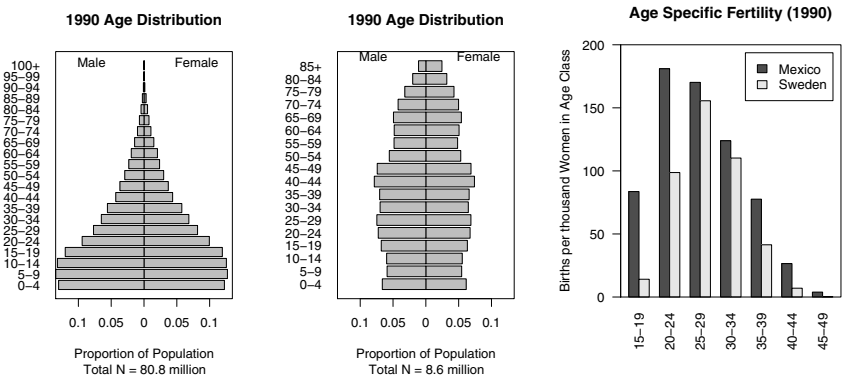
2

Density-independent Demography

Different populations have different numbers of individuals of different ages. Consider the human populations of Mexico and Sweden in 1990. Mexico had more individuals in total than Sweden, and a larger fraction of their population was of child bearing age or younger (Figs. 2.1a, 2.1b).



(a) Mexico (b) Sweden (c) Mexico *vs.* Sweden

Fig. 2.1: Demography of human populations of Mexico and Sweden. Based on 1990 data from US Census Bureau, Population Division, International Programs Center.

In addition, the age-specific fertility rate is higher in Mexico, especially for younger women (Fig. 2.1c). How did this happen, and why does Mexico have so many young people? What are the consequences of this for their culture, their use of resources, their domestic and foreign policies, and their future population growth? How about Sweden?

Demography is the study of populations with special attention to age or stage structure [113]. Originally, age-based human demography was the provenance of actuaries who helped governments keep track of the number citizens

M.H.H. Stevens, *A Primer of Ecology with R*, Use R, DOI: 10.1007/978-0-387-89882-7_2,

of different ages and thus, for instance, know how many would be available for conscription into the military.[1]The demography of a population is the age (or stage) structure and the survival, fertility, and other demographic rates associated with those ages or life history stages. *Age structure* is the number or relative abundance of individuals of different ages or age classes. *Stage structure* is the number or relative abundance of individuals of different stages. Stages are merely useful categories of individuals, such as size classes (e.g. diameters of tropical trees) or life history stages (e.g. egg, larvae, and adult anurans). Stages are particularly useful when (i) age is difficult to determine, and/or (ii) when stage is a better predictor of demographic rates (e.g. birth, death, survival) than is age. Demography is, in part, the study of how demographic rates vary among ages or stages, and the consequences of those differences.

There are a few ways to study a population's demography, and all ecology text books can provide examples. *Life tables* are lists of important demographic parameters such as survivorship, birth and death rates each age or age class.

Commonly, both age and stage based demography now take advantage of matrix algebra to simplify and synthesize age and stage specific demography [23]. This approach is essential when individuals don't proceed through stages in a simple sequential manner, perhaps reverting to an "earlier" stage. When used with age-based demography, these matrices are referred to as Leslie matrices [107]. L. P. Lefkovitch [100] generalized this approach to allow for complex demography. This could include, for instance, regression from a large size class to a smaller size class (e.g. a two-leaved woodland perennial to a one-leaved stage). Using matrices to represent a population's demography allows us to use the huge workshop of linear algebra tools to understand and predict growth in structured populations. Let's start with a hypothetical example.

2.1 A Hypothetical Example

Pretend you are managing a small nature reserve and you notice that a new invasive species, spotted mallwort (*Capitalia globifera*),[2] is popping up everywhere. You think you may need to institute some control measures, and to begin, you would like to understand its life cycle and population growth.

Careful examination of the flowers reveals perfect flowers,[3] and you find from the literature that mallwort is a perennial capable of self-fertilizing. The seeds germinate in early fall, grow the following spring and early summer to a small adult that has 2–3 leaves and which sometimes produce seeds. In the second year and beyond, if the plants survive, they grow to large adults which have four or more leaves and only then do they produce the majority of their seeds.

[1] In his chapter entitled "Interesting Ways to Think about Death" G.E. Hutchinson [84] cites C. F. Trenerry, E. L. Gover and A. S. Paul (*The Origins and Early History of Insurance*, London, P. S. King & Sons, Ltd.) for description of early Roman actuarial tables.

[2] Not a real species.

[3] Individual flowers possess both female and male reproductive structures.

The seeds do not seem to survive more than one year, so there is probably no seed bank.

You summarize what you have learned in the form of a *life cycle graph* (Fig. 2.1). Demographers use a life cycle graph to summarize stages that may be observed at a single repeated point in time (e.g., when you go out to explore in June). It also can include the probabilities that an individual makes the transition from one stage to another over one time step (e.g. one year), as well as the fecundities.

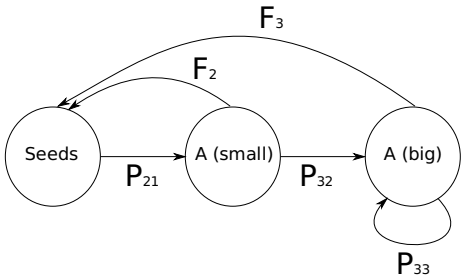


Fig. 2.2: Life cycle graph of the imaginary spotted mallwort (*Capitalia globifera*). P_{ij} is the probability that an individual in stage j transitions to stage i over a single fixed time interval between samples. F_i the number of progeny (transitioning into stage 1) produced by an individual in stage j. Thus, for mallwort, P_{21} is the probability that a seed (Seeds) makes it into the small adult stage (A-Small). P_{32} is the probability that a small adult shows up as a large adult the next year. F_2 is the average fertility for individuals in the small adult stage and F_3 is the average fertility for individuals in the large adult stage.

As the manager responsible for this small reserve, you decide to keep track of this new exotic species. After identifying a general area where the plant seems to have obtained a foothold, you established 50 permanent $1\,\mathrm{m}^2$ sample plots, located randomly throughout the invasion area. Each year, for two years, you sample in early summer when the fruits are on the plants (when the weather is pleasant and you can find interns and volunteers to help). In all plots you tag and count all first year plants (2–3 leaves), and all older plants (4+ leaves). You also are able to count fruits and have determined that there is one seed per fruit.

Now that you have your data for two years, you would like to figure out how quickly the population growing. You could simply keep track of the total population size, N, or just the large adults. You realize, however, that different stages may contribute very differently to growth, and different stages may be better for focused control efforts. A description, or model, of the population that includes different stages will provide this. We call such a model a *demographic model*, and it consists of a *population projection matrix*. The population projection matrix is a matrix that represents the life cycle graph.

We use the projection matrix to calculate all kinds of fun and useful stuff including

- The finite rate of increase, λ (the asymptotic population growth rate).
- The stable stage distribution (the population structure that would emerge if the demographic rates (P, F) do not change).
- Elasticty, the relative importance of each transition to λ.

2.1.1 The population projection matrix

The population projection matrix (a.k.a. the transition matrix) is simply an organized collection of the per capita contribution of each stage j to the next stage i in the specified time interval (often one year). These contributions, or transitions, consist of (i) the probabilities that an individual in stage j in one year makes it into stage i the next year, and (ii) the per capita fecundities for reproductive stages (eq. 2.1).

