "Flo adult 11 stems",
"Flo adult 12 stems", "Flo adult 13 stems", "Flo adult 14 stems",
"Flo adult 15 stems", "Flo adult 16 stems", "Flo adult 17 stems",
"Flo adult }18\mathrm{ stems", "Flo adult 19 stems", "Flo adult 20 stems",
"Flo adult }21\mathrm{ stems", "Flo adult 22 stems", "Flo adult 23 stems",
"Flo adult 24 stems")
cyp_lefk_fb_frame <- sf_create(sizes = sizevector, stagenames = stagevector,
    repstatus = repvector, obsstatus = obsvector, matstatus = matvector ,
    propstatus = propvector, immstatus = immvector, indataset = indataset,
    comments = comments)
```

Now we will create the MPM, first developing the standardized dataset, the supplemental data table, and the vital rate models.

```
cyp_lefk_fb_data <- verticalize3(data = cypdata, noyears = 6, firstyear = 2004,
    patchidcol = "patch", individcol = "plantid", blocksize = 4,
    sizeacol = "Inf2.04", sizebcol = "Inf.04", sizeccol = "Veg.04",
    repstracol = "Inf.04", repstrbcol = "Inf2.04", fecacol = "Pod.04",
    stageassign = cyp_lefk_fb_frame, stagesize = "sizeadded", NAas0 = TRUE,
    age_offset = 4)
seeds_per_fruit <- 5000
sl_mult <- 0.7
cyp_lefk_fb_supp <- supplemental(stage3 = c("SD", "P1", "P2", "P3", "SL", "SL",
        "D", "V1", "V2", "V3", "SD", "P1"),
    stage2 = c("SD", "SD", "P1", "P2", "P3", "SL", "SL", "SL", "SL", "SL", "rep",
        "rep"),
    eststage3 = c(NA, NA, NA, NA, NA, NA, "D", "V1", "V2", "V3", NA, NA),
    eststage2 = c(NA, NA, NA, NA, NA, NA, "D", "D", "D", "D", NA, NA),
    givenrate = c(0.08, 0.1, 0.1, 0.1, 0.05, 0.05, NA, NA, NA, NA, NA, NA),
    multiplier = c(NA, NA, NA, NA, NA, NA, sl_mult, sl_mult, sl_mult, sl_mult,
        0.5 * seeds_per_fruit, 0.5 * seeds_per_fruit),
    type = c("S", "S", "S", "S", "S", "S", "S", "S", "S", "S", "R", "R"),
    stageframe = cyp_lefk_fb_frame, historical = FALSE)
cyp_lefk_fb_vital <- modelsearch(cyp_lefk_fb_data, historical = FALSE,
    suite = "main", approach = "mixed",
    vitalrates = c("surv", "obs", "size", "repst", "fec"), patch = "patchid",
    sizedist = "negbin", size.trunc = TRUE, fecdist = "poisson", fec.zero = TRUE,
    size = c("size3added", "size2added", "size1added"), quiet = "partial")
>
> Developing global model of survival probability...
```

```
>
Global model of survival probability developed. Proceeding with model dredge...
>
Developing global model of observation probability...
Global model of observation probability developed. Proceeding with model dredge...
Developing global model of primary size...
Global model of primary size developed. Proceeding with model dredge...
Developing global model of reproduction probability...
Global model of reproduction probability developed. Proceeding with model dredge...
Developing global model of fecundity...
Global model of fecundity developed. Proceeding with model dredge...
Finished selecting best-fit models.
cyp_lefk_fb_mpm <- flefko2(stageframe = cyp_lefk_fb_frame,
    supplement = cyp_lefk_fb_supp, modelsuite = cyp_lefk_fb_vital,
    data = cyp_lefk_fb_data)
summary(cyp_lefk_fb_mpm)
>
> This ahistorical lefkoMat object contains }15\mathrm{ matrices.
Each matrix is square with }54\mathrm{ rows and columns, and a total of 2916 elements.
A total of }36165\mathrm{ survival transitions were estimated, with }2411\mathrm{ per matrix.
A total of }720\mathrm{ fecundity transitions were estimated, with 48 per matrix.
This lefkoMat object covers 1 population, 3 patches, and 5 time steps.
The dataset contains a total of }74\mathrm{ unique individuals and 320 unique transitions.
Vital rate modeling quality control:
>
Survival estimated with }74\mathrm{ individuals and 320 individual transitions.
Observation estimated with }70\mathrm{ individuals and 303 individual transitions.
Primary size estimated with }70\mathrm{ individuals and 288 individual transitions.
Secondary size transition not estimated.
Tertiary size transition not estimated.
Reproductive status estimated with }70\mathrm{ individuals and 288 individual transitions.
Fecundity estimated with }51\mathrm{ individuals and }118\mathrm{ individual transitions.
Juvenile survival not estimated.
Juvenile observation probability not estimated.
Juvenile primary size transition not estimated.
Juvenile secondary size transition not estimated.
Juvenile tertiary size transition not estimated.
Juvenile reproduction probability not estimated.
```

```
> Juvenile maturity transition probability not estimated.
>
> Survival probability sum check (each matrix represented by column in order):
> [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11] [,12]
> Min. 0.050 0.050 0.050 0.050 0.050 0.050 0.050 0.050 0.050 0.050 0.050 0.050
> 1st Qu. 0.991 0.991 0.991 0.991 0.991 0.991 0.991 0.991 0.991 0.991 0.991 0.991
> Median 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000
> Mean 0.916 0.917 0.917 0.917 0.914 0.916 0.917 0.917 0.917 0.914 0.914 0.916
> 3rd Qu. 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000
> Max. 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000
> [,13] [,14] [,15]
> Min. 0.050 0.050 0.050
> 1st Qu. 0.991 0.991 0.991
> Median 1.000 1.000 1.000
> Mean 0.916 0.916 0.913
> 3rd Qu. 1.000 1.000 1.000
> Max. 1.000 1.000 1.000
```

Now some basic analyses.

| lambda3 (cyp_lefk_fb_mpm) |  |  |  |  |
| :--- | ---: | ---: | ---: | :--- |
| $>$ | pop | patch | year2 | lambda |
| $>1$ | 1 | A | 2004 | 1.061031 |
| $>$ | 2 | 1 | A | 2005 |
| $\gg 3$ | 1 | A | 2006 | 1.025950 |
| $>$ | 4 | 1 | A | 2007 |
| $>$ | 1.0509811 |  |  |  |
| $>5$ | 1 | A | 2008 | 1.071040 |
| $>6$ | 1 | B | 2004 | 1.045524 |
| $>7$ | 1 | B | 2005 | 1.010656 |
| $>8$ | 1 | B | 2006 | 1.023959 |
| $>9$ | 1 | B | 2007 | 1.035926 |
| $>10$ | 1 | B | 2008 | 1.061884 |
| $>11$ | 1 | C | 2004 | 1.055358 |
| $>12$ | 1 | C | 2005 | 1.032827 |
| $>13$ | 1 | C | 2006 | 1.044500 |
| $>14$ | 1 | C | 2007 | 1.050822 |
| $>15$ | 1 | C | 2008 | 1.056175 |

### 16.0.3 Basic historical stage-based (Lefkovitch), raw (empirical) model

This life history model includes 11 stages: 1) dormant seed 2) first-year protocorm 3) second-year protocorm 4) third-year protocorm 5) seedling 6) dormant adult (no sprouts) 7) extra-small adult (1 sprout) 8) small adult ( $2-3$ sprouts) 9) medium adult ( $4-5$ sprouts) 10) large adult ( $6-7$ sprouts) 11) extra-large adult ( $8+$ sprouts)

The stageframe for this life history model can be programmed as follows:

```
data(cypdata)
sizevector <- c(0, 0, 0, 0, 0, 0, 1, 3, 6, 11, 19.5)
stagevector <- c("SD", "P1", "P2", "P3", "SL", "D", "XSm", "Sm", "Md", "Lg",
    "XLg")
```



Figure 16.3: Model 1. Basic stage-based model for Cypripedium candidum, a North American herbaceous plant species

```
repvector <- c(0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1)
obsvector <- c(0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1)
matvector <- c(0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1)
immvector <- c(0, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0)
propvector <- c(1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
indataset <- c(0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1)
binvec <- c(0, 0, 0, 0, 0, 0.5, 0.5, 1.5, 1.5, 3.5, 5)
comments <- c("Dormant seed", "1st yr protocorm", "2nd yr protocorm",
    "3rd yr protocorm", "Seedling", "Dormant adult",
    "Extra small adult (1 shoot)", "Small adult (2-4 shoots)",
    "Medium adult (5-7 shoots)", "Large adult (8-14 shoots)",
    "Extra large adult (>14 shoots)")
cyp_lefk_raw_frame <- sf_create(sizes = sizevector, stagenames = stagevector,
    repstatus = repvector, obsstatus = obsvector, matstatus = matvector,
    propstatus = propvector, immstatus = immvector, indataset = indataset,
    binhalfwidth = binvec, comments = comments)
```

Now we will create the MPM, first developing the standardized dataset and the supplemental data table.

