Use R!

M. Henry H. Stevens

A Primer of Ecology with R

Springer

M. Henry H. Stevens

A Primer of Ecology with R

SPIN 11683209

– Monograph –

April 7, 2009

Springer
Berlin Heidelberg New York
Hong Kong London
Milan Paris Tokyo

Preface

Goals and audience

In spite of the presumptuous title, my goals for this book are modest. I wrote it as

- the manual I wish I had in graduate school, and
- a primer for our graduate course in Population and Community Ecology at Miami University[1]

It is my hope that readers can enjoy the *ecological content* and ignore the R code, if they care to. Toward this end, I tried to make the code easy to ignore, by either putting boxes around it, or simply concentrating code in some sections and keeping it out of other sections.

It is also my hope that ecologists interested in learning R will have a rich yet gentle introduction to this amazing programming language. Toward that end, I have included some useful functions in an R package called `primer`. Like nearly all R packages, it is available through the R projects repositories, the CRAN mirrors. See the Appendix for an introduction to the R language.

I have a hard time learning something on my own, unless I can *do* something with the material. Learning ecology is no different, and I find that my students and I learn theory best when we write down formulae, manipulate them, and explore consequences of rearrangement. This typically starts with copying down, verbatim, an expression in a book or paper. Therefore, I encourage readers to take pencil to paper, and fingers to keyboard, and copy expressions they see in this book. After that, make sure that what I have done is correct by trying some of the same rearrangements and manipulations I have done. In addition, try things that aren't in the book — have fun.

A pedagogical suggestion

For centuries, musicians and composers have learned their craft in part by *copying by hand* to works of others. Physical embodiment of the musical notes

[1] Miami University is located in the Miami River valley in Oxford, Ohio, USA; the region is home to the Myaamia tribe that dwelled here prior to European occupation.

and their sequences helped them learn composition. I have it on great authority that most theoreticians (and other mathematicians) do the same thing — they start by copying down mathematical expressions. This physical process helps get the content under their skin and through their skull. I encourage you to do the same. Whether otherwise indicated or not, let the first assigned problem at the end of each chapter be to copy down, with a pencil and paper, the mathematical expression presented in that chapter. In my own self-guided learning, I have often taken this simple activity for granted and have discounted its value — to my own detriment. I am not surprised how often students also take this activity for granted, and similarly suffer the consequences. *Seeing* the logic of something is not always enough — sometimes we have to actually *recreate* the logic for ourselves.

Comparison to other texts

It may be useful to compare this book to others of a similar ilk. This book bears its closest similarities to two other wonderful primers: Gotelli's *A Primer of Ecology*, and Roughgarden's *Primer of Theoretical Ecology*. I am more familiar with these books than any other introductory texts, and I am greatly indebted to these authors for their contributions to my education and the discipline as a whole.

My book, geared toward graduate students, includes more advanced material than Gotelli's primer, but most of the ecological topics are similar. I attempt to start in the same place (e.g., "What is geometric growth?"), but I develop many of the ideas much further. Unlike Gotelli, I do not cover life tables at all, but rather, I devote an entire chapter to *demographic matrix models*. I include a chapter on community structure and diversity, including *multivariate distances*, *species-abundance distributions*, *species-area relations*, and *island biogeography*, as well as *diversity partitioning*. My book also includes code to implement most of the ideas, whereas Gotelli's primer does not.

This book also differs from Roughgarden's primer, in that I use the Open Source R programming language, rather than Matlab®, and I do not cover physiology or evolution. My philosphical approach is similar, however, as I tend to "talk" to the reader, and we fall down the rabbit hole together[2].

Aside from Gotelli and Roughgarden's books, this book bears similarity in content to several other wonderful introductions to mathematical ecology or biology. I could have cited repeatedly (and in some places did so) the following: Ellner and Guckenheimer (2006), Gurney and Nisbet (1998), Kingsland (1985), MacArthur (1972), Magurran (2004), May (2001), Morin (1999), Otto and Day (2006), and Vandermeer and Goldberg (2007). Still others exist, but I have not yet had the good fortune to dig too deeply into them.

Acknowledgements

I am indebted to Scott Meiners and his colleagues for their generous sharing of data, metadata, and statistical summaries from the Buell-Small Succession

[2] From *Alice's Adventures in Wonderland* (1865), L. Carroll (C. L. Dodgson).

Study (http://www.ecostudies.org/bss/), a 50+ year study of secondary succession (supported in part by NSF grant DEB-0424605) in the North American temperate deciduous forest biome. I would like to thank Stephen Ellner for Ross's Bombay death data and for R code and insight over the past few years. I am also indebted to Tom Crist and his colleagues for sharing some of their moth data (work supported by The Nature Conservancy Ecosystem Research Program NSF DEB-0235369).

I am grateful for the generosity of early reviewers and readers, each of whom has contributed much to the quality of this work: Jeremy Ash, Tom Crist, David Gorchov, Raphael Herrera-Herrera, Thomas Petzoldt, James Vonesh, as well as several anonymous reviewers, and the students of our Population and Community Ecology class. I am also grateful for the many conversations and emails shared with four wonderful mathematicians and theoreticians: Jayanth Banavar, Ben bolker, Stephen Ellner, Amit Shukla, and Steve Wright — I never have a conversation with these people without learning something. I have been particularly fortunate to have team-taught Population and Community Ecology at Miami University with two wonderful scientists and educators, Davd Gorchov and Thomas Crist. Only with this experience, of working closely with these colleagues, have I been able to attempt this book. It should go without saying, but I will emphasis, that the mistakes in this book are mine, and there would be many more but for the sharp eyes and insightful minds of many other people.

I am also deeply indebted to the R Core Development Team for creating, maintaining and pushing forward the R programming language and environment [173]. Like the air I breathe, I cannot imagine my (professional) life without it. I would especially like to thank Friedrich Leisch for the development of `Sweave`, which makes literate programming easy [106]. Because I rely on Aquamacs, ESS, LaTeX, and a host of other Open Source programs, I am deeply grateful to those who create and distribute these amazing tools.

A few other R packages bear special mention. First, Ben Bolker's text [13] and packages for modeling ecological data (`bbmle` and `emdbook`) are broadly applicable. Second, Thomas Petzoldt's and Karsten Rinke's `simecol` package provides a general computational architecture for ecological models, and implements many wonderful examples [158]. Much of what is done in this primer (especially in chapters 1, 3–6, 8) can be done with `simecol`, and sometimes done better. Third, Robin Hankin's `untb` package is an excellent resource for exploring ecological neutral theory (chapter 10) [69]. Last, I relied heavily on the `deSolve` [190] and `vegan` packages [151].

Last, and most importantly, I would like to thank those to whom this book is dedicated, whose love and senses of humor make it all worthwhile.

Martin Henry Hoffman Stevens
Oxford, OH, USA, Earth
February, 2009

Contents

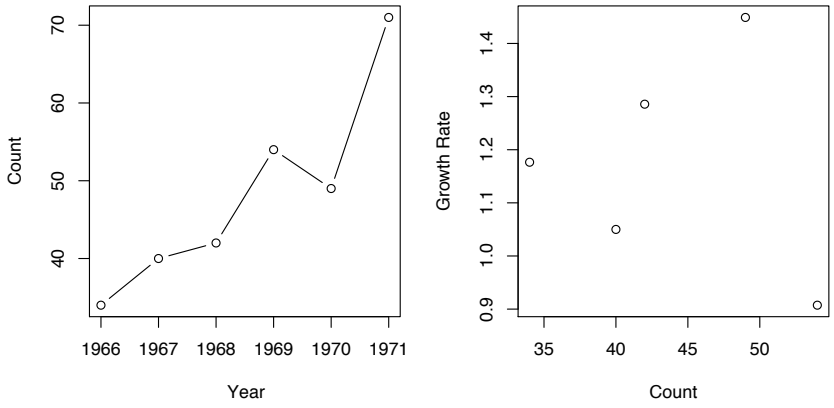
Part III Special Topics

Part I

Single Species Populations

1

Simple Density-independent Growth



(a) Counts of Song Sparrows (b) Relative Annual Change *vs.* N

Fig. 1.1: Song Sparrow (*Melospiza melodia*) counts in Darrtown, OH, USA. From Sauer, J. R., J. E. Hines, and J. Fallon. 2005. The North American Breeding Bird Survey, Results and Analysis 1966–2004. Version 2005.2. USGS Patuxent Wildlife Research Center, Laurel, MD.

Between 1966 and 1971, Song Sparrow (*Melospiza melodia*) abundance in Darrtown, OH, USA, seemed to increase very quickly, seemingly unimpeded by any particular factor (Fig. 1.1a). In an effort to manage this population, we may want to predict its future population size. We may also want to describe its growth rate and population size in terms of mechanisms that could influence its growth rate. We may want to compare its growth and relevant mechanisms to those of other Song Sparrow populations or even to other passerine populations. These goals, *prediction*, *explanation*, and *generalization*, are frequently the goals toward which we strive in modeling anything, including populations, communi-

ties, and ecosystems. In this book, we start with simple models of populations and slowly add complexity to both the form of the model, and the analysis of its behavior. As we move along, we also practice applying these models to real populations.

What is a model, and why are they important in ecology? First, a model is an abstraction of reality. A road map, for instance, that you might use to find your way from Mumbai to Silvasaa is a model of the road network that allows you to predict which roads will get you to Silvasaa. As such, it *excludes* far more information about western India than it includes. Partly as a result of excluding this information, it is eminently useful for planning a trip. Models in ecology are similar. Ecological systems are potentially vastly more complex than just about any other system one can imagine for the simple reason that ecosystems are composed of vast numbers of genetically distinct individuals, each of which is composed of at least one cell (e.g., a bacterium), and all of these individuals may interact, at least indirectly. Ecological models are designed to capture particular key features of these potentially complex systems. The goal is to capture a key feature that is particularly interesting and useful.

In this book, we begin with the phenomenon called *density-independent growth*. We consider it at the beginning of the book for a few reasons. First, the fundamental process of reproduction (e.g., making seeds or babies) results in a *geometric series*[1]. For instance, one cell divides to make two, those two cells each divide to make four, and so on, where reproduction for each cell results in two cells, *regardless of how many other cells are in the population* — that is what we mean by *density-independent*. This myopically observed event of reproduction, whether one cell into two, or one plant producing many seeds, is the genesis of a geometric series. Therefore, most models of populations include this fundamental process of geometric increase. Second, populations can grow in a density-independent fashion when resources are plentiful. Third, it behooves us to start with this simple model because most, more complex population models include this process.