Each element of the projection matrix (eq. 2.1) relates its column to its row. Thus P_{21} in our matrix, eq. 2.1 is the probability that an individual in stage 1 (seeds; respresented by the column 1) makes it to the next census period and shows up in stage 2 (1 year old small adult; represented by row 2). Similarly, P_{32} is the probability that an individual in stage 2 (a small one year old adult) has made it to the large adult stage at the next census period. The fecundities are not probabilities, of course, but are the per capita contribution of propagules from the adult stage to the seed stage. The population projection matrix allows us to multiply all of these transition elements by the abundances in each age class in one year to predict, or *project*, the abundances of all age classes in the following year.

$$\begin{pmatrix} 0 & F_2 & F_3 \\ P_{21} & 0 & 0 \\ 0 & P_{32} & P_{33} \end{pmatrix} \tag{2.1}$$

2.1.2 A brief primer on matrices

We refer to matrices by their rows and columns. A matrix with three rows and one column is a 3×1 matrix (a "three by one" matrix); we *always* state the number of rows first. Matrices are composed of *elements*; an element of a matrix is signified by its row and column. The element in the second row and first column is a_{21}.

To multiply matrices, we multiply and then sum each row by each column (eq. B.3). More specifically, we multiply each row element of matrix **A** times each column element of matrix **B**, sum them, and place this sum in the respective element of the final matrix. Consider the matrix multiplication in eq. B.3. We first multiply each element of row 1 of **A** ($a \quad b$), times the corresponding elements of column 1 of **B** ($m \quad n$), sum these products and place the sum in the first row, first column of the resulting matrix. We then repeat this for each row of **A** and each column of **B**

$$\mathbf{A} = \begin{pmatrix} a & b \\ c & d \end{pmatrix}; \ \mathbf{B} = \begin{pmatrix} m & o \\ n & p \end{pmatrix} \tag{2.2}$$

$$\mathbf{AB} = \begin{pmatrix} (am + bn) & (ao + bp) \\ (cm + dn) & (co + dp) \end{pmatrix} \tag{2.3}$$

This requires that the number of columns in the first matrix must be the same as the number of rows in the second matrix. It also means that the resulting matrix will have the same number of rows as the first matrix, and the same number of columns as the second matrix.

Matrices in R

Let's define two 2×2 matrices, filling in one by rows, and the other by columns.

```
> M <- matrix(1:4, nr = 2, byrow = T)
> M

     [,1] [,2]
[1,]    1    2
[2,]    3    4

> N <- matrix(c(10, 20, 30, 40), nr = 2)
> N

     [,1] [,2]
[1,]   10   30
[2,]   20   40
```

Following our rules above, we would multiply and then sum the first row of M by the first column of N, and make this element a_{11} of the resulting matrix product.

```
> 1 * 10 + 2 * 20

[1] 50
```

We multiply matrices using **%*%** to signify that we mean *matrix* multiplication.

```
> M %*% N

     [,1] [,2]
[1,]   50  110
[2,]  110  250
```

2.1.3 Population projection

With our spotted mallwort we could multiply our projection matrix by the observed abundances (seeds=Sd, small adults - SA, large adults - LA) to *project* the abundances of all age classes in subsequent years.

$$\begin{pmatrix} 0 & F_2 & F_3 \\ P_{21} & 0 & 0 \\ 0 & P_{32} & P_{33} \end{pmatrix} \begin{pmatrix} N_{Sd} \\ N_{SA} \\ N_{LA} \end{pmatrix} = \begin{pmatrix} (0 \times N_{Sd} + F_2 \times N_{SA} + F_3 \times N_{LA}) \\ (P_{21} \times N_{Sd} + 0 \times N_{SA} + 0 \times N_{LA}) \\ (0 \times N_{Sd} + P_{32} \times N_{SA} + 0 \times N_{LA}) \end{pmatrix} \tag{2.4}$$

The next step is to create the projection matrix. Let's pretend that over the two years of collecting these data, you found that of the small adults we tagged, about half (50%) survived to become large adults the following year. This means that the transition from stage 2 (small adults) to stage 3 (large adults) is $P_{32} = 0.50$. Of the large adults that we tagged, about 90% of those survived to the next year, thus $P_{33} = 0.90$. We estimated that, on average, each small adult produces 0.5 seeds (i.e. $F_2 = 0.50$) and each large adult produces 20 seeds (i.e. $F_3 = 20$). Last, we found that, on average, for every 100 seeds (fruits) we counted, we found about 30 small adults (one year olds), meaning that $P_{21} = 0.30$. Note that this requires that seeds survive until germination, germinate, and then survive until we census them the following summer. We can now fill in our population projection matrix, **A**.

$$\mathbf{A} = \begin{pmatrix} 0 & F_2 & F_3 \\ P_{21} & 0 & 0 \\ 0 & P_{32} & P_{33} \end{pmatrix} = \begin{pmatrix} 0 & 0.5 & 20 \\ 0.30 & 0 & 0 \\ 0 & 0.50 & 0.90 \end{pmatrix} \tag{2.5}$$

Next we can multiply it the projection matrix, **A**, by the last year for which we have data.

$$\begin{pmatrix} 0 & 0.5 & 20 \\ 0.3 & 0 & 0 \\ 0 & 0.5 & 0.9 \end{pmatrix} \begin{pmatrix} 100 \\ 250 \\ 50 \end{pmatrix} = \begin{pmatrix} (0 \times 100 + 0.5 \times 250 + 20 \times 50) \\ (0.3 \times 100 + 0 \times 250 + 0 \times 50) \\ (0 \times 100 + 0.5 \times 250 + 0.9 \times 50) \end{pmatrix} = \begin{pmatrix} 1125 \\ 30 \\ 170 \end{pmatrix} \tag{2.6}$$

If we wanted more years, we could continue to multiply the projection matrix by each year's projected population. We will observe that, at first, each stage increases or decreases in its own fashion (Fig. 2.3a), and that over time, they tend to increase in a more similar fashion. This is typical for demographic models. It is one reason why it is important to examine *stage-structured growth* rather than trying to lump all the stages together — we have a much richer description of how the population is changing.

Stage structured growth - one step

First, we create a population projection matrix, and a vector of stage class abundances for year zero.

```
> A <- matrix(c(0, 0.5, 20, 0.3, 0, 0, 0, 0.5, 0.9), nr = 3,
+     byrow = TRUE)
> N0 <- matrix(c(100, 250, 50), ncol = 1)
```

Now we perform matrix multiplication between the projection matrix and N_0.

```
> N1 <- A %*% N0
> N1

     [,1]
[1,] 1125
[2,]   30
[3,]  170
```

Note that the first stage declined, while the second and third stages increased.

Stage structured growth - multiple steps

Now we project our population over six years, using a for-loop. We use a for-loop, rather than `sapply`, because each year depends on the previous year (see the Appendix, sec. B.6). First, we set the number of years we want to project, and then create a matrix to hold the results. We put N_0 in the first column.

```
> years <- 6
> N.projections <- matrix(0, nrow = nrow(A), ncol = years +
+     1)
> N.projections[, 1] <- N0
```

Now we perform the iteration with the for-loop.