```
cyp_lefk_raw_data <- verticalize3(data = cypdata, noyears = 6, firstyear = 2004,
    patchidcol = "patch", individcol = "plantid", blocksize = 4,
    sizeacol = "Inf2.04", sizebcol = "Inf.04", sizeccol = "Veg.04",
    repstracol = "Inf.04", repstrbcol = "Inf2.04", fecacol = "Pod.04",
    stageassign = cyp_lefk_raw_frame, stagesize = "sizeadded", NAasO = TRUE,
```

```
    NRasRep = TRUE, age_offset = 4)
seeds_per_fruit <- 5000
sl_mult <- 0.7
cyp_lefk_raw_hsupp <- supplemental(stage3 = c("SD", "SD", "P1", "P1", "P2", "P2",
        "P3", "SL", "SL", "SL", "D", "XSm", "Sm", "D", "XSm", "Sm", "mat", "mat",
        "mat", "SD", "P1"),
    stage2 = c("SD", "SD", "SD", "SD", "P1", "P1", "P2", "P3", "SL", "SL", "SL",
        "SL", "SL", "SL", "SL", "SL", "D", "XSm", "Sm", "rep", "rep"),
    stage1 = c("SD", "rep", "SD", "rep", "SD", "rep", "P1", "P2", "P3", "SL",
        "SL", "SL", "SL", "P3", "P3", "P3", "SL", "SL", "SL", "mat", "mat"),
    eststage3 = c(NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, "D", "XSm", "Sm", "D",
        "XSm", "Sm", "mat", "mat", "mat", NA, NA),
    eststage2 = c(NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, "XSm", "XSm", "XSm",
        "XSm", "XSm", "XSm", "D", "XSm", "Sm", NA, NA),
    eststage1 = c(NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, "XSm", "XSm", "XSm",
        "XSm", "XSm", "XSm", "XSm", "XSm", "XSm", NA, NA),
    givenrate = c(0.08, 0.10, 0.10, 0.10, 0.10, 0.10, 0.10, 0.05, 0.05, 0.05, NA,
        NA, NA, NA, NA, NA, NA, NA, NA, NA, NA),
    multiplier = c(NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, sl_mult, sl_mult,
        sl_mult, sl_mult, sl_mult, sl_mult, 1, 1, 1, 0.5 * seeds_per_fruit,
        0.5 * seeds_per_fruit),
    type =c(1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 3, 3),
    stageframe = cyp_lefk_raw_frame, historical = TRUE)
cyp_lefk_raw_hmpm <- rlefko3(data = cyp_lefk_raw_data,
    stageframe = cyp_lefk_raw_frame, supplement = cyp_lefk_raw_hsupp,
    year = "all", patch = "all", stages = c("stage3", "stage2", "stage1"),
    size = c("size3added", "size2added", "size1added"),
    yearcol = "year2", patchcol = "patchid", indivcol = "individ")
summary(cyp_lefk_raw_hmpm)
>
> This historical lefkoMat object contains 12 matrices.
> Each matrix is square with }121\mathrm{ rows and columns, and a total of 14641 elements.
> A total of }516\mathrm{ survival transitions were estimated, with 43 per matrix.
> A total of }70\mathrm{ fecundity transitions were estimated, with }5.833\mathrm{ per matrix.
> This lefkoMat object covers 1 population, 3 patches, and 4 time steps.
>
> The dataset contains a total of 74 unique individuals and 320 unique transitions.
>
> Survival probability sum check (each matrix represented by column in order):
> [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11]
> Min. 0.000 0.0000 0.0000 0.000 0.000 0.000 0.00 0.000 0.000 0.0000 0.000
> 1st Qu. 0.000 0.0000 0.0000 0.000 0.000 0.000 0.00 0.000 0.000 0.0000 0.000
>Median 0.000 0.0000 0.0000 0.000 0.000 0.000 0.00 0.000 0.000 0.0000 0.000
l Mean 0. 0.107 0.0945 0.0851 0.101 0.158 0.158 0.14 0.169 0.119 0.0851 0.119
> 3rd Qu. 0.000 0.0000 0.0000 0.000 0.100 0.100 0.05 0.100 0.000 0.0000 0.000
> Max. 1.000 1.0000 1.0000 1.000 1.000 1.000 1.00 1.000 1.000 1.0000 1.000
```

```
> [,12]
>Min. 0.000
> 1st Qu. 0.000
> Median 0.000
> Mean 0.144
> 3rd Qu. 0.000
> Max. 1.000
```

Some basic analyses.

| lambda3(cyp_lefk_raw_hmpm) |  |  |  |  |
| :---: | :---: | :---: | :---: | :---: |
| > | pop | patch | year2 | lambda |
| > 1 | 1 | A | 2005 | 1.0 |
| $>2$ | 1 | A | 2006 | 1.0 |
| $>3$ | 1 | A | 2007 | 0.5 |
| $>4$ | 1 | A | 2008 | 1.0 |
| > 5 | 1 | B | 2005 | 1.0 |
| $>6$ | 1 | B | 2006 | 1.0 |
| > 7 | 1 | B | 2007 | 1.0 |
| $>8$ | 1 | B | 2008 | 1.0 |
| > 9 | 1 | C | 2005 | 1.0 |
| > 10 | 1 | C | 2006 | 1.0 |
| > 11 | 1 | C | 2007 | 1.0 |
| > 12 | 1 | C | 2008 | 1.0 |

### 16.0.4 Complex historical stage-based (Lefkovitch), function-based model



Figure 16.4: Model 2. Complex stage-based model for Cypripedium candidum

This life history model includes 54 stages: 1) dormant seed 2) first-year protocorm (juvenile) 3) second-year protocorm (juvenile) 4) third-year protocorm (juvenile) 5) seedling (juvenile) 6) dormant adult 7-30) non-flowering but sprouting adults with $1-24$ sprouts (1 stage per \# sprouts) 31-54) flowering adults with $1-24$ sprouts ( 1 stage per \# sprouts)

The stageframe for this life history model can be programmed as follows:

```
data(cypdata)
```

stagevector <- c("SD", "P1", "P2", "P3", "SL", "D", "V1", "V2", "V3", "V4",
"V5", "V6", "V7", "V8", "V9", "V10", "V11", "V12", "V13", "V14", "V15", "V16",
"V17", "V18", "V19", "V20", "V21", "V22", "V23", "V24", "F1", "F2", "F3",
"F4", "F5", "F6", "F7", "F8", "F9", "F10", "F11", "F12", "F13", "F14", "F15",
"F16", "F17", "F18", "F19", "F20", "F21", "F22", "F23", "F24")
indataset <- c(0, 0, 0, 0, 0, rep(1, 49))
sizevector $<-c(0,0,0,0,0, \operatorname{seq}(f r o m=0, t=24)$, seq (from = 1 , to $=24)$ )
repvector $<-c(0,0,0,0,0, \operatorname{rep}(0,25), \operatorname{rep}(1,24))$
obsvector $<-c(0,0,0,0,0,0, r e p(1,48))$
matvector $<-c(0,0,0,0,0, r e p(1,49))$
immvector <- c(0, 1, 1, 1, 1, rep $(0,49))$
propvector <- c(1, rep $(0,53))$
comments <- c("Dormant seed", "Yr1 protocorm", "Yr2 protocorm", "Yr3 protocorm",
"Seedling", "Veg dorm", "Veg adult 1 stem", "Veg adult 2 stems",
"Veg adult 3 stems", "Veg adult 4 stems", "Veg adult 5 stems",
"Veg adult 6 stems", "Veg adult 7 stems", "Veg adult 8 stems",
"Veg adult 9 stems", "Veg adult 10 stems", "Veg adult 11 stems",
"Veg adult 12 stems", "Veg adult 13 stems", "Veg adult 14 stems",
"Veg adult 15 stems", "Veg adult 16 stems", "Veg adult 17 stems",
"Veg adult 18 stems", "Veg adult 19 stems", "Veg adult 20 stems",
"Veg adult 21 stems", "Veg adult 22 stems", "Veg adult 23 stems",
"Veg adult 24 stems", "Flo adult 1 stem", "Flo adult 2 stems",
"Flo adult 3 stems", "Flo adult 4 stems", "Flo adult 5 stems",
"Flo adult 6 stems", "Flo adult 7 stems", "Flo adult 8 stems",
"Flo adult 9 stems", "Flo adult 10 stems", "Flo adult 11 stems",
"Flo adult 12 stems", "Flo adult 13 stems", "Flo adult 14 stems",
"Flo adult 15 stems", "Flo adult 16 stems", "Flo adult 17 stems",
"Flo adult 18 stems", "Flo adult 19 stems", "Flo adult 20 stems",
"Flo adult 21 stems", "Flo adult 22 stems", "Flo adult 23 stems",
"Flo adult 24 stems")
cyp_lefk_fb_frame <- sf_create(sizes = sizevector, stagenames = stagevector,
repstatus = repvector, obsstatus = obsvector, matstatus = matvector,
propstatus = propvector, immstatus = immvector, indataset = indataset,
comments $=$ comments)

Now we will create the MPM, first developing the standardized dataset, the supplemental data table, and the vital rate models.