1.1 A Very Specific Definition

Density-independence in a real population is perhaps best defined quite specifically and operationally as a lack of a statistical relation between the density of a population, and its *per capita* growth rate. The power to detect a significant relation depends, in part, on those factors which govern power in statistical relations between any two continuous variables: the number of observations, and the range of the predictor variable. Therefore, our conclusion, that a particular population exhibits density-independent growth, may be trivial if our sample size is small (i.e., few generations sampled), or if we sampled the population over a very narrow range of densities. Nonetheless, it behooves us to come back

[1] A mathematical series is typically a list of numbers that follow a rule, and that you sum together.

to this definition if, or when, we get caught up in the biology of a particular organism.

We could examine directly the relation between the growth rate and population size of our Song Sparrow population (Fig. 1.1b). We see that there is no apparent relation between the growth rate and the density of the population[2]. That is what we mean by "density-independent growth."

1.2 A Simple Example

Let's pretend you own a small piece of property and on that property is a pond. Way back in June 2000, as a present for Mother's Day, you were given a water lily (*Nymphaea odorata*), and you promptly planted it, with it's single leaf or frond, in the shallows of your pond. The summer passes, and your lily blossoms, producing a beautiful white flower. The following June (2001) you notice that the lily grew back, and that there were three leaves, not just one. Perhaps you cannot discern whether the three leaves are separate plants. Regardless, the pond now seems to contain three times the amount of lily pad that it had last year.

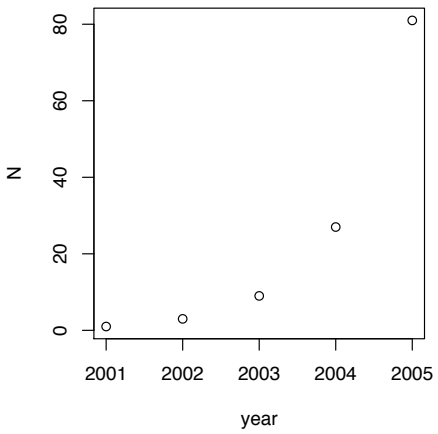
The following June (2002) you are pleased to find that instead of three leaves, you now have nine. In June 2003, you have 27 leaves, and in 2004 you have 81 leaves (Fig. 1.3). How do we describe this pattern of growth? How do we predict the size of the population in the future? Can we take lessons learned from our water lily and apply it to white-tailed deer in suburbia, or to bacteria in the kitchen sink?

We rely on theory to understand and describe the growth of our water lily in such a way as to apply it to other populations. The role of theory, and theoretical ecology, is basically three-fold. We might like theory to allow us to describe the pattern in sufficient detail (1) to provide a *mechanistic explanation* for how the lily grew as fast or as slowly as it did, (2) allow us to make *predictions* about the population size in the future, and (3) allow us to *generalize* among other lily populations or other species. These goals typically compete with each other, so real models are mathematical descriptions that result from tradeoffs among these goals which depend precisely on our particular needs [109].

1.3 Exploring Population Growth

So, how fast are the lilies of the example growing? Between years 1 and 2, it increased by 2 fronds; between years 2 and 3, it increased by 6. In subsequent years it increased by 18, and 54 fronds. The number changes each year (Fig. 1.3), so how do we predict the future, or even explain the present? Can we find a general rule that works for any year?

[2] Consider that if area is fixed, "count" or population size differs from density by a fixed multiplier



ht

Fig. 1.2: Hypothetical water lily population size through time.

Simple Graphing of Population Size (Fig. 1.3)

Here we create two vectors: population size, N, and years. Using `c()` allows us to create an arbitrary vector, and the colon, `:`, provides a sequence of consecutive integers.

```
> N <- c(1, 3, 9, 27, 81)
> year <- 2001:2005
> plot(year, N)
```

The lily population (Fig. 1.3) increases by a different amount each year. What about proportions — does it increase by a different proportion each year? Let's divide each year's population size by the previous year's size, that is, perform N_{t+1}/N_t for all t, where t is any particular point in time, and $t+1$ is the next point in time. For N, that amounts to 3/1, 9/3, ... What do we notice?

Vectorized math

Here we divide each element of one vector (the second through fifth element of N) by each element of another vector (the first through fourth elements of N).

```
> rates = N[2:5]/N[1:4]
> rates

[1] 3 3 3 3
```

Lo, and behold! all of these proportions are the same: 3. Every year, the lilies increase by the same proportion — they triple in abundance, increasing by 200%. That is the general rule that is specific to our population. It is general because it appears to apply to each year, and could potentially describe other populations; it is not, for instance, based on the photosynthetic rate in lily pads. It is specific because it describes a specific rate of increase for our water lily

population. We can represent this as

$$N_{2002} = 3 \times N_{2001}$$

where N_{2002} is the size of the population in 2002. If we rearrange this, dividing both sides by N_{2001}, we get

$$\frac{N_{2002}}{N_{2001}} = 3$$

where 2 is our rate of increase.

Generalizing this principle, we can state

$$N_{t+1} = 3N_t$$

$$\frac{N_{t+1}}{N_t} = 3.$$

1.3.1 Projecting population into the future

The above equations let us describe the rate of change population size N from one year to the next, but how do we predict the size 10 or 20 years hence? Let's start with one year and go from there.

$$\begin{aligned} N_{2002} &= 3N_{2001} \\ N_{2003} &= 3N_{2002} = 3\,(3N_{2001}) \\ N_{2004} &= 3N_{2003} = 3\,(3N_{2002}) = 3\,(3\,(2N_{2001})) \end{aligned}$$

So, ... what is the general rule that is emerging for predicting water lily N, some years hence? Recall that $3 \times 3 \times 3 = 3^3$ or $a \times a \times a = a^3$, so more generally, we like to state

$$N_t = \lambda^t N_0 \tag{1.1}$$

where t is the number of time units (years in our example), N_0 is the size of our population at the beginning, λ is the per capita rate of increase over the specified time interval and N_t is the predicted size of the population after t time units.

Note that lambda, λ, is the *finite rate of increase*. It is the per capita rate of growth of a population *if the population is growing geometrically*. We discuss some of the finer points of λ in Chapter 2. We can also define a related term, the *discrete growth factor*, r_d, where $\lambda = (1 + r_d)$.

Note that "time" is not in calendar years but rather in years since the initial time period. It is also the number of time steps. Imagine that someone sampled a population for five years, 1963–1967, then we have four time steps, and $t = 4$.

Projecting population size

Here we calculate population sizes for 10 time points beyond the initial. First we assign values for N_0, λ, and time.

```
> N0 <- 1
> lambda <- 2
> time <- 0:10
```

Next we calculate N_t directly using our general formula.

```
> Nt <- N0 * lambda^time
> Nt

 [1]    1    2    4    8   16   32   64  128  256  512 1024
```

1.3.2 Effects of initial population size

Let's explore the effects of initial population size. First, if we just examine equation 1.1, we will note that $N_t = N_0 \times \text{stuff}$. Therefore, if one population starts out twice as big as another, then it will always be twice as big, given geometric growth (Fig. 1.3a). We see that small initial differences diverge wildly over time (Fig. 1.3a), because "twice as big" just *looks* a lot bigger as the magnitude increases.

Effects of Initial Population Size

We first set up several different initial values, provide a fixed λ, and set times from zero to 4.

```
> N0 <- c(10, 20, 30)
> lambda <- 2
> time <- 0:4
```

We calculate population sizes at once using `sapply` to *apply* a function (`n*lambda^time`) to each element of the first argument (each element of `N0`).

```
> Nt.s <- sapply(N0, function(n) n * lambda^time)
> Nt.s

     [,1] [,2] [,3]
[1,]   10   20   30
[2,]   20   40   60
[3,]   40   80  120
[4,]   80  160  240
[5,]  160  320  480
```

The result is a matrix, and we see N_0 in the first row, and each population is in its own column. Note that population 2 is always twice as big as population 1.

If we change the *y*-axis scale to logarithms, however, we see that the lines are parallel! Logarithms are a little weird, but they allow us to look at, and think

about, many processes where rates are involved, or where we are especially interested in the relative magnitudes of variables. Consider the old rule we get when we take the logarithm of both sides of an equation, where the right hand side is a ratio.

$$y = \frac{a}{b} \tag{1.2}$$

$$\log y = \log\left(\frac{a}{b}\right) = \log a - \log b \tag{1.3}$$

Thus, when we change everything into logarithms, ratios (like λ) become *differences*, which result in straight lines in graphs (Fig. 1.3b). On a linear scale, populations that are changing at the same *rates* can look very different (Fig. 1.3a), whereas on a logarithmic scale, the populations will have *parallel* trajectories (Fig. 1.3b).

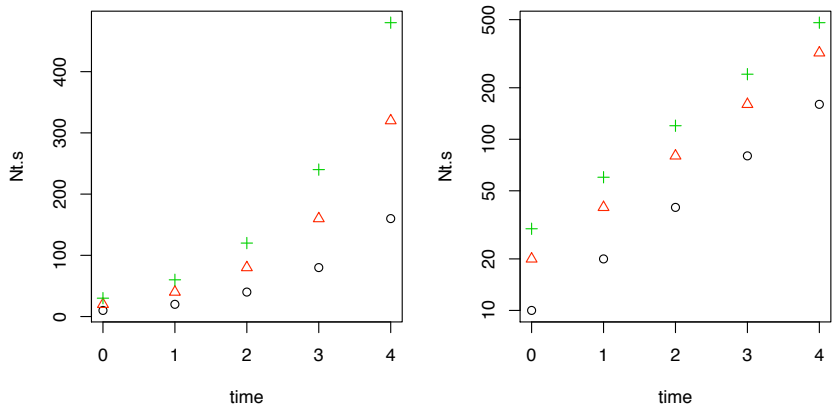
Graphing a Matrix (Figs. 1.3a, 1.3b)

We can use `matplot` to plot a matrix *vs.* a single vector on the X-axis. By default it labels the points according to the number of the column

```
> matplot(time, Nt.s, pch = 1:3)
```

We can also plot it with a log scale on the *y*-axis.

```
> matplot(time, Nt.s, log = "y", pch = 1:3)
```



(a) Arithmetic Scale on y (b) Logarithmic Scale on y

Fig. 1.3: Effects of variation in initial N on population size, through time. Different symbols indicate different populations.