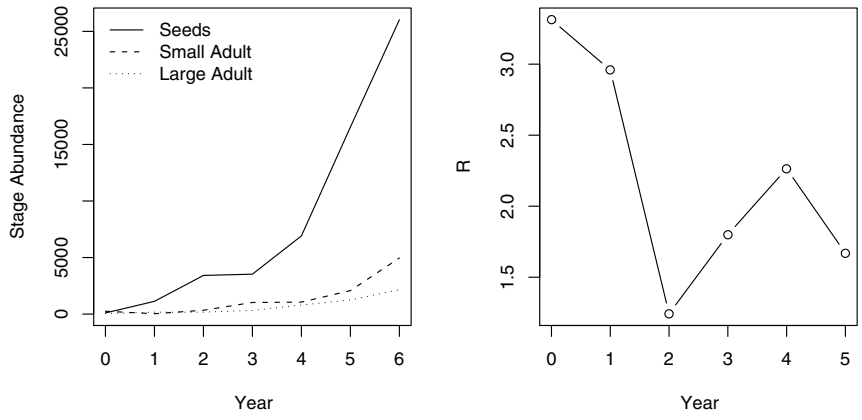
```
> for (i in 1:years) N.projections[, i + 1] <- A %*% N.projections[,
+     i]
```

Last, we graph the results for each stage (Fig. 2.3a). To graph a matrix, R is expecting that the data will be in columns, not rows, and so we need to *t*ranspose the projection matrix.

```
> matplot(0:years, t(N.projections), type = "l", lty = 1:3,
+     col = 1, ylab = "Stage Abundance", xlab = "Year")
> legend("topleft", legend = c("Seeds", "Small Adult", "Large Adult"),
+     lty = 1:3, col = 1, bty = "n")
```

2.1.4 Population growth

We have projected the stages for six years — what is its observed rate of increase, $R_t = N_{t+1}/N_t$? How do we even think about R and N in stage structured growth? The way we think about and calculate these is to add all the individuals in all stages to get a total N, and calculate R with that, as we did in Chapter 1.



(a) Projected Population (b) Annual Growth

Fig. 2.3: Population dynamics and annual growth ($R = N_{t+1}/N_t$) of spotted mallwort. Note that stage abundance is on a log-scale.

$$R_t = N_{t+1}/N_t. \tag{2.7}$$

If we do that for our mallwort, we can see that R_t changes with time (Fig. 2.3b). We can summarize the projection as $\mathbf{n}(t) = \mathbf{A}^t\mathbf{n}_0$, where $\mathbf{A}^t$ is $\mathbf{A}$ multiplied by itself t times.

Annual growth rate

Now let's calculate $R_t = N_{t+1}/N_t$ for each year t. We first need to sum all the stages, by *applying* the `sum` function to each column.

```
> N.totals <- apply(N.projections, 2, sum)
```

Now we get each R_t by dividing all the N_{t+1} by each N_t. Using negative indices cause R to drop that element.

```
> Rs <- N.totals[-1]/N.totals[-(years + 1)]
```

We have one fewer Rs than we do years, so let's plot each R in each year t, rather than each year $t + 1$ (Fig. 2.3b).

```
> plot(0:(years - 1), Rs, type = "b", xlab = "Year", ylab = "R")
```

2.2 Analyzing the Projection Matrix

You seem to have a problem on your hands (Fig. 2.3a). Being a well-trained scientist and resource manager, several questions come to mind: What the heck do I do now? What is this population likely to do in the future? Can these

data provide insight into a control strategy? How confident can I be in these projections?

After you get over the shock, you do a little more research on demographic models; Caswell [23] is the definitive treatise. You find that, indeed, there is a lot you can do get more information about this population that might be helpful.

Once you have obtained the projection matrix, **A**, you can analysis it using *eigenanalysis* to estimate

- λ, the finite rate of increase,
- stable stage structure,
- reproductive value, and
- sensitivities and elasticities.

Below, we explain each of these quantities. These quantities will help you determine which stages of spotted mallwort on which to focus eradication efforts.

2.2.1 Eigenanalysis

Eigenanalysis is a mathematical technique that summarizes multivariate data. Ecologists use eigenanalysis frequently, for (i) multivariate statistics such as ordination, (ii) stability analyses with two or more species, and (iii) analyzing population projection matrices. Eigenanalysis is simply a method to transform a square matrix into independent, orthogonal, and useful chunks — the eigenvectors and their eigenvalues. In demography, the most useful piece is the dominant eigenvalue and its corresponding vector.

Eigenanalysis is a technique that finds all the solutions for λ and $\mathbf{w}$ of

$$\mathbf{Aw} = \lambda\mathbf{w}, \tag{2.8}$$

where **A** is a particular summary of our data. [4] With projection matrix analysis, **A** is the projection matrix. λ is an *eigenvalue* and $\mathbf{w}$ is an *eigenvector*. If we write out eq. 2.8 for a 3×3 matrix, we would have

$$\begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{33} \\ a_{31} & a_{32} & a_{33} \end{pmatrix} \begin{pmatrix} w_{11} \\ w_{21} \\ w_{31} \end{pmatrix} = \lambda \begin{pmatrix} w_{11} \\ w_{21} \\ w_{31} \end{pmatrix} \tag{2.9}$$

There are typically an infinite number of solutions to this equation, and what eigenanalysis does is find set of solutions that are all independent of each other, and which capture all of the information in **A** in a particularly useful way.[5] Typically, the first solution captures the most important features of the

[4] For ordination, we analyze a correlation or covariance matrix, and for stability analyses, we use the matrix of pairwise partial differential equations between each pair of species. In these eigenanalyses of a square $i \times j$ matrix **A**, we can think of the elements of **A** describing the "effect" of stage (or species) j on stage (or species) i, where j is a column and i is a row.

[5] The number of solutions is infinite because they are just simple multiples of the set found with eigenanalysis.

projection matrix. We call this the dominant eigenvalue, λ_1 and its corresponding eigenvector, w_1. The first solution does not capture all of the information; the second solution captures much of the important remaining information. To capture all of the information in **A** requires as many solutions as there are columns of **A**. Nonetheless, the first solution is usually the most useful.

Eigenanalysis in R

Here we perform eigenanalysis on **A**.

```
> eigs.A <- eigen(A)
> eigs.A

$values
[1]  1.834+0.000i -0.467+1.159i -0.467-1.159i

$vectors
           [,1]               [,2]               [,3]
[1,] 0.98321+0i  0.97033+0.00000i  0.97033+0.00000i
[2,] 0.16085+0i -0.08699-0.21603i -0.08699+0.21603i
[3,] 0.08613+0i -0.02048+0.06165i -0.02048-0.06165i
```

Each eigenvalue and its corresponding eigenvector provides a solution to eq. 2.8.

The first, or dominant, eigenvalue is the long term asymptotic finite rate of increase λ. Its corresponding eigenvector provides the *stable stage distribution.*

We can also use eigenanalysis get the *reproductive values* of each stage out of **A**. To be a little more specific, **w** we described above are *right* eigenvectors, so-called because we solve for them with **w** on the right side of **A**. We will also generate *left* eigenvectors **v** (and their corresponding eigenvalues), where $\mathbf{vA} = \lambda(v)$. The dominant left eigenvector provides the reproductive values (section 2.2.4).

2.2.2 Finite rate of increase – λ

The asymptotic annual growth rate finite rate of increase is the dominant *eigenvalue* of the projection matrix. Eigenvalues are always referred to with the Greek symbol λ, and provides a solution to eq. (2.8). The dominant eigenvalue of any matrix, λ_1, is the eigenvalue with the largest absolute value, and it is frequently a complex number.[6] With projection matrices, λ_1 will always be positive and real.

We use eigenanalysis to solve eq. 2.8 and give us the answers — like magic. Another way to find λ_1 is to simply iterate population growth a very large number of times, that is, let t be very large. As t grows, the annual growth rate, N_{t+1}/N_t, approaches λ_1 (Fig. 2.4).

[6] When you perform eigenanalysis, it is common to get complex numbers, with real and imaginary parts. Eigenanalysis is, essentially, solving for the roots of the matrix, and, just like when you solved quadratic equations by hand in high school, it is possible to get complex numbers.