```
cyp_lefk_fb_data <- verticalize3(data = cypdata, noyears = 6, firstyear = 2004,
    patchidcol = "patch", individcol = "plantid", blocksize = 4,
    sizeacol = "Inf2.04", sizebcol = "Inf.04", sizeccol = "Veg.04",
    repstracol = "Inf.04", repstrbcol = "Inf2.04", fecacol = "Pod.04",
```

```
    stageassign = cyp_lefk_fb_frame, stagesize = "sizeadded", NAas0 = TRUE,
    age_offset = 4)
seeds_per_fruit <- 5000
sl_mult <- 0.7
cyp_lefk_fb_hsupp <- supplemental(stage3 = c("SD", "SD", "P1", "P1", "P2", "P3", "SL",
        "SL", "SL", "D", "V1", "V2", "V3", "D", "V1", "V2", "V3", "mat", "mat",
        "mat", "mat", "SD", "P1"),
    stage2 = c("SD", "SD", "SD", "SD", "P1", "P2", "P3", "SL", "SL", "SL", "SL",
        "SL", "SL", "SL", "SL", "SL", "SL", "D", "V1", "V2", "V3", "rep", "rep"),
    stage1 = c("SD", "rep", "SD", "rep", "SD", "P1", "P2", "P3", "SL", "P3", "P3",
        "P3", "P3", "SL", "SL", "SL", "SL", "SL", "SL", "SL", "SL", "mat", "mat"),
    eststage3 = c(NA, NA, NA, NA, NA, NA, NA, NA, NA, "D", "V1", "V2", "V3", "D",
        "V1", "V2", "V3", "mat", "mat", "mat", "mat", NA, NA),
    eststage2 = c(NA, NA, NA, NA, NA, NA, NA, NA, NA, "D", "D", "D", "D", "D",
        "D", "D", "D", "D", "V1", "V2", "V3", NA, NA),
    eststage1 = c(NA, NA, NA, NA, NA, NA, NA, NA, NA, "D", "D", "D", "D", "D",
        "D", "D", "D", "V1", "V1", "V1", "V1", NA, NA),
    givenrate =c(0.08, 0.08, 0.1, 0.1, 0.1, 0.1, 0.1, 0.05, 0.05, NA, NA, NA, NA,
        NA, NA, NA, NA, NA, NA, NA, NA, NA, NA),
    multiplier = c(NA, NA, NA, NA, NA, NA, NA, NA, NA, sl_mult, sl_mult,
        sl_mult, sl_mult, sl_mult, sl_mult, sl_mult, sl_mult, 1, 1, 1, 1,
        0.5 * seeds_per_fruit, 0.5 * seeds_per_fruit),
    type = c("S", "S", "S", "S", "S", "S", "S", "S", "S", "S", "S", "S", "S", "S",
        "S", "S", "S", "S", "S", "S", "S", "R", "R"),
    type_t12 = c("S", "F", "S", "F", "S", "S", "S", "S", "S", "S", "S", "S", "S",
        "S", "S", "S", "S", "S", "S", "S", "S", "S", "S"),
    stageframe = cyp_lefk_fb_frame, historical = TRUE)
cyp_lefk_fb_vital <- modelsearch(cyp_lefk_fb_data, historical = TRUE, approach = "mixed",
    vitalrates = c("surv", "obs", "size", "repst", "fec"), patch = "patchid",
    sizedist = "negbin", size.trunc = TRUE, fecdist = "poisson", fec.zero = TRUE,
    suite = "main", size = c("size3added", "size2added", "size1added"),
    quiet = "partial")
    Developing global model of survival probability...
>
    Global model of survival probability developed. Proceeding with model dredge...
    Developing global model of observation probability...
    Global model of observation probability developed. Proceeding with model dredge...
    Developing global model of primary size...
    Global model of primary size developed. Proceeding with model dredge...
    Developing global model of reproduction probability...
    Global model of reproduction probability developed. Proceeding with model dredge...
```

```
>
> Developing global model of fecundity...
>
> Global model of fecundity developed. Proceeding with model dredge...
>
> Finished selecting best-fit models.
cyp_lefk_fb_hmpm <- flefko3(stageframe = cyp_lefk_fb_frame, supplement = cyp_lefk_fb_hsupp,
    modelsuite = cyp_lefk_fb_vital, data = cyp_lefk_fb_data, sparse_output = TRUE)
summary(cyp_lefk_fb_hmpm)
>
> This historical lefkoMat object contains 15 matrices.
>
> Each matrix is square with 2916 rows and columns, and a total of 8503056 elements.
> A total of 1768620 survival transitions were estimated, with 117908 per matrix.
> A total of 35280 fecundity transitions were estimated, with }2352\mathrm{ per matrix.
> This lefkoMat object covers 1 population, 3 patches, and 5 time steps.
>
> The dataset contains a total of 74 unique individuals and 320 unique transitions.
>
> Vital rate modeling quality control:
>
> Survival estimated with 74 individuals and 320 individual transitions.
> Observation estimated with 70 individuals and 303 individual transitions.
> Primary size estimated with 70 individuals and 288 individual transitions.
> Secondary size transition not estimated.
> Tertiary size transition not estimated.
> Reproductive status estimated with 70 individuals and 288 individual transitions.
Fecundity estimated with 51 individuals and 118 individual transitions.
Juvenile survival not estimated.
Juvenile observation probability not estimated.
> Juvenile primary size transition not estimated.
> Juvenile secondary size transition not estimated.
> Juvenile tertiary size transition not estimated.
Juvenile reproduction probability not estimated.
Juvenile maturity transition probability not estimated.
>
> Survival probability sum check (each matrix represented by column in order):
> [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11] [,12]
Min. 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000
1st Qu. 0.965 0.965 0.965 0.965 0.965 0.964 0.965 0.965 0.965 0.964 0.964 0.965
Median 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000
Mean 0.820 0.820 0.820 0.820 0.820 0.820 0.820 0.820 0.820 0.820 0.820 0.820
3rd Qu. 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000
Max. 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000
[,13] [,14] [,15]
Min. 0.000 0.000 0.000
1st Qu. 0.965 0.964 0.964
Median 1.000 1.000 1.000
Mean 0.820 0.820 0.820
```

```
> 3rd Qu. 1.000 1.000 1.000
> Max. 1.000 1.000 1.000
```

Now some basic analyses.

| lambda3 (cyp_lefk_fb_hmpm) |  |  |  |  |
| :---: | :---: | :---: | :---: | :---: |
| > | pop | patch | year2 | lambda |
| > 1 | , | A | 2004 | 0.9897792 |
| $>2$ | 1 | A | 2005 | 0.9745008 |
| $>3$ | 1 | A | 2006 | 0.9800239 |
| $>4$ | 1 | A | 2007 | 0.9801853 |
| $>5$ | 1 | A | 2008 | 0.9937485 |
| $>6$ | 1 | B | 2004 | 0.9863332 |
| $>7$ | 1 | B | 2005 | 0.9719965 |
| $>8$ | 1 | B | 2006 | 0.9766326 |
| > 9 | 1 | B | 2007 | 0.9768889 |
| > 10 | 1 | B | 2008 | 0.9923980 |
| > 11 | 1 | C | 2004 | 0.9908893 |
| > 12 | 1 | C | 2005 | 0.9802220 |
| > 13 | 1 | C | 2006 | 0.9847044 |
| > 14 | 1 | C | 2007 | 0.9834079 |
| > 15 | 1 | C | 2008 | 0.9914346 |

### 16.0.5 Age-based (Leslie) raw (empirical) model

This life history model includes 7 main ages, assuming the possibility of survival past age 7. This model does not include a dormant seed stage, instead assuming that all seeds either germinate immediately or die.

This life history model does not require a stageframe to be programmed. Instead, we will standardize the dataset, develop a supplemental data table, and create the MPM.

```
data(cypdata)
cyp_lesl_data <- verticalize3(data = cypdata, noyears = 6, firstyear = 2004,
    patchidcol = "patch", individcol = "plantid", blocksize = 4,
    sizeacol = "Inf2.04", sizebcol = "Inf.04", sizeccol = "Veg.04",
    repstracol = "Inf.04", repstrbcol = "Inf2.04", fecacol = "Pod.04",
    stagesize = "sizeadded", NAas0 = TRUE, age_offset = 2)
germination <- 0.08
protocorm_to_seedling <- 0.10
seeding_to_adult <- 0.20
seeds_per_fruit <- 8000
cyp_lesl_supp <- supplemental(historical = FALSE, stagebased = FALSE,
    agebased = TRUE, age2 = c(1, 2), type = c(1, 1),
    givenrate = c(protocorm_to_seedling, seeding_to_adult))
cyp_lesl_raw_mpm <- rleslie(data = cyp_lesl_data, last_age = 7, fecage_min = 3,
    fecmod = (germination * seeds_per_fruit), supplement = cyp_lesl_supp)
```



Figure 16.5: Age-based model for Cypripedium candidum

Now some basic analyses.

```
lambda3(cyp_lesl_raw_mpm)
> Warning: A matrix with an eigenvalue of 0 has been detected.
> Warning: A matrix with an eigenvalue of 0 has been detected.
> pop patch year2 lambda
>11 1 1 1 2004 1.367981
> 2 1 1 1 2005 1.602825
> 3 1 1 1 2006 0.000000
> 4 1 1 1 2007 0.000000
> 5 1 1 1 2008 0.962963
cyp_lesl_raw_meanmpm <- lmean(cyp_lesl_raw_mpm)
lambda3(cyp_lesl_raw_meanmpm)
> pop patch lambda
> 1 1 1 1.42624
```


### 16.0.6 Age-based (Leslie) function-based model

This life history model includes 7 main ages, assuming the possibility of survival past age 7. This model does not include a dormant seed stage, instead assuming that all seeds either germinate imme-


Figure 16.6: Age-based model for Cypripedium candidum
diately or die.
This life history model does not require a stageframe to be programmed. Instead, we will standardize the dataset, develop a supplemental data table, and create the MPM.

```
data(cypdata)
cyp_lesl_data <- verticalize3(data = cypdata, noyears = 6, firstyear = 2004,
    patchidcol = "patch", individcol = "plantid", blocksize = 4,
    sizeacol = "Inf2.04", sizebcol = "Inf.04", sizeccol = "Veg.04",
    repstracol = "Inf.04", repstrbcol = "Inf2.04", fecacol = "Pod.04",
    stagesize = "sizeadded", NAas0 = TRUE, age_offset = 2)
cyp_lesl_vital <- modelsearch(cyp_lesl_data, historical = FALSE,
    approach = "mixed", suite = "cons", bestfit = "AICc&k", age = "obsage",
    vitalrates = c("surv", "fec"), fecdist = "poisson", indiv = "individ",
    year = "year2", year.as.random = TRUE, patch.as.random = TRUE,
    show.model.tables = TRUE, fec.zero = TRUE, global.only = TRUE,
    test.age = TRUE, quiet = "partial")
>
> Developing global model of survival probability...
>
> Global model of survival probability developed. Proceeding with model dredge...
>
> Observation probability will be treated as constant.
```

```
>
> Primary size will be treated as constant.
>
> Reproduction probability will be treated as constant.
>
> Developing global model of fecundity...
>
Global model of fecundity developed. Proceeding with model dredge...
germination <- 0.08
protocorm_to_seedling <- 0.10
seeding_to_adult <- 0.20
seeds_per_fruit <- 8000
cyp_lesl_supp <- supplemental(historical = FALSE, stagebased = FALSE,
    agebased = TRUE, age2 = c(1, 2), type = c(1, 1),
    givenrate = c(protocorm_to_seedling, seeding_to_adult))
```

Now we will create the MPM.