Note that changing the initial population size changes the intercept. It also changes the slope in linear space, but not in log-linear space. It changes the absolute rate of increase ($N_2 - N_1$), but not the relative rate of increase (N_2/N_1).

1.3.3 Effects of different per capita growth rates

Perhaps the most important single thing we can say about λ is that when $\lambda < 1$, the population shrinks, and when $\lambda > 1$ the population grows. If we examine eq 1.1, $N_t = \lambda^t N_0$, we will note that λ is exponentiated, that is, raised to a power[3]. It will always be true that when $\lambda > 1$ and $t > 1$, $\lambda^t > \lambda$. It will also be true that when $\lambda < 1$ and $t > 1$, $\lambda^t < \lambda$ (Fig. 1.4).

Thus we see the basis of a very simple but important truism. *When $\lambda > 1$, the population grows, and when $\lambda < 1$ the population shrinks* (Fig. 1.4). When $\lambda = 1$, the population size does not change, because $1^t = 1$ for all t.

Effects of Different λ (Fig. 1.4)

Here we demonstrate the effects on growth of $\lambda > 1$ and $\lambda < 1$. We set $N_0 = 100$, and time, and then pick three different λ.

```
> N0 <- 100
> time <- 0:3
> lambdas <- c(0.5, 1, 1.5)
```

We use `sapply` again to apply the geometric growth function to each λ. This time, `x` stands for each λ, which our function then uses to calculate population size. We then plot it, and add a reference line and a little text.

```
> N.all <- sapply(lambdas, function(x) N0 * x^time)

> matplot(time, N.all, xlab = "Years", ylab = "N", pch = 1:3)
> abline(h = N0, lty = 3)
> text(0.5, 250, expression(lambda > 1), cex = 1.2)
> text(0.5, 20, expression(lambda < 1), cex = 1.2)
```

The reference line is a *h*orizontal line with the *l*ine *ty*pe dotted. Our text simply indicates the regions of positive and negative growth.

We note that we have graphed *discrete* population growth. If we are counting bodies, and the population reproduces once per year, then the population will jump following all the births (or emergence from eggs). Further, it is probably always the case that following a bout of synchronous reproduction, we observe chronic ongoing mortality, with the result of population decline between spikes of reproduction. Nonetheless, unless we collect the data, we can't really say much about what goes on in between census periods.

1.3.4 Average growth rate

In any real data set, such as from a real population of *Nymphaea*, N_{t+1}/N_t will vary from year to year. Let's examine this with a new data set in which annual growth rate varies from year to year.

[3] What happens to y^x as x increases, if $y > 1$ — does y^x increase? What happens if $y < 1$ — does y^x decrease? The answer to both these questions is yes.

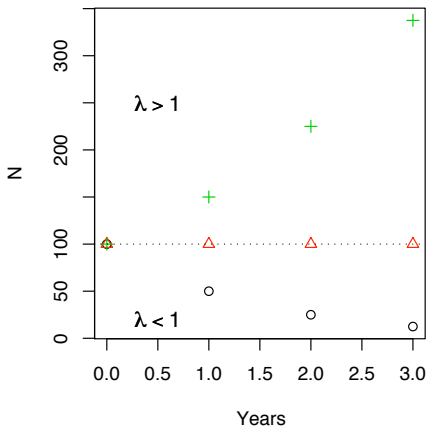


Fig. 1.4: Effects of variation in λ on population size through time. The dotted line indicates no change ($N_t = N_0, \quad \lambda = 1$). Numbers (1, 2, 3) indicate populations resulting from λ = (0.5, 1.0, 1.5), respectively. Any λ greater than 1 results in positive geometric growth; any $\lambda < 1$ results in negative geometric growth, or population decline.

Since growth rate varies from year to year, we may want to calculate average annual growth rate over several years. As we see below, however, the arithmetic averages are not really appropriate.

Consider that N_{t+1}/N_t may be a random variable which we will call R[4]. That is, this ratio from any one particular year could take on a wide variety of values, from close to zero, up to some (unknown) large number. Let's pick two out of a hat, where $R = 0.5, 1.5$. The arithmetic average of these is 1.0, so this might seem to predict that, on average, the population does not change. Let's project the population for two years using each R once.

$$\begin{aligned} N_0 &= 100 \\ N_1 &= N_0\,(0.5) = 50 \\ N_2 &= N_1\,(1.5) = 75 \end{aligned}$$

We started with 100 individuals, but the population shrank! Why did that happen? It happens because, in essence, we *multiply* the λ together, where $N_2 = N0\,R_1\,R_2$. In this case, then, what is a sensible "average"?

How do we calculate an average for things that we multiply together? We would like a value for R which would provide the solution to

$$\bar{R}^t = R_1 R_2 \ldots R_t \tag{1.4}$$

where t is the number of time steps and R_1 is the observed finite rate of increase from year 1 to year 2. The bar over R indicates a mean.

[4] Some authors use R for very specific purposes, much as one might use λ; here we just use it for a convenient letter to represent observed per capita change.

All we have to do is solve for R.

$$\left(\bar{R}^t\right)^{1/t} = (R_1 R_2 \ldots R_t)^{1/t} \tag{1.5}$$

$$\bar{R} = (R_1 R_2 \ldots R_t)^{1/t} \tag{1.6}$$

$$\tag{1.7}$$

We take the t-th root of the product of all the R. This is called the *geometric average*. Another way of writing this would be to use the product symbol, Π, as in

$$\bar{R} = \left(\prod_{i=1}^{t} R_i\right)^{1/t} \tag{1.8}$$

If we examine the Song Sparrow data (Fig. 1.5), we see that projections based on the geometric average R are less than when based on the arithmetic average; this is always the case.

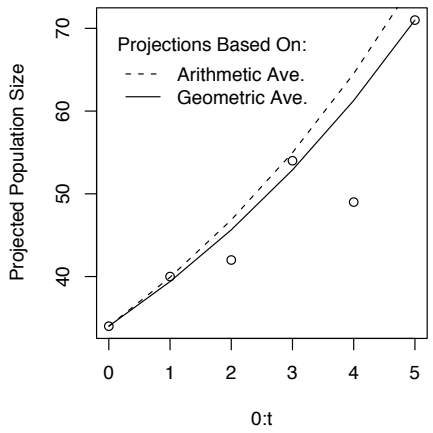


Fig. 1.5: Song Sparrow population sizes, and projections based on arithmetic and geometric mean R.

Comparing arithmetic and geometric averages (Fig. 1.5)

First we select the number of observed R ($t = 5$); this will require that we use six years of Song Sparrow data.

```
> t <- 5
> SS6 <- sparrows[1:(t + 1), ]
```

Next we calculate λ for each generation, from t to $t+1$, and calculate the arithmetic and geometric means.

```
> SSgr <- SS6$Count[2:(t + 1)]/SS6$Count[1:t]
> lam.A <- sum(SSgr)/t
> lam.G <- prod(SSgr)^(1/t)
```

Now we can plot the data, and the projections based on the two averages (Fig. 1.5).

```
> N0 <- SS6$Count[1]
> plot(0:t, SS6$Count, ylab = "Projected Population Size")
> lines(0:t, N0 * lam.A^(0:t), lty = 2)
> lines(0:t, N0 * lam.G^(0:t), lty = 1)
> legend(0, 70, c("Arithmetic Ave.", "Geometric Ave."),
+     title = "Projections Based On:",
+     lty = 2:1, bty = "n", xjust = 0)
```

1.4 Continuous Exponential Growth

Although many, many organisms are modeled well with discrete growth models (e.g., insects, plants, seasonally reproducing large mammals), many populations are poorly represented by discrete growth models. These populations (e.g., bacteria, humans) are often modeled as continuously growing populations. Such models take advantage of simple calculus, the mathematics of rates.

Whereas geometric growth is proportional change in a population over a specified finite time interval, exponential growth is proportional *instantaneous* change over, well, an instant.

Imagine a population of *Escherichia coli* started by inoculating fresh Luria-Bertania medium with a stab of *E. coli* culture. We start at time zero with about 1000 cells or CFUs (colony forming units), and wind up the next day with 10^{10} cells. If we used (incorrectly) a discrete growth model, we could calculate N_{t+1}/N_t and use this as an estimate for λ, where $\lambda = 10^{10}/10^3 = 10^7$ cells per cell per day. We know, however, that this is a pretty coarse level of understanding about the dynamics of this system. Each cell cycle is largely asynchronous with the others, and so many cells are dividing each second. We could simply define our time scale closer to the average generation time of a cell, for example $\lambda = 2$ cells cell^{-1} 0.5 h^{-1}, but the resulting discrete steps in population growth would still be a poor representation of what is going on. Rather, we see population size changing very smoothly from hour to hour, minute to minute. Can we come up with a better description? Of course.

1.4.1 Motivating continuous exponential growth

If we assume that our *E. coli* cells are dividing asynchronously, then many cells are dividing each fraction of a second — we would like to make that fraction of a second an *infinitely* small time step. Unfortunately, that would mean that we have an infinitely large number of time steps between $t = 0$ and $t = 1$ day, and we couldn't solve anything.

A long time ago, a very smart person[5] realized that geometric growth depends on how often you think a step of increase occurs. Imagine you think a population increases at an annual growth rate $\lambda = 1.5$. This represents a 50% increase or

$$N_1 = N_0 (1 + 0.5)$$

so the *discrete growth increment* is $r_d = 0.5$. You could reason that twice-annual reproduction would result in half of the annual r_d. You could then do growth over two time steps, and so we would then raise λ^2, because the population is taking two, albeit smaller, time steps. Thus we would have

$$N_1 = N_0 (1 + 0.5/2)^2 = N_0 (1 + 0.25)^2$$

What if we kept increasing the number of time steps, and decreasing the growth increment? We could represent this as

$$N_1 = N_0 \left(1 + \frac{r_d}{n}\right)^n$$
$$\frac{N_1}{N_0} = \left(1 + \frac{r_d}{n}\right)^n$$

Our question then becomes, what is the value of $\left(1 + \frac{r_d}{n}\right)^n$ as n goes to infinity? In mathematics, we might state that we would like the solution to

$$\lim_{n \to \infty} \left(1 + \frac{r_d}{n}\right)^n . \tag{1.9}$$

To begin with, we simply try larger and larger values of n, graph eq. 1.9 *vs.* n, and look for a limit (Fig. 1.6).