Finding λ

Next we explicitly find the index position of the largest absolute value of the eigenvalues. In most cases, it is the first eigenvalue.

```
> dom.pos <- which.max(eigs.A[["values"]])
```

We use that index to extract the largest eigenvalue. We keep the real part, using `Re`, dropping the imaginary part. (Note that although the dominant eigenvalue will be real, R will include an imaginary part equal to zero ($0i$) as a place holder if *any* of the eigenvalues have a non-zero imaginary part).

```
> L1 <- Re(eigs.A[["values"]][dom.pos])
> L1

[1] 1.834
```

`L1` is λ_1, the aysmptotic finite rate of increase.

Power iteration method of eigenanalysis

Because growth is an exponential process, we can figure out what is most important in a projection matrix by multiplying it by the stage structure many times. This is actually one way of performing eigenanalysis, and it is called the *power iteration method*. It is not terribly efficient, but it works well in some specific applications. (This method is *not* used by modern computational languages such as R.) The population size will grow toward infinity, or shrink toward zero, so we keep rescaling N, dividing the stages by the total N, just to keep things manageable.
Let t be big, and rescale N.

```
> t <- 20
> Nt <- N0/sum(N0)
```

We then create a for-loop that re-uses N_t for each time step, making sure we have an empty numeric vector to hold the output.

```
> R.t <- numeric(t)
> for (i in 1:t) R.t[i] <- {
+     Nt1 <- A %*% Nt
+     R <- sum(Nt1)/sum(Nt)
+     Nt <- Nt1/sum(Nt1)
+     R
+ }
```

Let's compare the result directly to the point estimate of λ_1 (Fig. 2.4).

```
> par(mar = c(5, 4, 3, 2))
> plot(1:t, R.t, type = "b", main = quote("Convergence Toward  " *
+     lambda))
> points(t, L1, pch = 19, cex = 1.5)
```

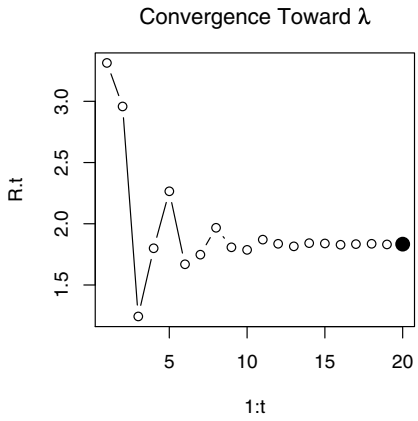


Fig. 2.4: Iterating the population, and recalculating $R_t = N_{t+1}/N_t$ at each time step converges eventually at the dominant eigenvalue, indicated as a solid point. It is possible to use the same power iteration method to get the other eigenvalues, but it is not worth the trouble.

2.2.3 Stable stage distribution

The relative abundance of the different life history stages is called the *stage distribution*, that is, the distribution of individuals among the stages. A property of a stage structured population is that, if all the demographic rates (elements of the population projection matrix) remain constant, its stage structure will approach a *stable stage distribution*, a stage distribution in which the **relative** number of individuals in each stage is constant. Note that a population can grow, so that the absolute number of individuals increases, but the relative abundances of the stages is constant; this is the stable stage distribution. If the population is not actually growing ($\lambda = 1$) and demographic parameters remain constant, then the population is *stationary* and will achieve a *stationary stage distribution*, where neither absolute nor relative abundances change.

How do we find the stable stage distribution? It also turns out that w_1, which is the corresponding eigenvector of λ_1 (eq. (2.8)), provides the necessary information. We scale the eigenvector w_1 by the sum of its elements because we are interested in the *distribution*, where all the stages should sum to one. [7] Therefore the stable stage distribution is

$$SSD = \frac{w_1}{\sum_{i=1}^{S} w_1} \tag{2.10}$$

where S is the number of stages.

Once a population reaches its stable stage distribution it grows exponentially,

[7] Eigenvectors can only be specified up to a constant, arbitrary multiplier.

$$\mathbf{N_t} = \mathbf{A^t N_0}$$
$$N_t = \lambda^t N_0$$

represented either in the matrix notation (for all stages), or simple scalar notation (for total N only).

Calculating the stable stage distribution

The dominant eigenvector, **w**, is in the same position as the dominant eigenvalue. We extract **w**, keeping just the real part, and divide it by its sum to get the stable stage distribution.

```
> w <- Re(eigs.A[["vectors"]][, dom.pos])
> ssd <- w/sum(w)
> round(ssd, 3)

[1] 0.799 0.131 0.070
```

This shows us that if the projection matrix does not change over time, the population will eventually be composed of 80% seeds, 13% small adults, and 7% large adults. Iterating the population projection will also eventually provide the stable stage distribution (e.g., Fig. 2.3a).

2.2.4 Reproductive value

If the stage structure gives us one measure of the importance of a stage (its abundance), then the *reproductive value* gives us one measure of the importance of an *individual* in each stage. Reproductive value is the expected contribution of each individual to present and future reproduction. We characterize all individuals in a stage using the same expected reproductive value.

We find each stage's reproductive value by solving for the dominant *left* eigenvector **v**, where

$$\mathbf{vA} = \lambda \mathbf{v} \tag{2.11}$$

Like the relation between the dominant right eigenvector and the stable stage distribution, this vector is actually *proportional* to the reproductive values. We typically scale it for $v_0 = 1$, so that all reproductive values are relative to that of the first stage class (e.g. newborns or seeds).

$$RV = \frac{v_1}{\sum_{i=1}^{S} v_1} \tag{2.12}$$

Calculating reproductive value

We get the left eigenvalues and -vectors by performing eigenanalysis on the transpose of the projection matrix. The positions of the dominant right and left eigenvalues are the same, and typically they are the first. We perform eigenanalysis, extracting just the the dominant left eigenvector; we then scale it, so the stage 1 has a reproductive value of 1.0.

```
> M <- eigen(t(A))
> v <- Re(M$vectors[, which.max(Re(M$values))])
> RV <- v/v[1]
> RV

[1]  1.000  6.113 21.418
```

Here we see a common pattern, that reproductive value, v, increases with age. In general, reproductive value of individuals in a stage increases with increasing probability of reaching fecund stages.

2.2.5 Sensitivity and elasticity

Sensitivity and elasticity tell us the relative importance of each transition (i.e. each arrow of the life cycle graph or element of the matrix) in determining λ. They do so by combining information on the stable stage structure and reproductive values.

The stage structure and reproductive values each in their own way contribute to the importance of each stage in determining λ. The stable stage distribution provides the relative abundance of individuals in each stage. Reproductive value provides the contribution to future population growth of individuals in each stage. Sensitivity and elasticity combine these to tell us the relative importance of each transition in determining λ.

Sensitivities of a population projection matrix are the direct contributions of each transition to determining λ. We would say, speaking in more mathematical terms, that the sensitivities for the elements a_{ij} of a projection matrix are the changes in λ, given small changes in each element, or $\delta\lambda/\delta a_{ij}$. Not surprisingly, then, these are derived from the stable stage distribution and the reproductive values. Specifically, the sensitivities are calculated as

$$\frac{\delta\lambda}{\delta a_{ij}} = \frac{v_{ij}w_{ij}}{\mathbf{v}\cdot\mathbf{w}} \tag{2.13}$$

where $v_i w_j$ is the product of each pairwise combination of elements of the dominant left and right eigenvectors, $\mathbf{v}$ and $\mathbf{w}$. The *dot product*, $\mathbf{v}\cdot\mathbf{w}$, is the sum of the pairwise products of each vector element. Dividing by this causes the sensitivities to be relative to the magnitudes of $\mathbf{v}$ and $\mathbf{w}$.