```
cyp_lesl_fb_mpm <- fleslie(data = cyp_lesl_data, modelsuite = cyp_lesl_vital,
    last_age = 7, fecage_min = 3, fecmod = (germination * seeds_per_fruit),
    supplement = cyp_lesl_supp)
summary(cyp_lesl_fb_mpm)
>
> This ahistorical lefkoMat object contains 5 matrices.
>
> Each matrix is square with 7 rows and columns, and a total of 49 elements.
> A total of 35 survival transitions were estimated, with 7 per matrix.
> A total of 25 fecundity transitions were estimated, with 5 per matrix.
> This lefkoMat object covers 1 population, 1 patch, and 5 time steps.
>
> The dataset contains a total of 74 unique individuals and 320 unique transitions.
>
> Vital rate modeling quality control:
>
> Survival estimated with 74 individuals and 320 individual transitions.
> Observation probability not estimated.
> Primary size transition not estimated.
> Secondary size transition not estimated.
> Tertiary size transition not estimated.
Reproduction probability not estimated.
Fecundity estimated with 51 individuals and 118 individual transitions.
Juvenile survival not estimated.
Juvenile observation probability not estimated.
Juvenile primary size transition not estimated.
Juvenile secondary size transition not estimated.
Juvenile tertiary size transition not estimated.
Juvenile reproduction probability not estimated.
Juvenile maturity transition probability not estimated.
```

```
> Survival probability sum check (each matrix represented by column in order):
> [,1] [,2] [,3] [,4] [,5]
> Min. 0.100 0.100 0.100 0.100 0.100
> 1st Qu. 0.573 0.573 0.573 0.573 0.573
> Median 0.946 0.946 0.946 0.946 0.946
> Mean 0.719 0.719 0.719 0.719 0.719
> 3rd Qu. 0.947 0.947 0.947 0.947 0.947
> Max. 0.949 0.949 0.949 0.949 0.949
```

Finally some analyses.

```
lambda3(cyp_lesl_fb_mpm)
> pop patch year2 lambda
> 1 1 1 1 2004 1.250918
> 2 1 1 1 2005 1.438559
> 3 1 1 1 2006 1.174798
> 4 1 1 1 2007 1.149201
> 5 1 1 1 2008 1.163000
```


### 16.0.7 Hybrid age-based (Leslie) model with dormant propagule



Figure 16.7: Hybrid age-based model for Cypripedium candidum including a dormant seed stage

This life history model includes 7 ages after germination, including 2 juvenile stages (protocorm and seedling) and 5 adult ages, plus a dormant seed stage.

We will use a Leslie model as the backbone for this. Leslie MPMs do not require stageframes to be built, but include auto-generated stageframes within the associated lefkoMat object that is produced. The first part of this exercise is the same as in the previous Leslie exercise.

```
data(cypdata)
cyp_lesl_data <- verticalize3(data = cypdata, noyears = 6, firstyear = 2004,
    patchidcol = "patch", individcol = "plantid", blocksize = 4,
```

```
    sizeacol = "Inf2.04", sizebcol = "Inf.04", sizeccol = "Veg.04",
    repstracol = "Inf.04", repstrbcol = "Inf2.04", fecacol = "Pod.04",
    stagesize = "sizeadded", NAas0 = TRUE, age_offset = 2)
cyp_lesl_vital <- modelsearch(cyp_lesl_data, historical = FALSE,
    approach = "mixed", suite = "cons", bestfit = "AICc&k", age = "obsage",
    vitalrates = c("surv", "fec"), fecdist = "poisson", indiv = "individ",
    year = "year2", year.as.random = TRUE, patch.as.random = TRUE,
    show.model.tables = TRUE, fec.zero = TRUE, global.only = TRUE,
    test.age = TRUE, quiet = "partial")
>
> Developing global model of survival probability...
>
> Global model of survival probability developed. Proceeding with model dredge...
>
Observation probability will be treated as constant.
Primary size will be treated as constant.
Reproduction probability will be treated as constant.
Developing global model of fecundity...
Global model of fecundity developed. Proceeding with model dredge...
germination <- 0.08
protocorm_to_seedling <- 0.10
seeding_to_adult <- 0.20
seeds_per_fruit <- 8000
cyp_lesl_supp <- supplemental(historical = FALSE, stagebased = FALSE,
    agebased = TRUE, age2 = c(1, 2), type = c(1, 1),
    givenrate = c(protocorm_to_seedling, seeding_to_adult))
```

Now we will create the Leslie MPM.

```
cyp_lesl_fb_mpm <- fleslie(data = cyp_lesl_data, modelsuite = cyp_lesl_vital,
    last_age = 7, fecage_min = 3, fecmod = (germination * seeds_per_fruit),
    supplement = cyp_lesl_supp)
summary(cyp_lesl_fb_mpm)
>
This ahistorical lefkoMat object contains 5 matrices.
Each matrix is square with 7 rows and columns, and a total of 49 elements.
A total of 35 survival transitions were estimated, with 7 per matrix.
A total of 25 fecundity transitions were estimated, with 5 per matrix.
This lefkoMat object covers 1 population, 1 patch, and 5 time steps.
The dataset contains a total of 74 unique individuals and 320 unique transitions.
Vital rate modeling quality control:
```

```
>
> Survival estimated with 74 individuals and 320 individual transitions.
> Observation probability not estimated.
> Primary size transition not estimated.
> Secondary size transition not estimated.
> Tertiary size transition not estimated.
> Reproduction probability not estimated.
> Fecundity estimated with 51 individuals and 118 individual transitions.
> Juvenile survival not estimated.
> Juvenile observation probability not estimated.
> Juvenile primary size transition not estimated.
> Juvenile secondary size transition not estimated.
> Juvenile tertiary size transition not estimated.
> Juvenile reproduction probability not estimated.
Juvenile maturity transition probability not estimated.
Survival probability sum check (each matrix represented by column in order):
> [,1] [,2] [,3] [,4] [,5]
Min. 0.100 0.100 0.100 0.100 0.100
1st Qu. 0.573 0.573 0.573 0.573 0.573
Median 0.946 0.946 0.946 0.946 0.946
Mean 0.719 0.719 0.719 0.719 0.719
3rd Qu. 0.947 0.947 0.947 0.947 0.947
Max. 0.949 0.949 0.949 0.949 0.949
```

Next we will create the age-hybrid MPM, and edit the structure to include the proper stage description and the proper transition values for the new stage.

```
cyp_hybrid <- add_stage(cyp_lesl_fb_mpm, add_before = 1,
    stage_name = "DormantSeed")
cyp_hybrid$ahstages$propstatus[1] <- 1 # Dormant seeds are propagules
cyp_hybrid <- edit_lM(cyp_hybrid,
    stage3 = c("DormantSeed", "Age1", "Age1", "DormantSeed"),
    stage2 = c("DormantSeed", "DormantSeed", "rep", "rep"),
    eststage3 = c(NA, NA, NA, "Age1"),
    eststage2 = c(NA, NA, NA, "rep"),
    givenrate =c(0.5, 0.2, NA, NA),
    multiplier = c(NA, NA, 0.5, 1),
    type =c(1, 1, 3, 2))
summary(cyp_hybrid)
>
> This ahistorical lefkoMat object contains 5 matrices.
>
> Each matrix is square with 8 rows and columns, and a total of 64 elements.
> A total of 45 survival transitions were estimated, with 9 per matrix.
> A total of 25 fecundity transitions were estimated, with 5 per matrix.
> This lefkoMat object covers 1 population, 1 patch, and 5 time steps.
>
> The dataset contains a total of }74\mathrm{ unique individuals and 320 unique transitions.
```

```
>
> Vital rate modeling quality control:
>
> Survival estimated with 74 individuals and 320 individual transitions.
> Observation probability not estimated.
> Primary size transition not estimated.
> Secondary size transition not estimated.
> Tertiary size transition not estimated.
> Reproduction probability not estimated.
> Fecundity estimated with 51 individuals and }118\mathrm{ individual transitions.
> Juvenile survival not estimated.
> Juvenile observation probability not estimated.
> Juvenile primary size transition not estimated.
> Juvenile secondary size transition not estimated.
> Juvenile tertiary size transition not estimated.
> Juvenile reproduction probability not estimated.
> Juvenile maturity transition probability not estimated.
>
> Survival probability sum check (each matrix represented by column in order):
> [,1] [,2] [,3] [,4] [,5]
> Min. 0.100 0.100 0.100 0.100 0.100
> 1st Qu. 0.575 0.575 0.575 0.575 0.575
> Median 0.946 0.946 0.946 0.946 0.946
```



```
> 3rd Qu. 0.947 0.947 0.947 0.947 0.947
> Max. 0.949 0.949 0.949 0.949 0.949
```

Finally some analyses.

| lambda3 (cyp_hybrid) |  |  |  |  |
| :---: | :---: | :---: | :---: | :---: |
|  |  | patch | year2 | lambda |
| > 1 | 1 | 1 | 2004 | 1.125229 |
| $>2$ | 1 | 1 | 2005 | 1.265645 |
| > 3 | 1 | 1 | 2006 | 1.072010 |
| > 4 | 1 | 1 | 2007 | 1.055135 |
| > 5 | 1 | 1 | 2008 | 1.064150 |

### 16.0.8 Simple raw (empirical) age-by-stage model

This life history model assumes 7 stages, including 4 aboveground adult stages of various sizes and 1 belowground adult stage.

The stageframe can be programmed as follows:

```
data(lathyrus)
sizevector <- c(0, 100, 13, 127, 3730, 3800, 0)
stagevector <- c("Sd", "Sdl", "VSm", "Sm", "VLa", "Flo", "Dorm")
repvector <- c(0, 0, 0, 0, 0, 1, 0)
obsvector <- c(0, 1, 1, 1, 1, 1, 0)
matvector <- c(0, 0, 1, 1, 1, 1, 1)
immvector <- c(1, 1, 0, 0, 0, 0, 0)
```



Figure 16.8: Raw age-by-stage life history model for Lathyrus vernus

```
propvector <- c(1, 0, 0, 0, 0, 0, 0)
indataset <- c(0, 1, 1, 1, 1, 1, 1)
binvec <- c(0, 100, 11, 103, 3500, 3800, 0.5)
lathframe_raw <- sf_create(sizes = sizevector, stagenames = stagevector,
    repstatus = repvector, obsstatus = obsvector, matstatus = matvector,
    immstatus = immvector, indataset = indataset, binhalfwidth = binvec,
    propstatus = propvector)
```

Next we will standardize the dataset, input the supplemental data, and create the MPM.

```
lathyrus$indiv_id <- paste(lathyrus$SUBPLOT, lathyrus$GENET)
lathvert_raw <- verticalize3(lathyrus, noyears = 4, firstyear = 1988,
    patchidcol = "SUBPLOT", individcol = "indiv_id", blocksize = 9,
    juvcol = "Seedling1988", sizeacol = "Volume88", repstracol = "FCODE88",
    fecacol = "Intactseed88", deadacol = "Dead1988", nonobsacol = "Dormant1988",
    stageassign = lathframe_raw, stagesize = "sizea", censorcol = "Missing1988",
    censorkeep = NA, censorRepeat = TRUE, censor = TRUE)
lathvert_raw_small <- subset(lathvert_raw, firstseen > 1988)
lathsupp2 <- supplemental(stage3 = c("Sd", "Sdl", "Sd", "Sdl"),
    stage2 = c("Sd", "Sd", "rep", "rep"), givenrate = c(0.345, 0.054, NA, NA),
    multiplier = c(NA, NA, 0.345, 0.054), type = c(1, 1, 3, 3),
    stageframe = lathframe_raw, historical = FALSE, stagebased = TRUE,
    agebased = TRUE)
lathmat2p_raw <- arlefko2(data = lathvert_raw_small, stageframe = lathframe_raw,
    supplement = lathsupp2, stages = c("stage3", "stage2", "stage1"),
    patch = "all", patchcol = "patchid", yearcol = "year2", agecol = "obsage",
    indivcol = "individ")
summary(lathmat2p_raw)
>
> This ahistorical lefkoMat object contains 12 matrices.
>
> Each matrix is square with }21\mathrm{ rows and columns, and a total of 441 elements.
> A total of 209 survival transitions were estimated, with 17.417 per matrix.
> A total of 12 fecundity transitions were estimated, with 1 per matrix.
> This lefkoMat object covers 1 population, 6 patches, and 2 time steps.
>
> The dataset contains a total of 345 unique individuals and 531 unique transitions.
>
> Survival probability sum check (each matrix represented by column in order):
> [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11] [,12]
> Min. 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000
> 1st Qu. 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000
> Median 0.000 0.000 0.000 0.399 0.000 0.399 0.000 0.399 0.000 0.399 0.000 0.000
> Mean 0.281 0.349 0.261 0.385 0.274 0.463 0.268 0.423 0.223 0.388 0.234 0.279
> 3rd Qu. 0.399 0.778 0.399 0.750 0.399 1.000 0.399 0.778 0.399 0.818 0.399 0.399
> Max. 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000
```

Finally, some basic analyses.

| lambda3(lathmat2p_raw) |  |  |  |  |
| :--- | ---: | ---: | ---: | ---: |
| $>$ | pop | patch | year2 | lambda |
| $>$ | 1 | 1 | 1 | 1989 | 0.345

### 16.0.9 Complex function-based age-by-stage model



Figure 16.9: Complex life history model of Lathyrus vernus using log leaf volume as the size classification metric

This life history model includes 21 life history stages, of which 19 are adult size-based stages. Here we program the stageframe.

```
data(lathyrus)
```

```
sizevector <- c(0, 4.6, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 1, 2, 3, 4, 5, 6, 7, 8, 9)
stagevector <- c("Sd", "Sdl", "Dorm", "Sz1nr", "Sz2nr", "Sz3nr", "Sz4nr",
    "Sz5nr","Sz6nr", "Sz7nr", "Sz8nr", "Sz9nr", "Sz1r", "Sz2r", "Sz3r", "Sz4r",
    "Sz5r", "Sz6r", "Sz7r", "Sz8r", "Sz9r")
repvector <- c(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1)
obsvector <- c(0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1)
matvector <- c(0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1)
immvector <- c(1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
propvector <- c(1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
indataset <- c(0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1)
binvec <- c(0, 4.6, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5,
                    0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5)
minima <- c(1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2)
maxima <- c(NA, 1, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,
    NA, NA, NA, NA, NA)
comments <- c("Dormant seed", "Seedling", "Dormant", "Size 1 Veg", "Size 2 Veg",
    "Size 3 Veg", "Size 4 Veg", "Size 5 Veg", "Size 6 Veg", "Size 7 Veg",
    "Size 8 Veg", "Size 9 Veg", "Size 1 Flo", "Size 2 Flo", "Size 3 Flo",
    "Size 4 Flo", "Size 5 Flo", "Size 6 Flo", "Size 7 Flo", "Size 8 Flo",
    "Size 9 Flo")
lathframeln <- sf_create(sizes = sizevector, stagenames = stagevector,
    repstatus = repvector, obsstatus = obsvector, propstatus = propvector,
    immstatus = immvector, matstatus = matvector, indataset = indataset,
    binhalfwidth = binvec, minage = minima, maxage = maxima, comments = comments)
```

Next we will standardize the dataset, input the supplemental data, create the vital rate models, and create our MPM.

```
lathyrus$indiv_id <- paste(lathyrus$SUBPLOT, lathyrus$GENET)
lathvertln <- verticalize3(lathyrus, noyears = 4, firstyear = 1988,
    patchidcol = "SUBPLOT", individcol = "indiv_id", blocksize = 9,
    juvcol = "Seedling1988", sizeacol = "lnVol88", repstracol = "FCODE88",
    fecacol = "Intactseed88", deadacol = "Dead1988", nonobsacol = "Dormant1988",
    stageassign = lathframeln, stagesize = "sizea", censorcol = "Missing1988",
    censorkeep = NA, censorRepeat = TRUE, NAas0 = TRUE, censor = TRUE)
lathvertln_small <- subset(lathvertln, firstseen > 1988)
lathvertln$feca3 <- round(lathvertln$feca3)
lathvertln$feca2 <- round(lathvertln$feca2)
lathvertln$feca1 <- round(lathvertln$feca1)
lathsupp2 <- supplemental(stage3 = c("Sd", "Sdl", "Sd", "Sdl"),
    stage2 = c("Sd", "Sd", "rep", "rep"), givenrate = c(0.345, 0.054, NA, NA),
    multiplier = c(NA, NA, 0.345, 0.054), type = c(1, 1, 3, 3),
```

```
    stageframe = lathframeln, historical = FALSE, stagebased = TRUE,
    agebased = TRUE)
lathmodelsln2 <- modelsearch(lathvertln_small, historical = FALSE,
    approach = "mixed", suite = "main", bestfit = "AICc&k", juvestimate = "Sdl",
    vitalrates = c("surv", "obs", "size", "repst", "fec"), sizedist = "gaussian",
    fecdist = "negbin", indiv = "individ", year = "year2", age = "obsage",
    year.as.random = TRUE, patch.as.random = TRUE, show.model.tables = TRUE,
    fec.zero = TRUE, test.age = TRUE, quiet = "partial")
>
> Developing global model of survival probability...
> Global model of survival probability developed. Proceeding with model dredge...
>
Developing global model of observation probability...
>
Global model of observation probability developed. Proceeding with model dredge...
Developing global model of primary size...
Global model of primary size developed. Proceeding with model dredge...
Developing global model of reproduction probability...
Global model of reproduction probability developed. Proceeding with model dredge...
Developing global model of fecundity...
Global model of fecundity developed. Proceeding with model dredge...
Developing global model of juvenile survival probability...
Global model of juvenile survival probability developed. Proceeding with model dredge...
Warning: Juvenile maturity status in time t+1 appears to be constant, and so will be set to constan
Developing global model of juvenile observation probability...
Global model of juvenile observation probability developed. Proceeding with model dredge...
Developing global model of juvenile primary size...
Could not properly estimate a global model for juvenile size.
Warning: Juvenile reproductive status in time t+1 appears to be constant, and so will be set to con
Finished selecting best-fit models.
lathmat2age <- aflefko2(year = "all", stageframe = lathframeln,
    modelsuite = lathmodelsln2, data = lathvertln_small, supplement = lathsupp2,
    continue = TRUE, reduce = FALSE)
summary(lathmat2age)
```

```
>
> This ahistorical lefkoMat object contains 2 matrices.
>
> Each matrix is square with 63 rows and columns, and a total of 3969 elements.
> A total of }1434\mathrm{ survival transitions were estimated, with }717\mathrm{ per matrix.
> A total of 72 fecundity transitions were estimated, with 36 per matrix.
> This lefkoMat object covers 1 population, 1 patch, and 2 time steps.
> The dataset contains a total of 345 unique individuals and 531 unique transitions.
>
> Vital rate modeling quality control:
>
Survival estimated with }247\mathrm{ individuals and 347 individual transitions.
Observation estimated with 203 individuals and 294 individual transitions.
Primary size estimated with }191\mathrm{ individuals and 266 individual transitions.
Secondary size transition not estimated.
Tertiary size transition not estimated.
Reproductive status estimated with }191\mathrm{ individuals and 266 individual transitions.
Fecundity estimated with 16 individuals and 19 individual transitions.
Juvenile survival estimated with }184\mathrm{ individuals and }184\mathrm{ individual transitions.
Juvenile observation estimated with }137\mathrm{ individuals and 137 individual transitions.
Juvenile primary size transition not estimated.
Juvenile secondary size transition not estimated.
Juvenile tertiary size transition not estimated.
Juvenile reproduction probability not estimated.
Juvenile maturity transition probability not estimated.
>
Survival probability sum check (each matrix represented by column in order):
> [,1] [,2]
Min. 0.000 0.000
1st Qu. 0.000 0.000
Median 0.185 0.185
Mean 0.352 0.365
3rd Qu. 0.861 0.955
Max. 1.000 1.000
```

Now some basic analyses.