[5] Jacob Bernoulli (1654–1705)

Numerical approximation of e

Here we use brute force to try to get an approximate solution to eq. 1.9. We'll let n be the number of divisions within one year. This implies that the finite rate of increase during each of these fractional time steps is r_d/n. Let the $\lambda = 2$ and therefore $r_d = 1$. Note that because $N_0 = 1$, we could ignore it, but let's keep it in for completeness.

```
> n <- 0:100; N0 <- 1; rd <- 1
```

Next, we calculate $\left(1 + \frac{r_d}{n}\right)^n$ for ever larger values of n.

```
> N1 <- N0 * (1 + rd/n)^n
```

Last, we plot the ratio and add some fancy math text to the plot (see `?plotmath` for details on mathematical typesetting in R).

```
> plot(n, N1/N0, type = "l")
> text(50, 2, "For n = 100,")
> text(50, 1.6, bquote((1 + frac("r"["d"], "n"))^"n" ==
+     .(round(N1[101]/N0, 3))))
```

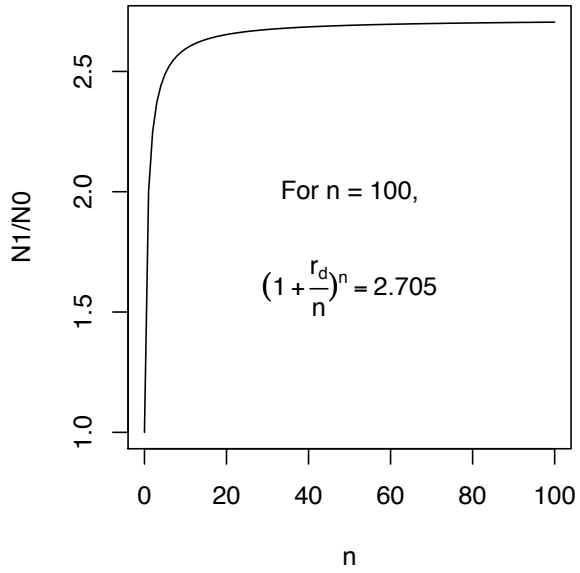


Fig. 1.6: The limit to subdividing reproduction into smaller steps. We can compare this numerical approximation to the true value, $e^1 = 2.718$.

Thus, when reproduction occurs continuously, the population can begin to add to itself right away. Indeed, if a population grew in a discrete annual step $N_{t+1} = N_t(1 + r_d)$, the same r_d, divided up into many small increments, would result in a much larger increase.

It turns out that the increase has a simple mathematical expression, and we call it the *exponential*, e. As you probably recall, e is one of the magic numbers in mathematics that keeps popping up everywhere. In this context, we find that

$$\lim_{n\to\infty}\left(1+\frac{r}{n}\right)^n = e^r \tag{1.10}$$

where e is the *exponential*.

This means that when a population grows geometrically, with infinitely small time steps, we say the population grows *exponentially*, and we represent that as,

$$N_t = N_0 e^{rt}. \tag{1.11}$$

We call r the *instantaneous* per capita growth rate, or the *intrinsic rate of increase*.

Projection of population size with continuous exponential growth is thus no more difficult than with discrete growth (Fig. 1.7).

Projecting a continuous population

We select five different values for r: two negative, zero, and two positive. We let t include the integers from 1 to 100. We then use `sapply` to *apply* our function of continuous exponential growth to each r, across all time steps. This results in a matrix where each row is the population size at each time t, and each column uses a different r.

```
> r <- c(-0.03, -0.02, 0, 0.02, 0.03)
> N0 <- 2; t <- 1:100
> cont.mat <- sapply(r, function(ri) N0 * exp(ri * t))
```

Next we create side-by-side plots, using both arithmetic and logarithmic scales, and add a legend.

```
> layout(matrix(1:2, nrow = 1))
> matplot(t, cont.mat, type = "l", ylab = "N", col = 1)
> legend("topleft", paste(rev(r)), lty = 5:1, col = 1, bty = "n",
+     title = "r")
> matplot(t, cont.mat, type = "l", ylab = "N", log = "y", col = 1)
```

1.4.2 Deriving the time derivative

We can also differentiate eq. 1.11 with respect to time to get the differential equation for instantaneous population growth rate. Recall that the chain rule tells us that the derivative of a product of two terms is the sum of the products of the derivative of one times the other original term.

$$\frac{\mathrm{d}}{\mathrm{d}t}(XY) = \frac{\mathrm{d}X}{\mathrm{d}t}Y + \frac{\mathrm{d}Y}{\mathrm{d}t}X$$

Therefore to begin to differentiate eq. 1.11, with respect to t, we have,

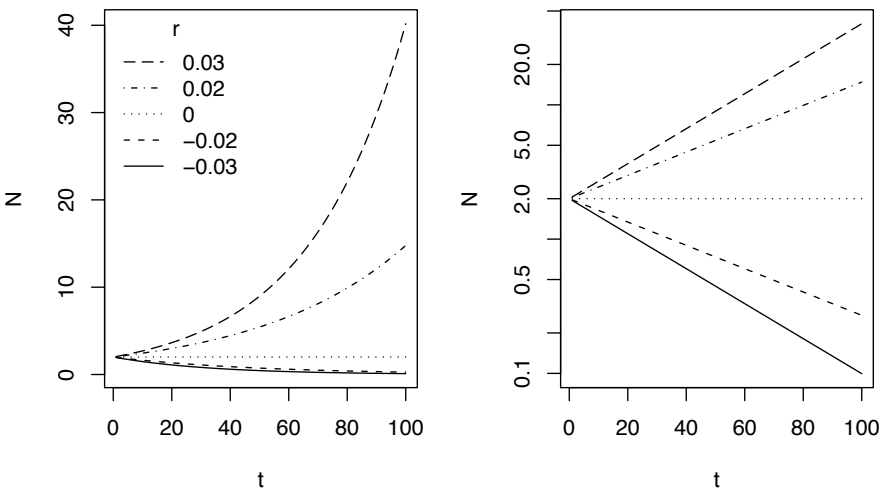


Fig. 1.7: Projecting continuous populations with different r.

$$\frac{\mathrm{d}}{\mathrm{d}t} N_0 e^{rt} = \frac{\mathrm{d}}{\mathrm{d}t} N_0 \cdot (e^r)^t + \frac{\mathrm{d}}{\mathrm{d}t} (e^r)^t \cdot N_0$$

Recall also that the derivative of a constant is zero, and the derivative of a^t is $\ln a\,(a)^t$, resulting in,

$$\frac{\mathrm{d}}{\mathrm{d}t} N_0 e^{rt} = 0 \cdot (e^r)^t + \ln e^r \cdot (e^r)^t \cdot N_0$$

Given that $\ln e = 1$, and that $N_0 e^{rt} = N$ for any time t, this reduces to eq. 1.12. The time derivative, or differential equation, for exponential growth is

$$\frac{\mathrm{d}N}{\mathrm{d}t} = rN. \tag{1.12}$$

1.4.3 Doubling (and tripling) time

For heuristic purposes, it is frequently nice to express differences in growth rates in terms of something more tangible than a dimensionless number like r. It is relatively concrete to say population X increases by about 10% each year ($\lambda = 1.10$), but another way to describe the rate of change of a population is to state the amount of *time* associated with a dramatic but comprehensible change. The *doubling time* of a population is the time required for a population to double in size. Shorter doubling times therefore mean more rapid growth.

We could determine doubling time graphically. If we examine population 3 in Fig. 1.4, we see that it takes about one and half years for the population size to change from $N = 100$ to $N = 200$. Not surprisingly, we can do better than

that. By doubling, we mean that $N_t = 2N_0$. To get the time at which this occurs, we solve eq. (1.11) for t,

$$2N_0 = N_0 e^{rt} \tag{1.13}$$

$$2 = e^{rt} \tag{1.14}$$

$$\ln(2) = rt \ln(e) \tag{1.15}$$

$$t = \frac{\ln(2)}{r}. \tag{1.16}$$

Thus, eq. 1.16 gives us the time required for a population to double, given a particular r. We could also get any arbitrary multiple m of any arbitrary initial N_0.

Creating a function for doubling time

We can create a *function* for this formula, and then evaluate it for different values of m and r. For $m = 2$, we refer to this as "doubling time." When we define the function and include arguments `r` and `m`, we also set a default value for `m=2`. This way, we do not always have to type a value for m; be default the function will return the doubling time.

```
> m.time <- function(r, m = 2) {
+     log(m)/r
+ }
```

Now we create a vector of r, and then use `m.time` to generate a vector of doubling times.

```
> rs <- c(0, 1, 2)
> m.time(rs)
```

```
[1]    Inf 0.6931 0.3466
```

Note that R tells us that when $r = 0$, it takes an infinite (`Inf`) amount of time to double. This is what we get when we try to divide by zero!

1.4.4 Relating λ and r

If we assume constant exponential and geometric growth, we can calculate r from data as easily as λ. Note that, so rearranging, we see that

$$N_t = N_0 e^{rt} \ln(N_t) = \ln(N_0) + rt.$$

In other words, r is the slope of the linear relation between $\ln(N_t)$ and time (Fig. 1.7), and $\ln(N_0)$ is the y-intercept. If the data make sense to fit a straight regression line to log-transformed data, the slope of that line would be r.

It also is plain that,

$$\lambda = e^r \tag{1.17}$$

$$\ln \lambda = r. \tag{1.18}$$

Summarizing some of what we know about how λ and r relate to population growth:

No Change	$\lambda = 1\ ,\ r = 0$
Population Growth	$\lambda > 1\ ,\ r > 0$
Population Decline	$\lambda < 1\ ,\ r < 0$

Remember — λ is for populations with discrete generations, and r is for continuously reproducing populations.

Units

What are the units for λ and r? As λ is a ratio of two population sizes, the units could be individuals/individual, thus rendering λ dimensionless. Similarly, we can view λ as the net number of individuals produced by individuals in the population such that the units are net new individuals per individual per time step, or $\text{inds}\,\text{ind}^{-1}\,t^{-1}$. The intrinsic rate of increase, r, is also in units of $\text{inds}\,\text{ind}^{-1}\,t^{-1}$

Converting between time units

A nice feature of r as opposed to λ is that r can be used to scale easily among time units. Thus, $r = 0.1\,\text{inds}\,\text{ind}^{-1}\,\text{year}^{-1}$ becomes $r = 0.1/365 = 0.00027\,\text{inds}\,\text{ind}^{-1}\,\text{day}^{-1}$. *You cannot do this with* λ. If you would like to scale λ from one time unit to another, first convert it to r using logarithms, make the conversion, then convert back to λ.