Sensitivity of projection matrices

Let's calculate sensitivities now. First we calculate the numerator for eq. 2.13.

```
> vw.s <- v %*% t(w)
```

Now we sum these to get the denominator, and then divide to get the sensitivities. (The dot product $\mathbf{v} \cdot \mathbf{w}$ yields a 1×1 matrix; in order to divide by this quantity, the simplest thing is to cause the dot product to be a simple scalar rather than a matrix (using `as.numeric`), and then R will multiply each element.)

```
> (S <- vw.s/as.numeric(v %*% w))
```

```
      [,1]    [,2]   [,3]
[1,] 0.258 0.04221 0.0226
[2,] 1.577 0.25798 0.1381
[3,] 5.526 0.90396 0.4840
```

We see from this that the most important transition exhibited by the plant is s_{21}, surviving from the seed stage to the second stage (the element s_{31} is larger, but is not a transition that the plant undergoes).

Elasticities are sensitivities, weighted by the transition probabilities. Sensitivities are large when reproductive value and or the stable age distribution are high, and this makes sense biologically because these factors contribute a lot to λ. We may, however, be interested in how a *proportional* change in a transition element influences lambda—how does a 10% increase in seed production, or a 25% decline in juvenile survival influence λ? For these answers, we need to adjust sensitivities to account for the relative magnitudes of the transition elements, and this provides the elasticities, e_{ij}, where

$$e_{ij} = \frac{a_{ij}}{\lambda} \frac{\delta \lambda}{\delta a_{ij}}. \tag{2.14}$$

Elasticity of projection matrices

In R, this is also easy.

```
> elas <- (A/L1) * S
> round(elas, 3)
```

```
      [,1]  [,2]  [,3]
[1,] 0.000 0.012 0.246
[2,] 0.258 0.000 0.000
[3,] 0.000 0.246 0.238
```

Note that all the elasticities except the seed production by small adults appear equally important. Specifically, the same proportional change in any of these elements will result in approximately the same change in λ.

There are two nice features of elasticities. First, *impossible transitions have elasticities equal to zero*, because we multiply by the projection matrix itself.

Second, the *elasticities sum to zero*, and so it is easier to compare elasticities among differ matrices and different organisms.

Once we have the sensitivities and elasticities, we can really begin to see what is controlling the growth rate of a stage (or age) structured population. Although these values do not tell us which stages and transitions will, in reality, be influenced by natural phenomona or management practices, they provide us with the predicted *effects* on λ of a proportional change in a demographic rate, P or F. This is particularly important in the management of invasive (or endangered) species where we seek to have the maximum impact for the minimum amount of effort and resources [23, 48].

2.2.6 More demographic model details

Births

For demographic models, a "birth" is merely the appearance in the first stage. If we census birds, a "birth" might be a fledging, if this is the youngest age class we sampled. If we census plants, we might choose to count seeds as the first age class, or we might use seedling, or some size threshold as the first stage. Regardless of the first stage- or age-class we use, a birth is the first appearance of an individual in the first stage.

Pre- vs. post-breeding census

Note that you are sampling the population of mallwort at a particular time of year. This sampling happens to be a *postbreeding census* because you captured everything right after breeding, when progeny were observed directly. The projection matrix would look different, and the interpretation of the matrix elements would differ, if we had used a *prebreeding census*, sampling the population before breeding. In particular, the projection matrix would have only two stages (small and large adults), because no seeds would be present at the time of sampling. The contribution of adults to the youngest stage, therefore, would represent both fertility and survival to the juvenile stage in late spring. Nonetheless, both models would be equivalent, generating the same λ.

Birth pulse vs. birth flow model

Another assumption we are making is that individuals set seed, or give birth, all at once. We refer to the relevant model as a *birth-pulse model*. On the other hand, if we assume that we have continuous reproduction, we do things quite differently, and would refer to this as a *birth-flow model*. Whether a population is breeding continuously over a year, or whether reproduction is seasonal, will influence how we estimate fecundities. Even for synchronously breeding populations, many models pool years into a single age class or stage. As result, we need to be careful about how we approximate probabilities that will differ among individuals within the age- or stage-class.

These details can get very confusing, and smart people don't always get it right. Therefore, consult an expert [23, 48], and remember that the stages of life cycle graph and matrix are the stages that you collect at one point in time.

2.3 Confronting Demographic Models with Data

This section uses R extensively throughout.

It is common to create a demographic matrix model with real data, and then use that model for an applied purpose (e.g., [44, 50]). A central question, however, is just how confident we can be in our model, and the values we derive from it. It turns out that we can use our data to derive confidence intervals on important parameters.

In Chapter 1, we used resampling to draw observed annual changes in bird counts at random to generate growth trajectories and confidence intervals on population size. Here we resample raw data to find confidence limits on λ. The method used here, *bootstrapping*, and related data-based inference techniques have a large literature [126]. Davison and Hinkley [46] have an comprehensive R-based text. Such randomization methods are very useful for a wide range of models in ecology, where the data do not conform clearly to parametric distributions or to situations like demographic models [140] or null models [60] for which analytical approximations are difficult or not possible.

The basic idea of bootstrapping is to

1. calculate the *observed parameter(s)* of interest (e.g., a mean, or λ) with your model and data,
2. resample your data *with replacement* to create a large number of datasets and recalculate your parameter(s) for each resampled dataset to generate a *distribution* of the *bootstrapped*[8] parameter(s),
3. Calculate a confidence interval for the bootstrapped parameter values — this will provide an estimate of the confidence you have in your observed parameter. This will provide an *empirical confidence interval*.

2.3.1 An Example: *Chamaedorea* palm demography

Chamaedorea radicalis Mart. (Arecaceae) is an forest understory palm of northern Mexico, and it is one of approximately 100 *Chamaedorea* species, many of which are economically valuable as either small, shade-tolerant potting plants or as harvested leaves in floral arrangements. Its demography is interesting for a number of reasons, including both management and as an example of a population that appears to be maintained through source-sink dynamics [12]. Berry *et al.* modeled *Chamaedorea* demography with five stages (Fig. 2.5). Demography is also influenced by substrate type, by livestock browsing, and harvesting [12]. Here we use a subset of the data to illustrate the generation of demographic parameters and confidence intervals.

This study was conducted in the montane mesophyll forests of Sierra de Guatemala mountain range, near the communities of San José and Alta Cimas within the El Cielo Biosphere Reserve, Tamaulipas, Mexico (22°55'–23°30'N and 99°02'–99°30'W). Villagers within El Cielo (palmilleros) harvest adult *C.*

[8] "Bootstrapped" estimates are thus named because you are picking yourself up by your own bootstraps – a seemingly impossible task.

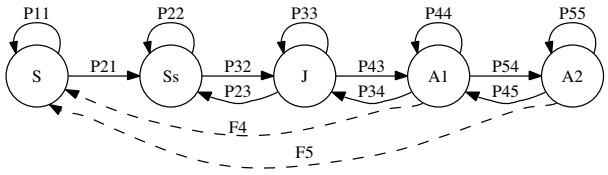


Fig. 2.5: Life cycle graph for *Chamaedorea radicalis*. Classification criteria are based on the number of leaflets on the youngest fully-expanded leaf. Life-history stage transitions are indicated by arrows with solid lines and reproduction is indicated by dashed lines. Abbreviations: S-seed, Ss-seedling (bifid leaves), J-Juvenile (3–9 leaflets), A1-small adult (10–24 leaflets), A2-large adult (> 24 leaflets). Source: [12, 50]

radicalis leaves for sale to international cut-foliage markets. Harvested leaves are usually >= 40 cm in length, and have minimal damage from insects or pathogens [50]. These palm leaves are the only natural resource that these villagers are authorized to harvest, and provide the main source of income for most families. Although *C. radicalis* is dioecious (more complications!), Berry et al. [12] used a one sex model, because its simplifying assumptions were well supported with data. Data collected allowed a postbreeding census model with a birth-pulse dynamic.