```
lambda3(lathmat2age)
> pop patch year2 lambda
> 1 <NA> 1989 0.345
> 2 1 <NA> 1990 0.345
```


### 16.0.10 Ahistorical discretized integral projection model (IPM)

This life history model includes 103 stages, of which 100 are adult, size-classified IPM bins. Here we program the stageframe.

```
data(lathyrus)
```

sizevector <- c(0, 100, 0, 1, 9)


Figure 16.10: Life history model of Lathyrus vernus for use in IPM construction. Not all adult stages are shown.

```
stagevector <- c("Sd", "Sdl", "Dorm", "ipm", "ipm")
repvector <- c(0, 0, 0, 1, 1)
obsvector <- c(0, 1, 0, 1, 1)
matvector <- c(0, 0, 1, 1, 1)
immvector <- c(1, 1, 0, 0, 0)
propvector <- c(1, 0, 0, 0, 0)
indataset <- c(0, 1, 1, 1, 1)
binvec <- c(0, 100, 0.5, 0.5, 0.5)
comments <- c("Dormant seed", "Seedling", "Dormant", "ipm adult stage",
    "ipm adult stage")
lathframeipm <- sf_create(sizes = sizevector, stagenames = stagevector,
    repstatus = repvector, obsstatus = obsvector, propstatus = propvector,
    immstatus = immvector, matstatus = matvector, comments = comments,
    indataset = indataset, binhalfwidth = binvec, ipmbins = 100, roundsize = 3)
```

Now we standardize the dataset, input the supplemental data, estimate the vital rate models, and create the IPM.

```
lathyrus$indiv_id <- paste(lathyrus$SUBPLOT, lathyrus$GENET)
lathvertipm <- verticalize3(lathyrus, noyears = 4, firstyear = 1988,
    individcol = "indiv_id", blocksize = 9, juvcol = "Seedling1988",
    sizeacol = "lnVol88", repstracol = "FCODE88", fecacol = "Intactseed88",
    deadacol = "Dead1988", nonobsacol = "Dormant1988", stageassign = lathframeipm,
    stagesize = "sizea", censorcol = "Missing1988", censorkeep = NA,
    censorRepeat = TRUE, censor = TRUE, NAas0 = TRUE, NRasRep = TRUE)
> Warning: Some stages occurring in the dataset do not match any characteristics
> in the input stageframe.
lathvertipm$feca3 <- round(lathvertipm$feca3)
lathvertipm$feca2 <- round(lathvertipm$feca2)
lathvertipm$feca1 <- round(lathvertipm$feca1)
lathsupp2 <- supplemental(stage3 = c("Sd", "Sdl", "Sd", "Sdl"),
    stage2 = c("Sd", "Sd", "rep", "rep"),
    givenrate = c(0.345, 0.054, NA, NA),
    multiplier = c(NA, NA, 0.345, 0.054),
    type = c(1, 1, 3, 3), stageframe = lathframeipm, historical = FALSE)
lathmodels2ipm <- modelsearch(lathvertipm, historical = FALSE,
    approach = "mixed", suite = "size", juvestimate = "Sdl",
    vitalrates = c("surv", "obs", "size", "fec"), bestfit = "AICc&k",
    sizedist = "gaussian", fecdist = "negbin", fec.zero = TRUE, indiv = "individ",
    year = "year2", year.as.random = TRUE, juvsize = TRUE,
    show.model.tables = TRUE, quiet = "partial")
>
> Developing global model of survival probability...
>
> Could not properly estimate a global model for survival probability.
>
    Developing global model of observation probability...
```

```
>
Global model of observation probability developed. Proceeding with model dredge...
Developing global model of primary size...
Global model of primary size developed. Proceeding with model dredge...
Reproduction probability will be treated as constant.
Developing global model of fecundity...
Global model of fecundity developed. Proceeding with model dredge...
Developing global model of juvenile survival probability...
Global model of juvenile survival probability developed. Proceeding with model dredge...
Warning: Juvenile maturity status in time t+1 appears to be constant, and so will be set to constan
Developing global model of juvenile observation probability...
Global model of juvenile observation probability developed. Proceeding with model dredge...
Developing global model of juvenile primary size...
Could not properly estimate a global model for juvenile size.
Warning: Juvenile reproductive status in time t+1 appears to be constant, and so will be set to con
Finished selecting best-fit models.
lathmat2ipm <- flefko2(stageframe = lathframeipm, modelsuite = lathmodels2ipm,
    supplement = lathsupp2, data = lathvertipm, reduce = FALSE)
summary(lathmat2ipm)
>
> This ahistorical lefkoMat object contains 3 matrices.
>
> Each matrix is square with 103 rows and columns, and a total of 10609 elements.
A total of }30608\mathrm{ survival transitions were estimated, with 10202.667 per matrix.
A total of }600\mathrm{ fecundity transitions were estimated, with 200 per matrix.
This lefkoMat object covers 1 population, 1 patch, and 3 time steps.
The dataset contains a total of 1053 unique individuals and 2527 unique transitions.
Vital rate modeling quality control:
Survival not estimated.
Observation estimated with }858\mathrm{ individuals and 2121 individual transitions.
Primary size estimated with }845\mathrm{ individuals and }1916\mathrm{ individual transitions.
Secondary size transition not estimated.
Tertiary size transition not estimated.
Reproduction probability not estimated.
```

```
> Fecundity estimated with 335 individuals and 599 individual transitions.
> Juvenile survival estimated with 281 individuals and 281 individual transitions.
> Juvenile observation estimated with 210 individuals and 210 individual transitions.
> Juvenile primary size transition not estimated.
> Juvenile secondary size transition not estimated.
> Juvenile tertiary size transition not estimated.
> Juvenile reproduction probability not estimated.
> Juvenile maturity transition probability not estimated.
>
> Survival probability sum check (each matrix represented by column in order):
> [,1] [,2] [,3]
> Min. 0.000 0.000 0.000
> 1st Qu. 1.000 1.000 1.000
> Median 1.000 1.000 1.000
> Mean 0.984 0.984 0.984
> 3rd Qu. 1.000 1.000 1.000
> Max. 1.000 1.000 1.000
```

Now some basic analyses.

| lambda3(lathmat2ipm) |  |  |  |  |
| :--- | ---: | ---: | ---: | ---: |
| $>$ | pop | patch | year2 | lambda |
| $>$ | 1 | 1 | <NA> | 1988 |
|  | 1.0000000 |  |  |  |
| $>$ | 2 | 1 | <NA> | 1989 |
| $>$ | 3 | 1 | <NA> | 1990 |

### 16.0.11 Historical discretized integral projection model (IPM)

This life history model includes 103 stages, of which 100 are adult, size-classified IPM bins. Here we program the stageframe.

```
data(lathyrus)
sizevector <- c(0, 100, 0, 1, 9)
stagevector <- c("Sd", "Sdl", "Dorm", "ipm", "ipm")
repvector <- c(0, 0, 0, 1, 1)
obsvector <- c(0, 1, 0, 1, 1)
matvector <- c(0, 0, 1, 1, 1)
immvector <- c(1, 1, 0, 0, 0)
propvector <- c(1, 0, 0, 0, 0)
indataset <- c(0, 1, 1, 1, 1)
binvec <- c(0, 100, 0.5, 0.5, 0.5)
comments <- c("Dormant seed", "Seedling", "Dormant", "ipm adult stage",
    "ipm adult stage")
lathframeipm <- sf_create(sizes = sizevector, stagenames = stagevector,
    repstatus = repvector, obsstatus = obsvector, propstatus = propvector,
    immstatus = immvector, matstatus = matvector, comments = comments,
    indataset = indataset, binhalfwidth = binvec, ipmbins = 100, roundsize = 3)
```

Now we standardize the dataset, input the supplemental data, estimate the vital rate models, and create the IPM.


Figure 16.11: Life history model of Lathyrus vernus for use in IPM construction. Not all adult stages are shown.