1.5 Comments on Simple Density-independent Growth Models

It almost goes without saying that if we are considering density-independent growth models to be representative of real populations, we might feel as though we are making a very long list of unrealistic assumptions. These might include no immigration or emigration, no population structure (i.e. all individuals are identical), and you can probably come up with many others [58]. However, I would argue vociferously that we are making only one assumption:

> *N increases by a constant per capita rate over the time interval(s) of interest.*

Think about that. I am *not* saying that competition is not occurring, or that no death is occurring, or that all individuals are reproductively viable, or there is no abiotic disturbance, or that there is no population genetic structure. I am just saying that for the time period of interest, all things balance out, or are of no substantive consequence, and the population chugs along at a particular pace.

If the per capita rate is constant, then there can be no statistical relation between the size of the population and its per capita growth rate. *In the absence of such a relation*, we say that the growth rate is density-independent.

Other ecologists will disagree with my sentiments regarding an absence of assumptions. That's OK — still others may agree with these sentiments. Take it upon yourself to acquire multiple perspectives and evaluate them yourself.

Both λ and r obviously depend on birth rates and death rates. For instance, we could view geometric growth as

$$N_{t+1} = N_t + BN_t - DN_t \tag{1.19}$$

where B is the number of births per individual and D is the probability of an individual dying during the specified time interval. Lambda, in this case, is $1+(B-D)$ and $r_d = B-D$. This form would be nice if we had data on births and deaths, because, after all, one goal of Science is to explain complex phenomena (e.g., λ) in terms of their determinants (e.g., B and D). Similarly, we can state $r = b - d$ where b and d are per capita instanteous rates. Such an advance in understanding the determinants would be great.

Perhaps now is a good time to think about all the assumptions others might tell us we are making when we present the above formulation. Are all individuals in the population equally likely to produce progeny and/or die? Will birth and death rates vary over time or as the size of the population changes more? How will resource supply rate influence these rates? Is there really no immigration or emigration? These are very interesting questions.

Simple density-independent growth provides, in some sense, a null hypothesis for the dynamic behavior of a population. Malthus warned us that organisms produce more progeny than merely replacement value, and population growth is an exponential (or geometric) process [125]. The question then becomes "What causes population growth to differ from a constant rate of change?" That, in a nutshell, is what the rest of the book is about.

1.6 Modeling with Data: Simulated Dynamics

The main purpose of this section[6] is to begin to understand the mechanics of simulation. The preceding sections (the bulk of the chapter) emphasized understanding the deterministic underpinnings of simple forms of density independent growth: geometric and exponential growth. This section explores the simulation of density independent growth.

When we model populations, perhaps to predict the size of a population in the future, we can take a variety of approaches. One type of approach emphasizes deterministic prediction, using, for instance, $\bar{R}$. Another approach is to *simulate* population dynamics and we take this up in this next section.

To project population growth into the future should include some quantification of the uncertainty with our guess. Simulation is one way we can project

[6] This section emphasizes work in R.

populations and quantify the uncertainty. The way one often does that is to use the original data and sample it randomly to calculate model parameters. In this fashion, the simulations are random, but based on our best available knowldge, i.e., the real data. The re-use of observed data occurs in many guises, and it is known generally as bootstrapping or resampling.

1.6.1 Data-based approaches

In using our data to predict population sizes, let us think about three levels of biological organization and mechanism: population counts, changes in population counts, and individual birth and death probabilities. First, our count data alone provide a sample of a very large number of different possible counts. If we assume that there will be no trend over time, then a simple description of our observed counts (e.g., mean and confidence intervals) provide all we need. We can say "Song Sparrow counts in the Breeding Bird Survey in Darrtown, OH, are about 51."

Second, we could use the observed *changes* in population counts $R_t = N_{t+1}/N_t$ as our data. We would then draw an R_t at random from among the many observed values, and project the population one year forward. We then repeat this into the future, say, for ten years. Each simulation of a ten year period will result in a different ten year trajectory because we draw R_t at random from among the observed R_t. However, if we do many such simulations, we will have a *distribution* of outcomes that we can describe with simple statistics (e.g., median, mean, quantiles).

Third, we might be able to estimate the individual probabilities of births and deaths in the entire Darrtown population, and use those probabilities and birth rates to simulate the entire population into the future. In such an *individual-based* simulation, we would simulate the fates of individuals, keeping track of all individual births and deaths.

There are myriad others approaches, but these give you a taste of what might be possible. In this section we focus on the second of these alternatives, in which we use observed R_t to simulate the dynamics of Song Sparrow counts.

Here we investigate Song Sparrow (*Melospize melodia*) dynamics using data from the annual U.S. Breeding Bird Survey (http://www.mbr-pwrc.usgs.gov/bbs/). Below we will

1. look at and collecting the data (annual R's),
2. simulate one projection,
3. scale up to multiple simulations,
4. simplify simulations and perform 1000's, and
5. analyze the output.

1.6.2 Looking at and collecting the data

Let's start by looking at the data. *Looking at the data is always a good idea — it is a principle of working with data.* We first load the data from the `PET`

R package, and look at the names of the data frame. We then choose to `attach` the data frame, because it makes the code easier to read[7].

```
> names(sparrows)

[1] "Year"           "Count"          "ObserverNumber"

> attach(sparrows)
```

Now we plot these counts through time (Fig. 1.8).

```
> plot(Count ~ Year, type = "b")
```

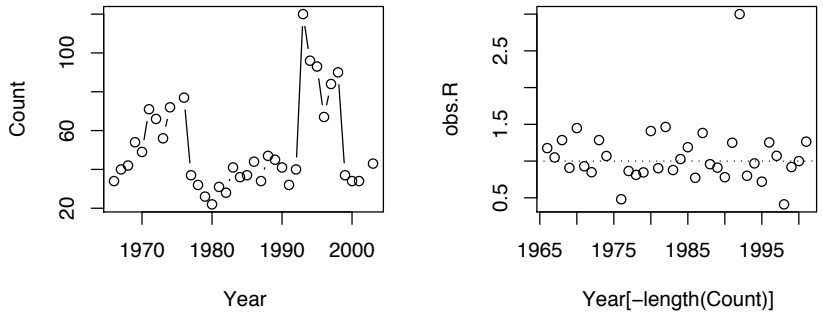


Fig. 1.8: Observations of Song Sparrows in Darrtown, OH (http://www.mbr-pwrc.usgs.gov/bbs/).

We see that Song Sparrow counts[8] at this site (the DARRTOWN transect, OH, USA) fluctuated a fair bit between 1966 and 2003. They never were completely absent and never exceeded ~ 120 individuals.

Next we calculate annual $R_t = N_{t+1}/N_t$, that is, the observed growth rate for each year t[9].

```
> obs.R <- Count[-1]/Count[-length(Count)]
```

Thus our *data* are the observed R_t, not the counts *per se*. These R form the basis of everything else we do. Because they are so important, let's plot these as well. Let's also indicate $R = 1$ with a horizontal dotted line as a visual cue

[7] I typically do not use `attach` but rather choose to always define explicitly the parent data frame I am using. It helps me reduce silly mistakes.

[8] Recall that these are *samples* or observations of sparrows. These are *not* population sizes. Therefore, we will be simulating sparrows counts, not sparrow population sizes.

[9] The use of "`-`" in the index tells Rto exclude that element (e.g., `-1` means "exclude the first element of the vector").

for zero population growth. Note that we exclude the last year because each R_t is associated with N_t rather than N_{t+1}.

```
> plot(obs.R ~ Year[-length(Count)])
> abline(h = 1, lty = 3)
```

One thing that emerges in our graphic data display (Fig. 1.8) is we have an unusually high growth rate in the early 1990's, with the rest of the data clustered around 0.5–1.5. We may want to remember that.

1.6.3 One simulation

Now that we have our randomly drawn Rs, we are ready to simulate dynamics. A key assumption we will make is that *these R are representative of R in the future, and that each is equally likely to occur*. We then *resample* these observed R *with replacement* for each year of the simulation. This random draw of observed R's then determines one random realization of a possible population trajectory. Let's begin.

First, we decide how many years we want to simulate growth.

```
> years <- 50
```

This will result in 51 population sizes, because we have the starting year, year 0, and the last year.

Next we draw 50 R at random with replacement. This is just like having all 35 observed R written down on slips of paper and dropped into a paper bag. We then draw one slip of paper out of the bag, write the number down, and put the slip of paper back in the bag, and then repeat this 49 more times. This is *resampling with replacement*[10]. The R function `sample` will do this. Because this is a pseudorandom[11] process, we use `set.seed` to make your process the same as mine, i.e., repeatable.

```
> set.seed(3)
> sim.Rs <- sample(x = obs.R, size = years, replace = TRUE)
```

Now that we have these 50 R, all we have to do is use them to determine the population size through time. For this, we need to use what programmers call a *for-loop* (see B.6 for further details). In brief, a for-loop repeatedly *loops* through a particular process, with one loop *for* each value of some indicator variable. Here we calculate each sparrow count in the next year, N_{t+1}, using the count in the current year N_t and the randomly drawn R *for* each year t.

---

[10] We could resample *without* replacemnt. In that case, we would be assuming that all of these R_t are important and *will* occur at some point, but we just don't know when — they constitute the entire universe of possiblities. Sampling *with* replacement, as we do above, assumes that the observed R_t are all equally likely, but none is particularly important — they are just a sample of what is possible, and they might be observed again, or they might not.

[11] A *pseudorandom* process is the best computers can do — it is a complex deterministic process that generates results that are indistinguishable from random.