2.3.2 Strategy

There are an infinite number of ways to do anything in R, and I am certain that my approach to this bootstrapping is not the very best way, but it is *useful*. It gives valid answers in a reasonable amount of time, and that is what we want from a model.

This is how we proceed in this instance:

1. We import the data and look at it. The appearance of the data, how the data are entered for instance, will influence subsequent decisions about how to proceed.
2. We extract the relevant data and calculate the projection matrix elements (fecundities and transition probabilities). We first do it all piecewise, to figure out what we are doing. Then we can wrap it up inside a function putting `funcname <- function(data1, data2, data3)` at the beginning and collecting and returning all relevant parameters at the end (see sec. B.4.1 for writing functions).
3. We also create a function to generate all the demographic parameters that we will eventually want (λ, elasticities, etc.).
4. Last, we combine these two functions into one that also resamples the original data (with replacement), and then calls the data extraction and calculation functions to generate the new parameters for the bootstrapped data.
5. The bootstrapping is repeated B times.
6. Having generated B bootstrapped estimates of all the parameters, we can then calculate confidence intervals for any parameter that we like.

2.3.3 Preliminary data management

Let's import the data and have a look at it. For these purposes, we will assume that the data are clean and correct. Obviously, if I were doing this for the first time, data-checking and clean-up would be an important first step. Here we simply load them from the `primer` package.

```
> data(stagedat)
> data(fruitdat)
> data(seeddat)
```

Now I look at the structure of the data to make sure it is at least approximately what I think it is.

```
> str(stagedat)
```

```
'data.frame':        414 obs. of  4 variables:
 $ PalmNo: int  1 2 3 4 5 6 7 8 9 10 ...
 $ Y2003 : int  4 5 5 4 3 2 4 3 3 4 ...
 $ Y2004 : int  5 4 5 5 4 3 5 3 4 4 ...
 $ Y2005 : int  5 5 5 5 4 3 5 4 4 5 ...
```

The stage data provide the stage of each individual in the study. Each row is an individual, and its ID number is in column 1. Data in columns 2–4 identify its stage in years 2003–2005.

We can count, or tabulate, the number of individuals in each stage in 2004.

```
> table(stagedat[["Y2004"]])
```

```
  0   2   3   4   5
 17  58  48 126 165
```

We see, for instance, that in 2004 there were 165 individuals in stage 5. We also see that 17 individuals were dead in 2004 (stage = 0); these were alive in either 2003 or 2005.

The fruit data have a different structure. Each row simply identifies the stage of each individual (col 1) and its fertility (number of seeds) for 2004.

```
> str(fruitdat)
```

```
'data.frame':        68 obs. of  2 variables:
 $ Stage: int  4 4 4 4 4 4 4 4 4 4 ...
 $ Y2004: int  6 0 0 0 0 0 0 0 0 0 ...
```

We can tabulate the numbers of seeds (columns) of each stage (rows).

```
> table(fruitdat[["Stage"]], fruitdat[["Y2004"]])
```

```
     0  1  2  3  4  5  6  8 15 22 30 37 70 98 107 109
  4 28  0  0  0  0  0  1  0  0  0  0  0  0  0   0   0
  5 23  1  1  1  2  2  0  1  1  1  1  1  1  1   1   1
```

For instance, of the individuals in stage 4 (row 1), 28 individuals had no seeds, and one inidvidual had 6 seeds. Note also that only stage 4 and 5 had plants with *any* seeds.

The seed data are the fates of each seed in a sample of 400 seeds, in a data frame with only one column.

```
> table(seeddat)
```

```
seeddat
  0   1   2
332  11  57
```

Seeds may have germinated (2), remained viable (1), or died (0).

2.3.4 Estimating projection matrix

Now we work through the steps to create the projection matrix from individuals tagged in year 2003 and re-censused in 2004. If we convert the life cycle graph (Fig. 2.5) into a transition matrix.

$$\begin{pmatrix} P_{11} & 0 & 0 & F_4 & F_5 \\ P_{21} & P_{22} & P_{23} & 0 & 0 \\ 0 & P_{32} & P_{33} & P_{34} & 0 \\ 0 & 0 & P_{43} & P_{44} & P_{45} \\ 0 & 0 & 0 & P_{54} & P_{55} \end{pmatrix} \tag{2.15}$$

Along the major diagonal (where $i = j$) the P_{ij} represent the probability that a palm stays in the same stage. In the lower off-diagonal ($i > j$) the P_{ij} represent the probability of growth, that an individual grows from stage j into stage i. In the upper off-diagonal ($i < j$) the P_{ij} represent the probability of regression, that an individual regresses from stage j back into stage i. The F_i represent the fertility of stage i.

As a practical matter, we will use basic data manipulation in R to transform the raw datat into transition elements. We had no particular reason for having the data in this form, this is simply how the data were available.

We first create a zero matrix that we will then fill.

```
> mat1 <- matrix(0, nrow = 5, ncol = 5)
```

Fertilities

For each stage, we get mean fertility by applying `mean` to each stage of the 2004 fertility data. Here `Stage` is a factor and `tapply` will caculate a mean for each level of the factor. We will assume that half of the seeds are male. Therefore, we divide fertility by 2 to calculate the fertility associated with just the female seeds.

```
> ferts <- tapply(fruitdat$Y2004, fruitdat$Stage, mean)/2
> ferts
```

```
     4      5
0.1034 6.6667
```

These fertilities, F_4 and F_5, are the transitions from stages 4 and 5 (adults) to stage 1 (seeds). Next we insert the fertilities (`ferts`) into the matrix we established above.

```
> mat1[1, 4] <- ferts[1]
> mat1[1, 5] <- ferts[2]
```

Seed transitions

Now we get the frequencies of each seed fate (die, remain viable but dormant, or germinate), and then divide these by the number of seeds tested (the length of the seed vector); this results in proportions and probabilities.

```
> seed.freqs <- table(seeddat[, 1])
> seedfates <- seed.freqs/length(seeddat[, 1])
> seedfates
```

```
     0      1      2
0.8300 0.0275 0.1425
```

The last of these values is P_{21}, the transition from the first stage (seeds) to the stage 2 (seedlings). The second value is the transition of seed dormancy ($P_{1,1}$), that is, the probability that a seed remains a viable seed rather than dying or becoming a seedling.

Next we insert the seed transitions into our projection matrix.

```
> mat1[1, 1] <- seedfates[2]
> mat1[2, 1] <- seedfates[3]
```

Vegetative stage transitions

Here we calculate the transition probabilities for the vegetative stages. The pair of for-loops will calculate these transitions and put them into stages 2–5.The functions inside the for-loops (a) subset the data for each stage in 2003, (b) count the total number of individuals in each stage in 2003 (year j), (c) sum the number of individuals in each stage in 2004, given each stage for 2003, and then (d) calculate the proportion of each stage in 2003 that shows up in each stage in 2004.

```
> for (i in 2:5) {
+     for (j in 2:5) mat1[i, j] <- {
+         x <- subset(stagedat, stagedat$Y2003 == j)
+         jT <- nrow(x)
+         iT <- sum(x$Y2004 == i)
+         iT/jT
+     }
+ }
```

Here we can see the key parts of a real projection matrix.

```
> round(mat1, 2)
```

```
     [,1] [,2] [,3] [,4] [,5]
[1,] 0.03 0.00 0.00 0.10 6.67
[2,] 0.14 0.70 0.05 0.01 0.00
[3,] 0.00 0.23 0.42 0.04 0.00
[4,] 0.00 0.00 0.46 0.67 0.07
[5,] 0.00 0.00 0.02 0.26 0.90
```

Compare these probabilities and fertilities to the life cycle graph and its matrix (Fig. 2.5, eq. (2.15)).