```
lathyrus$indiv_id <- paste(lathyrus$SUBPLOT, lathyrus$GENET)
lathvertipm <- verticalize3(lathyrus, noyears = 4, firstyear = 1988,
    individcol = "indiv_id", blocksize = 9, juvcol = "Seedling1988",
    sizeacol = "lnVol88", repstracol = "FCODE88", fecacol = "Intactseed88",
    deadacol = "Dead1988", nonobsacol = "Dormant1988", stageassign = lathframeipm,
    stagesize = "sizea", censorcol = "Missing1988", censorkeep = NA,
    censorRepeat = TRUE, censor = TRUE, NAasO = TRUE, NRasRep = TRUE)
> Warning: Some stages occurring in the dataset do not match any characteristics
> in the input stageframe.
lathvertipm$feca3 <- round(lathvertipm$feca3)
lathvertipm$feca2 <- round(lathvertipm$feca2)
lathvertipm$feca1 <- round(lathvertipm$feca1)
lathsupp3 <- supplemental(stage3 = c("Sd","Sd","Sdl","Sdl","npr","Sd","Sdl"),
    stage2 = c("Sd", "Sd", "Sd", "Sd", "Sdl", "rep", "rep"),
    stage1 = c("Sd", "rep", "Sd", "rep", "Sd", "mat", "mat"),
    eststage3 = c(NA, NA, NA, NA, "npr", NA, NA),
    eststage2 = c(NA, NA, NA, NA, "Sdl", NA, NA),
    eststage1 = c(NA, NA, NA, NA, "Sdl", NA, NA),
    givenrate = c(0.345, 0.345, 0.054, 0.054, NA, NA, NA),
    multiplier = c(NA, NA, NA, NA, NA, 0.345, 0.054),
    type = c(1, 1, 1, 1, 1, 3, 3), type_t12 = c(1, 2, 1, 2, 1, 1, 1),
    stageframe = lathframeipm, historical = TRUE)
lathmodels3ipm <- modelsearch(lathvertipm, historical = TRUE, approach= "mixed",
    suite = "size", vitalrates = c("surv", "obs", "size", "fec"),
    juvestimate = "Sdl", bestfit = "AICc&k", sizedist = "Gaussian",
    fecdist = "negbin", fec.zero = TRUE, indiv = "individ", year = "year2",
    year.as.random = TRUE, juvsize = TRUE, show.model.tables = TRUE,
    quiet = "partial")
>
    Developing global model of survival probability...
    Global model of survival probability developed. Proceeding with model dredge...
    Developing global model of observation probability...
    Global model of observation probability developed. Proceeding with model dredge...
    Developing global model of primary size...
    Global model of primary size developed. Proceeding with model dredge...
    Reproduction probability will be treated as constant.
    Developing global model of fecundity...
    Global model of fecundity developed. Proceeding with model dredge...
```

```
> Developing global model of juvenile survival probability...
>
> Global model of juvenile survival probability developed. Proceeding with model dredge...
> Warning: Juvenile maturity status in time t+1 appears to be constant, and so will be set to constan
>
> Developing global model of juvenile observation probability...
>
> Global model of juvenile observation probability developed. Proceeding with model dredge...
>
Developing global model of juvenile primary size...
Could not properly estimate a global model for juvenile size.
Warning: Juvenile reproductive status in time t+1 appears to be constant, and so will be set to con
Finished selecting best-fit models.
lathmat3ipm <- flefko3(stageframe = lathframeipm, modelsuite = lathmodels3ipm,
    supplement = lathsupp3, data = lathvertipm, reduce = FALSE,
    sparse_output = TRUE)
summary(lathmat3ipm)
>
> This historical lefkoMat object contains 3 matrices.
Each matrix is square with }10609\mathrm{ rows and columns, and a total of 112550881 elements.
A total of 3106216 survival transitions were estimated, with 1035405.333 per matrix.
A total of }60600\mathrm{ fecundity transitions were estimated, with 20200 per matrix.
This lefkoMat object covers 1 population, 1 patch, and 3 time steps.
The dataset contains a total of }1053\mathrm{ unique individuals and 2527 unique transitions.
Vital rate modeling quality control:
Survival estimated with }931\mathrm{ individuals and 2246 individual transitions.
Observation estimated with }858\mathrm{ individuals and 2121 individual transitions.
Primary size estimated with }845\mathrm{ individuals and 1916 individual transitions.
Secondary size transition not estimated.
Tertiary size transition not estimated.
Reproduction probability not estimated.
Fecundity estimated with 335 individuals and 599 individual transitions.
Juvenile survival estimated with }281\mathrm{ individuals and 281 individual transitions.
Juvenile observation estimated with 210 individuals and 210 individual transitions.
Juvenile primary size transition not estimated.
Juvenile secondary size transition not estimated.
Juvenile tertiary size transition not estimated.
Juvenile reproduction probability not estimated.
Juvenile maturity transition probability not estimated.
Survival probability sum check (each matrix represented by column in order):
    [,1] [,2] [,3]
Min. 0.000 0.000 0.000
```

> 1st Qu. 0.9971 .0001 .000
> Median 1.0001 .0001 .000
$\begin{array}{ll}> & \text { Mean } 0.9730 .9720 .974\end{array}$
> 3rd Qu. 1.0001 .0001 .000
> Max. 1.0001 .0001 .000
Now some basic analyses.