We begin by creating an empty output vector that is the right length to hold our projection, which will be the number of Rs plus one[12].

```
> output <- numeric(years + 1)
```

We want to start the projection with the sparrow count we had in the last year (the "maximum," or biggest, year) of our census data.

```
> output[1] <- Count[Year == max(Year)]
```

Now the fun really starts to take off, as we finally use the for-loop. For each year t, we multiply N_t by the randomly selected R_t to get N_{t+1} and put it into the $t+1$ element of `output`.

```
> for (t in 1:years) output[t + 1] <- {
+     output[t] * sim.Rs[t]
+ }
```

Let's graph the result.

```
> plot(0:years, output, type = "l")
```

It appears to work (Fig. 1.9a) — at least it did something! Let's review what we have done. We

- had a bird count each year for 36 years. From this we calculated 35 R (for all years except the very last).
- decided how many years we wanted to project the population (50 y).
- drew *at random and with replacement* the observed R — one R for each year we want to project.
- got ready to do a simulation with a for-loop — we created an empty vector and put in an initial value (the last year's real data).
- performed each year's calculation, and put it into the vector we made.

So what does Fig. 1.9a represent? It represents one possible outcome of a trajectory, if we assume that R has an equal probability of being any of the observed R_t. This *particular* trajectory is very unlikely, because it would require one particular sequence of Rs. However, our simulation assumes that it is *no less likely* than any other particular trajectory.

As only one realization of a set of randomly selected R, Fig. 1.9a tells us very little. What we need to do now is to replicate this process a very large number of times, and examine the *distribution* of outcomes, including moments of the distribution such as the mean, median, and confidence interval of eventual outcomes.

1.6.4 Multiple simulations

Now we create a way to perform the above simulation several times. There are a couple tricks we use to do this. We still want to start small so we can figure out the steps as we go. Here is what we would do next.

[12] Remember that we always have one more population count than we do R_t.

- We start by specifying that we want to do 10 simulations, where one simulation is what we did above.
- We will need to use $50 \times 10 = 500$ randomly drawn Rs and store those in a matrix.
- To do the ten separate, independent simulations, we will use `sapply`, to "apply" our simulations ten times. We have to use a for-loop for each population simulation, because each N_t depends on the previous N_{t-1}. We use `sapply` and related functions for when we want to do more than one independent operation.

Here we specify 10 simulations, create a matrix of the 10×50 randomly drawn R.

```
> sims = 10
> sim.RM <- matrix(sample(obs.R, sims * years, replace = TRUE),
+     nrow = years, ncol = sims)
```

Next we get ready to do the simulations. First, to hold each projection temporarily, we will reuse `output` as many times as required. We then *apply* our for-loop projection as many times as desired, for each value of `1:sims`.

```
> output[1] <- Count[Year == max(Year)]
> outmat <- sapply(1:sims, function(i) {
+     for (t in 1:years) output[t + 1] <- output[t] * sim.RM[t,
+         i]
+     output
+ })
```

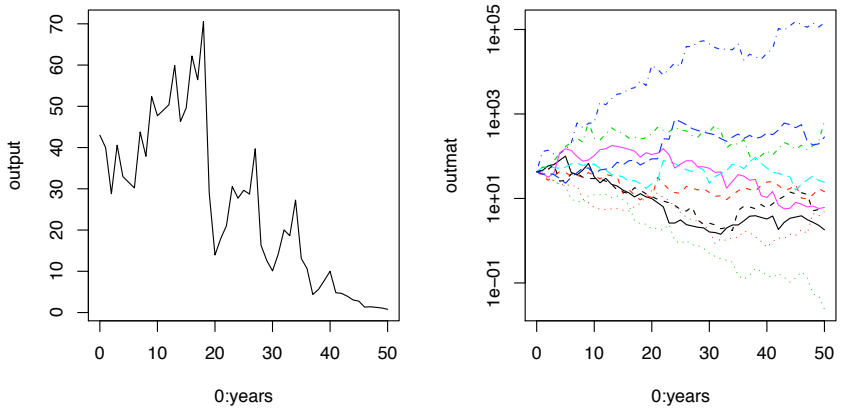
Now let's peek at the results (Fig. 1.9b). This is fun, but also makes sure we are not making a heinous mistake in our code. Note we use log scale to help us see the small populations.

```
> matplot(0:years, outmat, type = "l", log = "y")
```

What does it mean that the simulation has an approximately even distribution of final population sizes *on the log scale* (Fig. 1.9b)? If we plotted it on a linear scale, what would it look like?[13]

Rerunning this simulation, with new R each time, will show different dynamics every time, and that is the point of simulations. Simulations are a way to make a few key assumptions, and then leave the rest to chance. In that sense it is a null model of population dynamics.

[13] Plotting it on the log scale reveals that the relative change is independent of population size; this is true because the rate of change is geometric. If we plotted it on a linear scale, we would see that many trajectories result in small counts, and only a few get really big. That is, the median size is pretty small, but a few populations get huge.



(a) A single simulation (b) Ten simulations

Fig. 1.9: Simulated population dynamics with R drawn randomly from observed Song Sparrow counts.

1.6.5 Many simulations, with a function

Let's turn our simulation into a user-defined function[14] that simplifies our lives. We also add another assumption: individuals are irreducible. Therefore, let us use `round(,0)` to round to zero decimal places, i.e., the nearest integer[15].

Our user defined function, `PopSim`, simply wraps the previous steps up in a single function[16]. The output is a matrix, like the one we plotted above.

```
> PopSim <- function(Rs, N0, years = 50, sims = 10) {
+     sim.RM = matrix(sample(Rs, size = sims * years, replace = TRUE),
+         nrow = years, ncol = sims)
+     output <- numeric(years + 1)
+     output[1] <- N0
+     outmat <- sapply(1:sims, function(i) {
+         for (t in 1:years) output[t + 1] <- round(output[t] *
+             sim.RM[t, i], 0)
+         output
+     })
+     return(outmat)
+ }
```

If you like, try to figure out what each step of the simulation is doing. Consider it one of the end-of-chapter problems. Rely on on the code above to help you decipher the function.

[14] For user-defined functions, see sec. B.4.1.

[15] We could use also use `floor` to round down to the lowest integer, or `ceiling` to round up.

[16] This process, of working through the steps one at a time, and then wrapping up the steps into a function, is a useful work flow.

Now we have the pleasure of using this population simulator to examine a number of things, including the sizes of the populations after 50 years. I first simulate 1000 populations[17], and use `system.time` to tell me how long it takes on my computer.

```
> system.time(output <- PopSim(Rs = obs.R, N0 = 43, sims = 1000))

   user  system elapsed
  0.331   0.004   0.335
```

This tells me that it took less than half a second to complete 1000 simulations. That helps me understand how long 100 000 simulations might take. We also check the dimensions of the output, and they make sense.

```
> dim(output)

[1]   51 1000
```

We see that we have an object that is the size we think it should be. We shall assume that everything worked way we think it should.

1.6.6 Analyzing results

We extract the last year of the simulations (last row), and summarize it.

```
> N.2053 <- output[51, ]
> summary(N.2053, digits = 6)

    Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
     0.0     14.0     66.0   1124.6    291.8 332236.0
```

We see from this summary that the median final population size, among the 1000 simulations, is 66 individuals (median=50% quantile). While at least one of the populations has become extinct (min. = 0), the maximum is huge (max. = 332236). The `quantile` function allows us to find a form of empirical confidence intervals, including, approximately, the central 95% of the observations.[18]

```
> quantile(N.2053, prob = c(0.0275, 0.975))

2.75% 97.5%
    0  5967
```

These quantiles suggest that in 2053, we might observe sparrow counts anywhere from 0 to 5967, where zero and ~ 6000 are equally likely.

Notice the huge difference between the mean, $N = 1125$, and the median, N=66. Try to picture a histogram for which this would be true. It would be skewed right (long right hand tail), as with the lognormal distribution; this is common in ecological data.

---

[17] If we were doing this in a serious manner, we might do 10–100 000 times.

[18] Note that there are many ways to estimate quantiles (R has nine ways), but they are approximately similar to percentiles.

Let's make a histogram of these data. Exponentially growing populations, like this one, frequently show a lognormal distribution of abundances. Indeed, some say the "natural" unit of a population is $log(N)$, rather than N. We will plot two frequency distributions of the final populations, one on the orignal scale, one using the logarithms of the final population sizes plus 1 (we use $N+1$ so that we can include 0's — what is $log(0)$? $log(1)$?).

```
> hist(N.2053, main = "N")
> hist(log10(N.2053 + 1), main = "log(N+1)")
> abline(v = log10(quantile(N.2053, prob = c(0.0275, 0.975)) +
+     1), lty = 3)
```

We added some reference lines on the second histogram, showing the 2.5 and 97.5% quantiles (Fig. 1.10). You can see that the logarithms of the population sizes are much more well-behaved, more symmetrical.

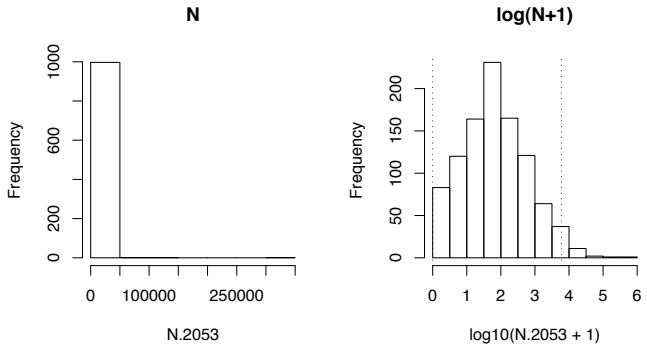


Fig. 1.10: Exploratory graphs of the distributions of the final simulated population sizes.

Can we really believe this output? To what can we compare our output? One thing that occurs to me is to compare it to the lower and upper bounds that we might contrive from *deterministic* projections.

To compare the simulation to deterministic projections, we could find the 95% t-distribution based confidence limits for the geometric mean of R. If we use our rules regarding the geometric mean, we would find that the logarithm of the *geometric* mean of R is the arthmetic mean of the $\log R$. So, one thing we could do is calculate the t-based confidence limits[19] on $\log R$, backtransform these to project the population out to 2053 with lower and upper bounds.

Here we take find the logarithms, caculate degrees of freedom and the relevant quantiles for the t distribution.