The diagonal elements $P_{j,j}$ are stasis probabilities, that an individual remains in that stage. Growth, from one stage to the next, is the lower off-diagonal, $P_{j+1,j}$. Regression, moving back one stage, is the upper off diagonal, $P_{j-1,j}$. The fertilities are in the top row, in columns 4 and 5. Note that there is a transition element in our data that is not in eq. (2.15): P_{53}. This corresponds to very rapid growth — a real event, albeit highly unusual.

A function for all transitions

What a lot of work! The beauty, of course, is that we can put all of those lines of code into a single function, called, for instance, `ProjMat`, and all we have to supply are the three data sets. You could examine this function by typing `ProjMat` on the command line, with no parentheses, to see the code and compare it to our code above. You code also try it with data.

```
> ProjMat(stagedat, fruitdat, seeddat)
```

This provides the observed transition matrix (results not shown).

2.3.5 Eigenanalyses

Next we want to do all those eigenanalyses and manipulations that gave us λ, the stable age distribution,reproductive value, and the sensitivity and elasticity matrices. All of this code is wrapped up in the function `DemoInfo`. Convince yourself it is the same code by typing `DemoInfo` with no parentheses at the prompt. Here we try it out on the projection matrix we created above, and examine the components of the output.

```
> str(DemoInfo(mat1))
```

```
List of 6
 $ lambda       : num 1.13
 $ SSD          : num [1:5] 0.5632 0.195 0.0685 0.0811 0.0922
 $ RV           : num [1:5] 1 7.76 14.37 20.18 33.95
 $ Sensitivities: num [1:5, 1:5] 0.072 0.559 1.034 1.452 2.442 ...
 $ Elasticities : num [1:5, 1:5] 0.00174 0.0702 0 0 0 ...
 $ PPM          : num [1:5, 1:5] 0.0275 0.1425 0 0 0 ...
```

We find that `DemoInfo` returns a *list* with six named *components*. The first component is a scalar, the second two are numeric vectors, and the last three are numeric matrices. The last of these is the projection matrix itself; it is often useful to return that to prove to ourselves that we analyzed the matrix we intended to.

2.3.6 Bootstrapping a demographic matrix

All of the above was incredibly useful and provides the best estimates of most or all the parameters we might want. However, it does not provide any idea of the certainty of those parameters. By bootstrapping these estimates by resampling our data, we get an idea of the uncertainty.

Here we work through the steps of resampling our data, as we build a function, step by step, inch by inch. The basic idea of resampling is that we assume that our sample data are the best available approximation of the entire population. Therefore, we draw, with replacement, new data sets from the original one. See the last section in Chapter 1 for ideas regarding simulations and bootstrapping.

We will create new resampled (bootstrapped) data sets, where the rows of the original data sets are selected at random with replacement. We then apply `ProjMat` and `DemoInfo`.

The first step is to get the number of observations in the original data.

```
> nL <- nrow(stagedat)
> nF <- nrow(fruitdat)
> nS <- nrow(seeddat)
```

With these numbers, we will be able to resample our original data sets getting the correct number of resampled observations.

Now we are going to use `lapply` to perform everything multiple times. By "everything," I mean

1. resample the observations to get bootstrapped data sets for vegetative stages, seed fates, and fertilities,
2. calculate the projection matrix based on the three bootstrapped data sets,
3. perform eigenanalysis and calculate λ, stage structure, sensitivities, and elasticities.

All of that is one replicate simulation, $n = 1$.

For now, let's say $n = 5$ times as a trial. Eventually this step is the one we will ask R to do 1000 or more times.

```
> n <- 5
```

Next we use `lapply` to do *everything*, that is, a replicate simulation, n times. It will store the n replicates in a *list*, n *components* long. Each of the n components will be the output of `DemoInfo`, which is itself a list.

```
> n <- 5
> out <- lapply(1:n, function(i) {
+     stageR <- stagedat[sample(1:nL, nL, replace = TRUE),
```

```
+         ]
+     fruitR <- fruitdat[sample(1:nF, nF, replace = TRUE),
+         ]
+     seedR <- as.data.frame(seeddat[sample(1:nS, nS, replace = TRUE),
+         ])
+     matR <- ProjMat(stagedat = stageR, fruitdat = fruitR,
+         seeddat = seedR)
+     DemoInfo(matR)
+ })
```

This code above uses `sample` to draw row numbers at random and with replacement to create random draws of data (`stageR, fruitR, and seedR`). We then use `ProjMat` to generate the projection matrix with the random data, and use `DemoInfo` to perform all the eigenanalysis and demographic calculations.

Let's look at a small subset of this output, just the five λ generated from five different bootstrapped data sets. The object `out` is a list, so using `sapply` on it will do the same thing to each component of the list. In this case, that something is to merely extract the bootstrapped λ.

```
> sapply(out, function(x) x$lambda)
```

```
[1] 1.114 1.127 1.195 1.114 1.096
```

We see that we have five different estimates of λ, each the dominant eigenvalue of a projection matrix calculated from bootstrapped data.

We now have all the functions we need to analyze these demographic data. I have put all these together in a function called `DemoBoot`, whose arguments (inputs) are the raw data, and n, the number of bootstrapped samples.

```
> args(DemoBoot)
```

```
function (stagedat = NULL, fruitdat = NULL, seeddat = NULL, n = 1)
NULL
```

2.3.7 The demographic analysis

Now we are armed with everything we need, including estimates and means to evaluate uncertainty, and we can move on to the ecology. We first interpret point estimates of of demographic information, including λ and elasticities. Then we ask whether λ differs significantly from 1.0 using our bootstrapped confidence interval.

First, point estimates based on original data.

```
> estims <- DemoInfo(ProjMat(stagedat, fruitdat, seeddat))
> estims$lambda
```

```
[1] 1.134
```

Our estimate of λ is greater than one, so the population seems to be growing.

Which transitions seem to be the important ones?

```
> round(estims$Elasticities, 4)
```

```
       [,1]   [,2]   [,3]   [,4]   [,5]
[1,] 0.0017 0.0000 0.0000 0.0009 0.0693
[2,] 0.0702 0.1196 0.0030 0.0005 0.0000
[3,] 0.0000 0.0738 0.0470 0.0049 0.0000
[4,] 0.0000 0.0000 0.0712 0.1234 0.0145
[5,] 0.0000 0.0000 0.0044 0.0793 0.3162
```

It appears that the most important transition is persistence in the largest adult stage ($a_{5,5} = 0.3$). Specifically, proportional changes to the persistence in this stage, neither regressing nor dying, are predicted to have the largest postive effect on the lambda of this population.