```
lambda3(lathmat3ipm)
> pop patch year2 lambda
> 1 1 <NA> 1988 0.9971545
> 2 1 <NA> 1989 0.9999400
> 3 1 <NA> 1990 0.9999195
```


## Bibliography

Åberg, P. (1990). Measuring size and choosing category size for a transition matrix study of the seaweed Ascophyllum nodosum. Marine Ecology Progress Series, 63:281-287.

Ballard, W. B., Whitman, J. S., and Reed, D. J. (1991). Population dynamics of moose in south-central Alaska. Wildlife Monographs, 114:3-49.

Bartoń, K. A. (2014). MuMIn: multi-model inference.
Bates, D., Maechler, M., Bolker, B., and Walker, S. (2015). Fitting linear mixed-effects models using lme4. Journal of Statistical Software, 67(1):1-48.

Beamonte-Barrientos, R., Velando, A., Drummond, H., and Torres, R. (2010). Senescence of maternal effects: aging influences egg quality and rearing capacities of a long-lived bird. The American Naturalist, 175(4):469-480.
Beissinger, S. R. and Westphal, M. I. (1998). On the use of demographic models of population viability in endangered species management. Journal of Wildlife Management, 62(3):821-841.

Brooks, M. E., Kristensen, K., Benthem, K. J. v., Magnusson, A., Berg, C. W., Nielsen, A., Skaug, H. J., Machler, M., and Bolker, B. M. (2017). glmmTMB balances speed and flexibility among packages for zero-inflated generalized linear mixed modeling. The $R$ Journal, 9(2):378-400.

Burnham, K. P. and Anderson, D. R. (2002). Model selection and multimodel inference: a practical information-theoretic approach. Springer-Verlag New York, Inc., New York, New York, USA.

Caswell, H. (1989). Analysis of life table response experiments i. decomposition of effects on population growth rate. Ecological Modelling, 46(3-4):221-237.

Caswell, H. (2001). Matrix population models: construction, analysis, and interpretation. Sinauer Associates, Inc., Sunderland, Massachusetts, USA, second edition edition.

Caswell, H. (2010). Life table response experiment analysis of the stochastic growth rate. Journal of Ecology, 98(2):324-333.

Caswell, H. (2019). Sensitivity analysis: matrix methods in demography and ecology. Demographic Research Methods. Springer Nature, Cham, Switerland.

Caswell, H. and Salguero-Gómez, R. (2013). Age, stage and senescence in plants. Journal of Ecology, 101(3):585-595.

Caswell, H., Vries, C. d., Hartemink, N., Roth, G., and Daalen, S. F. v. (2018). Age $\times$ stage-classified demographic analysis: a comprehensive approach. Ecological Monographs, 88(4):560-584.

Childs, D. Z., Rees, M., Rose, K. E., Grubb, P. J., and Ellner, S. P. (2003). Evolution of complex flowering strategies: an age- and size-structured integral projection model. Proceedings Of The Royal Society Of London Series B-Biological Sciences, 270(1526):1829-1838.

Crone, E. E., Ellis, M. M., Morris, W. F., Stanley, A., Bell, T., Bierzychudek, P., Ehrlén, J., Kaye, T. N., Knight, T. M., Lesica, P., Oostermeijer, G., Quintana-Ascencio, P. F., Ticktin, T., Valverde, T., Williams, J. L., Doak, D. F., Ganesan, R., McEachern, K., Thorpe, A. S., and Menges, E. S. (2013). Ability of matrix models to explain the past and predict the future of plant populations. Conservation Biology, 27(5):968-978.

Dahlgren, J. P. and Ehrlén, J. (2011). Incorporating environmental change over succession in an integral projection model of population dynamics of a forest herb. Oikos, 120(8):1183-1190.

Davison, R., Jacquemyn, H., Adriaens, D., Honnay, O., de Kroon, H., and Tuljapurkar, S. (2010). Demographic effects of extreme weather events on a short-lived calcareous grassland species: stochastic life table response experiments. Journal of Ecology, 98(2):255-267.

Davison, R., Nicole, F., Jacquemyn, H., and Tuljapurkar, S. (2013). Contributions of covariance: decomposing the components of stochastic population growth in Cypripedium calceolus. The American Naturalist, 181(3):410-420.

Davison, R., Stadman, M., and Jongejans, E. (2019). Stochastic effects contribute to population fitness differences. Ecological Modelling, 408:108760.

Descamps, S., Boutin, S., Berteaux, D., and Gaillard, J.-M. (2008). Age-specific variation in survival, reproductive success and offspring quality in red squirrels: evidence of senescence. Oikos, 117(9):1406-1416.
deVries, C. and Caswell, H. (2018). Demography when history matters: construction and analysis of second-order matrix population models. Theoretical Ecology, 11(2):129-140.

Doak, D. F., Waddle, E., Langendorf, R. E., Louthan, A. M., Isabelle Chardon, N., Dibner, R. R., Keinath, D. A., Lombardi, E., Steenbock, C., Shriver, R. K., Linares, C., Begoña Garcia, M., Funk, W. C., Fitzpatrick, S. W., Morris, W. F., and DeMarche, M. L. (2021). A critical comparison of integral projection and matrix projection models for demographic analysis. Ecological Monographs, 91(2):e01447. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/ecm.1447.

Dogra, H. and Dani, K. G. S. (2019). Defining features of age-specific fertility and seed quality in senescing indeterminate annuals. American Journal of Botany, 106(0):604-610.

Easterling, M. R., Ellner, S. P., and Dixon, P. M. (2000). Size-Specific Sensitivity: Applying a New Structured Population Model. Ecology, 81(3):694-708.

Ehrlén, J. (1995). Demography of the perennial herb Lathyrus vernus. I. Herbivory and individual performance. Journal of Ecology, 83(2):287-295. Publisher: [Wiley, British Ecological Society].

Ehrlén, J. (2000). The dynamics of plant populations: does the history of individuals matter? Ecology, 81(6):1675-1684.

Ehrlén, J. (2002). Assessing the lifetime consequences of plant-animal interactions for the perennial herb Lathyrus vernus (Fabaceae). Perspectives in Plant Ecology, Evolution and Systematics, $5(3): 145-163$.

Ehrlén, J. and Eriksson, O. (1996). Seedling recruitment in the perennial herb Lathyrus vernus. Flora, 191(4):377-383.

Ehrlén, J. and Lehtila, K. (2002). How perennal are perennial plants? Oikos, 98:308-322.
Ehrlén, J. and Munzbergova, Z. (2009). Timing of flowering: opposed selection on different fitness components and trait covariation. The American Naturalist, 173(6):819-830.

Ehrlén, J. and Van Groenendael, J. (2001). Storage and the delayed costs of reproduction in the understorey perennial Lathyrus vernus. Journal of Ecology, 89(2):237-246.

Ellner, S. P. and Rees, M. (2006). Integral projection models for species with complex demography. American Naturalist, 167(3):410-428.

Garcia, M. B., Dahlgren, J. P., and Ehrlén, J. (2011). No evidence of senescence in a 300-year-old mountain herb. Journal of Ecology, 99(6):1424-1430.

Greenwood, M. (1938). The first life table. Notes and Records of the Royal Society of London, 1(2):7072. Publisher: Royal Society.

Honnay, O., Coart, E., Butaye, J., Adriaens, D., van Glabeke, S., and Roldan-Ruiz, I. (2006). Low impact of present and historical landscape configuration on the genetics of fragmented Anthyllis vulneraria populations. Biological Conservation, 127(4):411-419. Publisher: Elsevier.

Jenks, G. F. (1967). The data model concept in statistical mapping. International Yearbook of Cartography, 7:186-190.

Jensen, A. (1995). Simple density-dependent matrix model for population projection. Ecological Modelling, 77(1):43-48. Publisher: Elsevier.

Jones, O. R., Barks, P., Stott, I., James, T. D., Levin, S., Petry, W. K., Capdevila, P., Che-Castaldo, J., Jackson, J., Römer, G., Schuette, C., Thomas, C. C., and Salguero-Gómez, R. (2022). Rcompadre and Rage - Two R packages to facilitate the use of the COMPADRE and COMADRE databases and calculation of life-history traits from matrix population models. Methods in Ecology and Evolution, 13(4):770-781. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1111/2041-210X.13792.

Kendall, B. E., Fujiwara, M., Diaz-Lopez, J., Schneider, S., Voigt, J., and Wiesner, S. (2019). Persistent problems in the construction of matrix population models. Ecological Modelling, 406:33-43.

Krauss, J., Steffan-Dewenter, I., and Tscharntke (2004). Landscape occupancy and local population size depends on host plant distribution in the butterfly Cupido minimus. Biological Conservation, 120(3):355-361. Publisher: Elsevier.

Lefkovitch, L. (1965). The study of population growth in organisms grouped by stages. Biometrics, 21(1):1-18.

Leslie, P. H. (1945). On the use of matrices in certain population mathematics. Biometrika, 33(3):183212. Publisher: [Oxford University Press, Biometrika Trust].

Leslie, P. H. (1959). The properties of a certain lag type of population growth and the influence of an external random factor on a number of such populations. Physiological Zoology, 32(3):151-159. Publisher: The University of Chicago Press.

Merow, C., Dahlgren, J. P., Metcalf, C. J. E., Childs, D. Z., Evans, M. E. K., Jongejans, E., Record, S., Rees, M., Salguero-Gómez, R., and McMahon, S. M. (2014). Advancing population ecology with integral projection models: a practical guide. Methods in Ecology and Evolution, 5(2):99-110.

Metcalf, C. J. E., McMahon, S. M., Salguero-Gómez, R., and Jongejans, E. (2013). IPMpack: an R package for integral projection models. Methods in Ecology and Evolution, 4(2):195-200.

Morris, W. F. and Doak, D. F. (2002). Quantitative conservation biology: theory and practice of population viability analysis. Sinauer Associates, Inc., Sunderland, Massachusetts, USA.

Pantoja, P. O., Paine, C. T., and Vallejo-Marin, M. (2018). Natural selection snd outbreeding depression suggest adaptive differentiation in the invasive range of a clonal plant. Proceedings Of The Royal Society B-Biological Sciences, 285:20181091.

Piessens, K., Adriaens, D., Jacquemyn, H., and Honnay, O. (2009). Synergistic effects of an extreme weather event and habitat fragmentation on a specialised insect herbivore. Oecologia, 159(1):117126.

Salguero-Gómez, R., Jones, O. R., Archer, C. R., Bein, C., Buhr, Farack, C., Gottschalk Fränce, Hartmann, A., Henning, A., Hoppe, G., Römer, G., Ruoff, T., Sommer, V., Wille, J., Voigt, J., Zeh, S., Vieregg, D., Buckley, Y. M., Che-Castaldo, J., Hodgson, D., Scheuerlein, A., Caswell, H., and Vaupel, J. W. (2016). COMADRE: a global data base of animal demography. Journal of Animal Ecology, 85(2):371-384.

Salguero-Gómez, R. and Casper, B. B. (2010). Keeping plant shrinkage in the demographic loop. Journal of Ecology, 98:312-323.

Salguero-Gómez, R., Jones, O. R., Archer, C. R., Buckley, Y. M., Che-Castaldo, J., Caswell, H., Hodgson, D., Scheuerlein, A., Conde, D. A., Brinks, E., de Buhr, H., Farack, C., Gottschalk, F., Hartmann, A., Henning, A., Hoppe, G., Römer, G., Runge, J., Ruoff, T., Wille, J., Zeh, S., Davison, R., Vieregg, D., Baudisch, A., Altwegg, R., Colchero, F., Dong, M., de Kroon, H., Lebreton, J.-D., Metcalf, C. J. E., Neel, M. M., Parker, I. M., Takada, T., Valverde, T., Vélez-Espino, L. A., Wardle, G. M., Franco, M., and Vaupel, J. W. (2015). The Compadre Plant Matrix Database: an open online repository for plant demography. Journal of Ecology, 103(1):202-218.

Shefferson, R. P., Kull, T., Hutchings, M. J., Selosse, M.-A., Jacquemyn, H., Kellett, K. M., Menges, E. S., Primack, R. B., Tuomi, J., Alahuhta, K., Hurskainen, S., Alexander, H. M., Anderson, D. S., Brys, R., Brzosko, E., Dostálik, S., Gregg, K. B., Ipser, Z., Jäkäläniemi, A., Jersáková, J., Kettle, W. D., McCormick, M. K., Mendoza, A., Miller, M. T., Moen, A., Øien, D.-I., Püttsepp, ., Roy, M., Sather, N., Sletvold, N., Štipková, Z., Tali, K., Warren II, R. J., and Whigham, D. F. (2018). Drivers of vegetative dormancy across herbaceous perennial plant species. Ecology Letters, 21(5):724-733.

Shefferson, R. P., Kurokawa, S., and Ehrlén, J. (2021). lefko3: analysing individual history through size-classified matrix population models. Methods in Ecology and Evolution, 12(2):378-382. _eprint: https://besjournals.onlinelibrary.wiley.com/doi/pdf/10.1111/2041-210X.13526.

Shefferson, R. P., Mizuta, R., and Hutchings, M. J. (2017). Predicting evolution in response to climate change: the example of sprouting probability in three dormancy-prone orchid species. Royal Society Open Science, 4(1):160647.

Shefferson, R. P., Sandercock, B. K., Proper, J., and Beissinger, S. R. (2001). Estimating dormancy and survival of a rare herbaceous perennial using mark-recapture models. Ecology, 82(1):145-156.

Shefferson, R. P., Warren II, R. J., and Pulliam, H. R. (2014). Life history costs make perfect sprouting maladaptive in two herbaceous perennials. Journal of Ecology, 102(5):1318-1328.

Tuljapurkar, S. (1990). Population dynamics in variable environments. Lecture Notes in Biomathematics. Springer-Verlag Berlin-Heidelberg, Berlin, Germany.
van den Broek, J. (1995). A score test for zero inflation in a Poisson distribution. Biometrics, 51(2):738743.

Wardle, G. M. (1998). A graph theory approach to demographic loop analysis. Ecology, 79(7):25392549.

Yee, T. W. (2015). Vector Generalized Linear and Additive Models: With an Implementation in R.
Yee, T. W. and Wild, C. (1996). Vector generalized additive models. Journal of the Royal Statistical Society, Series B, 58(3):481-493.

Zeileis, A., Kleiber, C., and Jackman, S. (2008). Regression models for count data in R. Journal of Statistical Software, 27(1):1-25. Number: 1.