[19] Remember: the t-distribution needs the degrees of freedom, and a 95% confidence region goes from the 2.5% and the 97.5% percentiles.

```
> logOR <- log(obs.R)
> n <- length(logOR)
> t.quantiles <- qt(c(0.025, 0.975), df = n - 1)
```

Next we calculate the standard error of the mean, and the 95% confidence limits for $\log R$.

```
> se <- sqrt(var(logOR)/n)
> CLs95 <- mean(logOR) + t.quantiles * se
```

We backtransform to get R, and get a vector of length 2.

```
> R.limits <- exp(CLs95)
> R.limits
```

```
[1] 0.8968 1.1302
```

What do we see immediately about these values? One is less than 0, and one is greater than 0. This means that for the lower limit, the population will shrink (geometrically), while for the upper limit, the population will increase (geometrically). Let's go ahead and make the 50 y projection.

```
> N.Final.95 <- Count[Year == max(Year)] * R.limits^50
> round(N.Final.95)
```

```
[1]     0 19528
```

Here we see that the lower bound for the deterministic projection is the same (extinction) as the simulation, while the upper bound is much greater than that for the simulation. Why would that be? Perhaps we should examine the assumptions of our deteministic approach.

We started by assuming that the $\log R$ could be approximated with the t distribution, one of the most pervasive distributions in statistics and life. Let's check that assumption. We will compare the $\log R$ to the theoretical values for a t distribution. We scale `logOR` to make the comparison more clear.

```
> qqplot(qt(ppoints(n), df = n - 1), scale(logOR))
> qqline(scale(logOR))
```

How do we interpret these results? If the distribution of an observed variable is consistent with a particular theoretical distribution, the ordered quantiles of data will be a linear (straight line) function of the theoretical quantiles of the theoretical distribution. Deviations from that straight line illustrate how the data deviate. Here we see that the data have three outliers that are much more extreme (greater and smaller) than expected in the t-distribution, and more data are cluster around the center of the distribution than we would expect. We should ask whether those extreme values are mistakes in data collection or recording or whether they are every bit as accurate as all the other measurements.

Compare our two different confidence limits. These provide two different answers to our original question, "what might be the Song Sparrow count at this site in 2053?" Both of these assume a similar underlying model, density

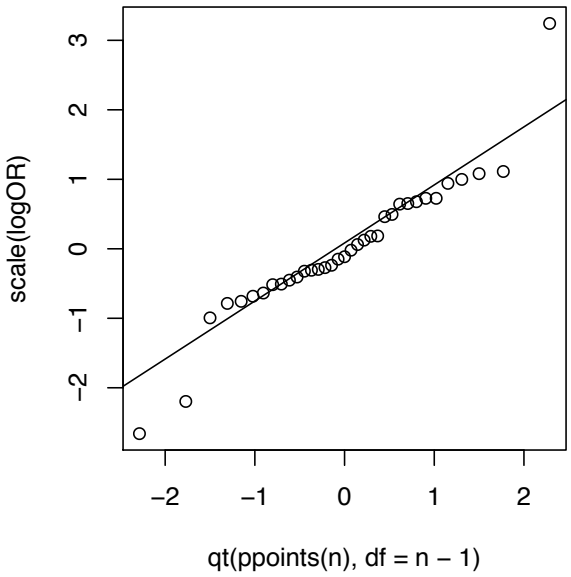


Fig. 1.11: Quantile-quantile plot used to compare $\log R$ to a t-distribution. Scaling `logOR` in this case means that we subtracted the mean and divided by the standard deviation. A histogram performs a similar service but is generally less discriminating and informative.

independent growth, but give different answers. Of which approach are we more confident? Why? What assumptions does each make?

We can be quite sure that our assumption regarding the t-distribution of our R is unsupported — our data have outliers, relative to a t-distribution. What would this do? It would increase the variance of our presumed distribution, and lead to wider confidence intervals, even though most of the data conform to a narrower distribution. Our simulation procedure, on the other hand, rarely samples those extreme points and, by chance, samples observed R that fall much closer to the median. This can occasionally be a problem in simulations based on too little data — the data themselves do not contain enough variability. Imagine the absurdity of a data-based simulation that relies on one observation — it would be *very* precise (but wrong)!

Our conclusions are based on a model of discrete density-independent population growth — what assumptions are we making? are they valid? Are our unrealistic assumptions perhaps nonetheless a good approximation of reality? We will revisit these data later in the book (Chapter 3) to examine one of these assumptions. We do not need to answer these questions now, but it is essential, and fun, to speculate.

1.7 Summary

In this chapter, we have explored the meaning of density-independent population growth. It is a statistically demonstrable phenomenon, wherein the per captia growth rate exhibits no relation with population density. It is a useful starting point for conceptualizing population growth. We have derived discrete geometric and continuous exponential growth and seen how they are related. We have caculated doubling times. We have discussed the assumptions that different people might make regarding these growth models. Last, we have used simulation to explore prediction and inference in a density-independent context.

Problems

1.1. Geometric growth Analyze the following data, relying on selected snippets of previous code.
(a) In the years 1996 through 2005, lily population sizes are N = 150, 100, 125, 200, 225, 150, 100, 175, 100, 150. Make a graph of population size versus time.
(b) Calculate R for each year; graph R *vs.* time.
(c) Calculate arithmetic and geometric average growth rates of this population.
(d) Based on the appropriate average growth rate, what would be the expected population size in 2025? What would the estimated population size be if you used the inappropriate mean? Do not use simulation for this.
(d*) Given these data, develop simulations as above with the user-defined function, `PopSim`. Describe the distribution of projected population sizes for 2010.

1.2. Doubling Time
(a) Derive the formula for doubling time in a population with contiunous exponential growth.
(b) What is the formula for tripling time?
(c) If we are modeling humans or *E. coli*, would a model of geometric, or exponential growth be better? Why?
(d) If an *E. coli* population grew from 1000 cells to 2×10^9 cells in 6 h, what would its intrinsic rate of increase be? Its doubling time?

1.3. Human Population Growth
(a) There were about 630 million people on the planet in 1700, and 6.3 billion in 2003 [33]. What was the intrinsic rate of increase, r?
(b) Graph the model of human population size population size from 1700 to 2020.
(c) Add points on the graph indicating the population doublings from 1700 onward.
(d*) What will prevent humans from outweighing the planet by the end of this century? What controls human population growth? Do these controls vary spatially across the planet? See Cohen [33] to get going.

1.4. R functions
Find the R functions in Chapter 1. Demonstrate their uses.

4

Populations in Space

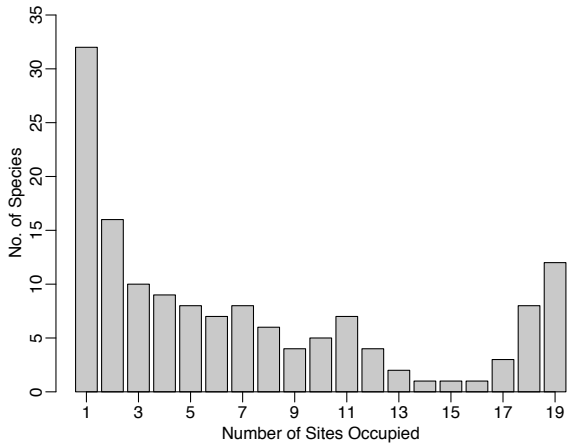


Fig. 4.1: A frequency distribution of the number of plant species (y-axis) that occupy different numbers of grassland remnants (x-axis). Note the U-shaped (bimodal) distribution of the number of sites occupied. Other years were similar [35]

Over relatively large spatial scales, it is not unusual to have many species that seem to occur everywhere, and even more species that seem to be found in only one or a few locations. For example, Scott Collins and Susan Glenn [35] showed that in grasslands, each separated by up to 4 km, there were more species occupying only one site (Fig. 4.1, left-most bar) than two or more sites, and also that there are more species occupying all the sites than most intermediate numbers of sites (Fig. 4.1, right-most bar), resulting in a U-shaped frequency distribution. Illke Hanski [70] coined the rare and common species "satellite" and "core" species, respectively, and proposed an explanation. Part of the answer seems to come from *the effects of immigration and emigration in a spatial context*. In this chapter we explore mathematical representations of individuals

and populations that exist in space, and we investigate the consequences for populations and collections of populations.

4.1 Source-sink Dynamics

In Chapters 1-3, we considered *closed* populations. In contrast, one could imagine a population governed by *b*irths plus *i*mmigration, and *d*eaths plus *e*migration (a *BIDE* model). Ron Pulliam [172] proposed a simple model that includes all four components of *BIDE* which provides a foundation for thinking about connected subpopulations. We refer to the dynamics of these as *source-sink dynamics*. Examples of such linked populations might include many different types of species. For instance, a source-sink model could describe linked populations of a single species might occupy habitat patches of different quality, where organisms might disperse from patch to patch.

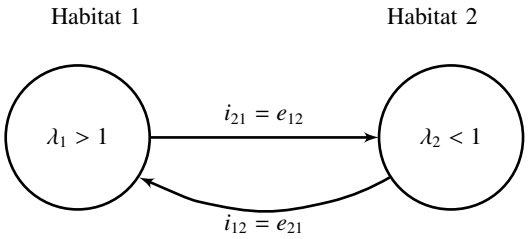


Fig. 4.2: The simplest source-sink model.

The concept

The general idea of source-sink populations begins with the idea that spatially separated subpopulations occupy distinct patches, and each exhibit their own intrinisic dynamics due to births and deaths; that is, we could characterize a λ for each subpopulation. In addition, individuals move from one patch to another; that is, they immigrate and emigrate from one patch (or subpopulation) to another. Therefore, the number of individuals we observe in a particular patch is due, not only to the λ in that population, but also to the amount of immigration, i, and emigration, e.

Subpopulations with more births than deaths, $\lambda > 1$, and with more emigration than immigration, $e > i$, are referred to as *source populations*. Subpopulations with fewer births than deaths, $\lambda < 1$, and with more immigration than emigration, $i > e$, are referred to as *sink populations*.

When we think about what might *cause* variation in λ, we typically refer to the *quality* of patches or habitats. Quality might be inferred from λ, or it might actually be the subject of investigation and independent of λ — typically we think of high quality habitat as having $\lambda > 1$ and poor quality habitat as having $\lambda < 1$.

The equations

Pulliam envisioned two linked bird populations where one could track adult reproduction, and adult and juvenile survival and estimate λ, per capita growth rate separately for each population. For the first population, the number of birds in patch 1 at time $t+1$, $n_{1,t+1}$, is the result of adult survival P_A, reproduction β_1, and survival of the juveniles P_J. Thus,

$$n_{1,t+1} = P_A n_t + P_J \beta_1 n_{1,t} = \lambda_1 n_1. \tag{4.1}$$

Here $\beta_1 n_{1,t}$ is production of juveniles, and P_J is the survival of those juveniles to time $t+1$. Pulliam described the second poopulation in the same fashion as

$$n_{2,t+1} = P_A n_t + P_J \beta_2 n_{1,t} = \lambda_2 n_2. \tag{4.2}$$

Pulliam then assumed, for simplicity's sake, that the two populations vary only in fecundity (β), which created differences in λ_1 and λ_2. He called population 1 the *source population* ($\lambda_1 > 1$) and population 2 the *sink population* ($\lambda_2 < 1$). He also assumed that birds in excess of the number of territories in the source population emigrated from the source habitat to the sink habitat. Therefore, the source population held a constant density (all territories filled), but the size of the population in the sink depended on both its own growth rate $\lambda_2 < 1$ and also the number of immigrants.