We stated above that the population appears to be growing. However, this was based on a sample of the population, and not the entire population. One way to make inferences about the population is to ask whether the confidence interval for λ lies above 1.0. Let's use `DemoBoot` to bootstrap our confidence interval for λ.[9] First, we'll run the bootstrap, and plot the λ's.

```
> system.time(out.boot <- DemoBoot(stagedat, fruitdat, seeddat,
+     n = 1000))

   user  system elapsed
  9.606   0.020   9.630

> lambdas <- sapply(out.boot, function(out) out$lambda)

> hist(lambdas, prob = T)
> lines(density(lambdas))
```

From this it seems clear that the population is probably growing ($\lambda > 1.0$), because the lower limit of the histogram is relatively large (Fig. 2.6). We need to get a real confidence interval, however. Here we decide on a conventional α and then calculate quantiles, which will provide the median (the 50th percentile), and the lower and upper limits to the 95% confidence interval.[10]

```
> alpha <- 0.05
> quantile(lambdas, c(alpha/2, 0.5, 1 - alpha/2))

 2.5%   50% 97.5%
1.058 1.126 1.196
```

From this we see that the 95% confidence interval (i.e. the 0.025 and 0.975 quantiles) does not include 1.0. Therefore, we conclude that under the conditions experienced by this population in this year, this *Chamaedorea* population, from which we drew a sample, could be characterized as having a long-term asymptotic growth rate, λ, that is greaater than 1.0, and therefore would be likely to increase in abundance, if the environment remains the same.

[9] The number of replicates needed for a bootstrap depend in part on how close the interval is to critical points. If, for instance, your empirical P-value seems to be very close to your cutoff of $\alpha = 0.05$, then you should increase the replicates to be sure of your result. These days $n = 1000$ is considered a bare minimum.

[10] Quantiles are ordered points in a cumulative probability distribution function.

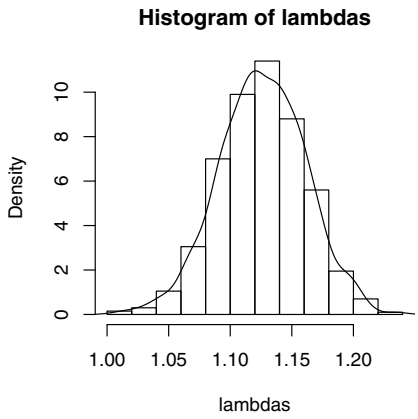


Fig. 2.6: The frequency distribution for our bootstrapped λ. Note that it is fairly symmetrical, and largely greater than 1.0.

A caveat and refinement

Bootstrapping as we have done above, known variously as the basic or percentile bootstrap, is not a cure-all, and it can give inappropriate estimation and inferrence under some circumstances. A number of refinements have been proposed that make bootstrapping a more precise and accurate procedure [46]. The problems are worst when the data are such that the bootstrap replicates are highly skewed, so that the mean and median are quite different. When the data are relatively symmetric, as ours is (Fig. 2.6), the inference is relatively reliable.

Often skewness will cause the mean of the bootstrap samples to differ from our observed estimate, and we refer to this as *bias*. We should adjust the bootstrapped samples for this bias [140]. Here we calculate the bias.

```
> bias <- mean(lambdas) - estims$lambda
> bias
```

```
[1] -0.007923
```

We find that the bias is very small; this gives us confidence the our confidence intervals are pretty good. Nonetheless, we can be thorough and correct our samples for this bias. We subtract the bias from the bootstrapped λ to get our confidence interval.

```
> quantile(lambdas - bias, c(alpha/2, 0.5, 1 - alpha/2))
```

```
 2.5%   50% 97.5%
1.066 1.134 1.204
```

These bias-corrected quantiles also indicate that this population in this year can be characterized by a $\lambda > 1$.

If we want to infer something about the future success of this population, we need to make additional assumptions. First, we must assume that our sample was representative of the population; we have every reason to expect it is. Second, we need to assume that this year was representative of other years. In particular, we need to assume that the weather, the harvest intensity, and the browsing intensity are all representative. Clearly, it would be nice to repeat this for other years, and to try to get other sources of information regarding these factors.

2.4 Summary

Demography is the study of structured populations. Structure may be described by age or stage, and is represented by life cycle graphs and a corresponding projection or transition matrix of transition probabilities and fertilities. The finite rate of increase, λ, and the stable stage/age distribution are key characteristics of a population, and are estimated using eigenalysis; populations will grow geometrically at the per capita rate of λ only when the population has reached its stable stage/age distribution. We measure the importance of transition elements with sensitivities and elasticities, the absolute or relative contributions λ of transition elements. Demographic information is frequently useful for endangered and invasive species.

Problems

2.1. Demographic analysis of a plant population
Goldenseal (*Hydrastis canadensis*) is a wild plant with medicinal properties that is widely harvested in eastern North American. Its rhizome (the thick underground stem) is dug up, and so harvesting can and frequently does have serious negative impacts on populations. A particular population of goldenseal is tracked over several years and investigators find, tag, and monitor several sizes of individuals [57]. After several years of surveys, they identify six relevant stages: dormant seed, seedling, small 1-leaved plant, medium 1-leaved plant, large 1-leaved plant, fertile plant (flowering, with 2 leaves). They determine that the population project matrix is:

$$\mathbf{A} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 1.642 \\ 0.098 & 0 & 0 & 0 & 0 & 0.437 \\ 0 & 0.342 & 0.591 & 0.050 & 0.095 & 0 \\ 0 & 0.026 & 0.295 & 0.774 & 0.177 & 0.194 \\ 0 & 0 & 0 & 0.145 & 0.596 & 0.362 \\ 0 & 0 & 0 & 0.016 & 0.277 & 0.489 \end{pmatrix} \tag{2.16}$$

(a) Draw a life cycle graph of this population of goldenseal. Include the matrix elements associated with each transition.
(b) Start with $\mathbf{N} = (0\,10\,10\,10\,10\,10)$ and graph population dynamics for all

stages for 10 years.
(c) Determine the stable stage distribution.
(d) Determine λ. Explain what this tells us about the population, including any assumptions regarding the stable stage distribution.
(d) Determine the elasticities. Which transition(s) are most influential in determining growth rate?
(e) Discuss which stages might be most suitable for harvesting; consider this question from both a financial and ecological perspective.

2.2. Demographic analysis of an animal population

Crouse et al. [44] performed a demographic analysis of an endangered sea turtle species, the loggerhead (*Caretta caretta*). Management of loggerhead populations seemed essential for their long term survival, and a popular management strategy had been and still is to protect nesting females, eggs, and hatchlings. The ground breaking work by Crouse[11] and her colleagues compiled data to create a stage-based projection matrix to analyze quantitatively which stages are important and least important in influencing long-term growth rate. This work led to US Federal laws requiring that US shrimp fishermen use nets that include Turtle Excluder Devices (TEDs, http://www.nmfs.noaa.gov/pr/species/turtles/teds.htm). Crouse et al. determined the transition matrix for their loggerhead populations:

$$\mathbf{A} = \begin{pmatrix} 0 & 0 & 0 & 0 & 127 & 4 & 80 \\ 0.6747 & 0.7370 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.0486 & 0.6610 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.0147 & 0.6907 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.0518 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.8091 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.8091 & 0.8089 \end{pmatrix} \tag{2.17}$$

(a) Draw a life cycle graph of this loggerhead population. Include the matrix elements associated with each transition.
(b) Determine the stable stage distribution.
(c) Determine λ. Explain what this tells us about the population, including any assumptions regarding the stable stage distribution.
(d) Determine the elasticities. Which transition(s) are most influential in determining growth rate?
(e) What is the predicted long-term relative abundance of all stages? What do we call this?
(f) If your interest is to maximize long-term growth rate, in which stage(s) should you invest protection measures? Which stages are least likely to enhance long-term growth rate, regardless of protective measures?
(g) Start with $\mathbf{N} = (0\,10\,10\,10\,10\,10\,10)$ and graph dynamics for all stages for 10 years.

[11] Crouse was a graduate student at the time — graduate students are the life-blood of modern science, doing cutting edge work and pushing their fields forward.