A result

One of his main theoretical findings was that *population density can be a misleading indicator of habitat quality* (Fig. 4.3). If we assume that excess individuals in the source migrate to the sink, then as habitat quality and reproduction increase in the source population, the source population comprises *an ever decreasing proportion* of the total population! That is, as λ_1 gets larger, $n_1/(n_1+n_2)$ gets smaller. Thus, density can be a very misleading predictor of long-term population viability, if the source population is both productive and exhibits a high degree of emigration.

A model

We can use a matrix model to investigate source-sink populations [12]. Let us mix up typical demographic notation (e.g., Chapter 2) with that of Pulliam [172], so that we can recognize Pulliam's quantities in a demographic matrix model setting. Further, let us assume a pre-breeding census, in which we count adults. The population dynamics would thus be governed by $\mathbf{A}$

$$\mathbf{A} = \begin{pmatrix} P_{A1} + P_{J1}\beta_1 & M_{12} \\ M_{21} & P_{A2} + P_{J2}\beta_2 \end{pmatrix} \tag{4.3}$$

where the upper left element (row 1, column 1) reflects the within-patch growth characteristics for patch 1. The lower right quadrant (row 2, and column 2) reflects the within-patch growth characteristics of patch 2.

We then assume, for simplicity, that migration, M, is exclusively from the source to the sink ($M_{21} > 0$, $M_{12} = 0$). We further assume that $\lambda_1 > 1$ but all excess individuals migrate to patch 2, so $M_{21} = \lambda_1 - 1 > 0$. Then $\mathbf{A}$ simplifies to

$$\mathbf{A} = \begin{pmatrix} 1 & 0 \\ \lambda_1 - 1 & \lambda_2 \end{pmatrix} \tag{4.4}$$

The spatial demographic Pulliam-like model

We first assign λ for the source and sink populations, and create a matrix.

```
> L1 <- 2; L2 <- 0.4
> A <- matrix(c(1, 0, L1 - 1, L2), nrow = 2, byrow = TRUE)
```

We can then use eigenanalysis, as we did in Chapter 2 for stage structured populations. The dominant eigenvalue will provide the long term asymptotic total population growth. We can calculate the stable "stage" distribution, which in this case is the distribution of individuals between the two habitats.

```
> eigen(A)

$values
[1] 1.0 0.4

$vectors
       [,1] [,2]
[1,] 0.5145    0
[2,] 0.8575    1
```

From the dominant eigenvalue, we see Pulliam's working assumption that the total population growth is set at $\lambda = 1$. We also see from the dominant eigenvector that the sink population actually contains more individuals than the source population ($0.51/(0.51+0.86) < 0.5$).
We could graph these results as well, for a range of λ_1. Here we let `p1` be the proportion of the population in the source.

```
> L1s <- seq(1, 3, by = 0.01)
> p1 <- sapply(L1s, function(l1) {
+     A[2, 1] <- l1 - 1
+     eigen(A)$vectors[1, 1]/sum(eigen(A)$vectors[, 1])
+ })
> plot(L1s, p1, type = "l", ylab = "Source Population",
+      xlab = expression(lambda[1]))
```

4.2 Two Types of Metapopulations

Our logistic model (Chapter 3) is all well and good, but it has no concept of *space* built into it. In many, and perhaps most circumstances in ecology, space has the potential to influence the dynamics of populations and ecosystem fluxes [101, 102, 116]. The logistic equation represents a *closed* population, with

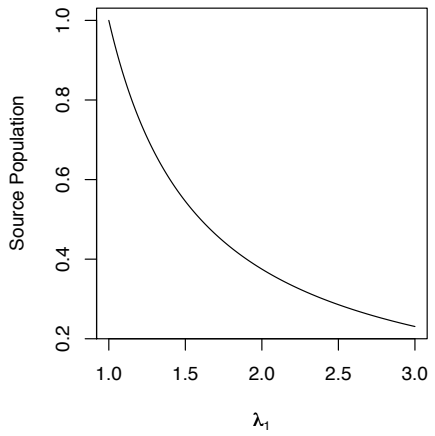


Fig. 4.3: The declining *relative* abundance in the high quality habitat in a source-sink model. The proportion of the total population $(n_1/(n_1+n_2))$ in the *source* population may decline with increasing habitat quality and growth rate λ_1 habitat.

no clear accounting for emigration or immigration. In particular cases, however, consideration of space may be essential. What will we learn if we start considering space, such that sites are open to receive immigrants and lose emigrants?

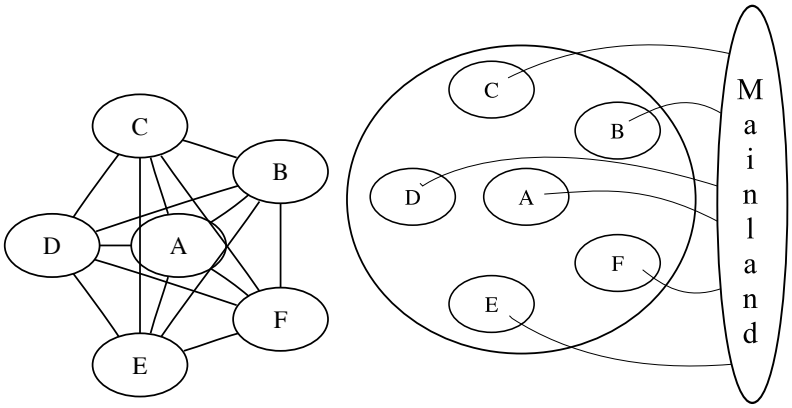
First we consider ideas associated with different types of "collections;" we then consider a mathematical framework for these ideas.

A single spatially structured population

One conceptual framework that we will consider below is that of a single closed population, where individuals occupy sites in an implicitly spatial context (Fig. 4.4). Consider a population in space, where a *site* is the space occupied by one individual. One example might be grasses and weeds in a field. In such a population, for an individual within our population to successfully reproduce and add progeny to the population, the individual must first actually occupy a site. For progeny to establish, however, a propagule must arrive at a site that is unoccupied. Thus the more sites that are already occupied, the less chance there is that a propagule lands on an unoccupied site. Sites only open up at some constant per capita rate as individuals die at a per capita death rate.

A metapopulation

The other conceptual framework that we consider here is that of *metapopulations*. A metapopulation is a population of populations, or a collection of populations (Fig. 4.4). Modeling metapopulations emerged from work in pest management when Levins [110] wanted to represent the dynamics of the proportion of fields infested by a pest. He assumed that a field was either occupied by the pest, or not. The same models used to represent a population of individuals that occupy sites (above) can also be used to represent populations that occupy



(a) A closed collection (b) An open collection

Fig. 4.4: Collections of sites. (a) Sites may be recolonized via internal propagule production and dispersal only, or (b) sites may receive immigrants from an outside source that is not influenced by the collection. Each site (A-F) may be a spot of ground potentially occupied by a single plant, or it may be an oceanic island potentially occupied by a butterfly population. Sites may also be colonized via both internal and external sources.

sites, with conceptually similar ecological interpretation. In this case, each *site* is a location that either contains a population or not. In this framework, we keep track of the proportion of all populations that remain extant, that is, the proportion of sites that are occupied. As with a single population (above), the metapopulation is closed, in the sense that there exists a finite number of sites which may exchange migrants.

Whether we consider a single spatial population, or single metapopulation, we can envision a *collection of sites* connected by dispersal. Each site may be a small spot of ground that is occupied by a plant, or it may be an oceanic island that is occupied by a population. All we know about a single site is that it is occupied or unoccupied. If the site is occupied by an individual, we know nothing of how big that individual is; if the site is occupied by a population, we know nothing about how many indiviuals are present. The models we derive below keep track of the proportion of sites that are occupied. These are known loosely as *metapopulation models*. Although some details can differ, whether we are modeling a collection of spatially discrete individuals in single population or a collection of spatially discrete populations, these two cases share the idea that there are a collection of sites connected by migration, and each is subject to extinction.

The most relevant underlying biology concerns colonization and extinction in our collection of sites (Fig. 4.4). In this chapter, we will assume that all sites experience equal rates; when we make this assumption, we greatly simplify everything, and we can generalize across all sites. All of the models we con-

sider are simple elaborations of what determines colonization and extinction. Another useful concept to consider is whether the collection of sites receives propagules from the outside, from some external source that is not influenced by the collection of sites (Fig. 4.4).

4.3 Related Models

Here we derive a single mathematical framework to describe our two types of models. In all cases, we will consider how total rates of colonization, C, and extinction, E, influence the the rate of change of p, the proportion of sites that are occupied,

$$\frac{\mathrm{d}p}{\mathrm{d}t} = C - E. \tag{4.5}$$

We will consider below, in a somewhat orderly fashion, several permutations of how we represent colonization and extinction of sites (e.g., [62, 63]).

4.3.1 The classic Levins model

Levins [110] proposed what has come to be known as the classic metapopulation model,

$$\frac{\mathrm{d}p}{\mathrm{d}t} = c_i p\,(1 - p) - ep. \tag{4.6}$$

This equation describes the dynamics of the proportion, p, of a set of fields invaded by a pest (Fig. 4.5a). The pest colonizes different fields at a total rate governed by the rate of propagule production, c_i, and also on the proportion of patches that contain the pest, p. Thus, propagules are being scattered around the landscape at rate $c_i p$. The rate at which p changes, however, is also related to the proportion of fields that are unoccupied, $(1 - p)$, and therefore available to become occupied and increase p. Therefore the total rate of colonization is $c_i p(1-p)$. The pest has a constant local extinction rate e, so the total extinction rate in the landscape is ep.

The parameters c_i and e are very similar to r of continuous logistic growth, insofar as they are dimensionless instantaneous rates. However, they are sometimes thought of as probabilities. The parameter c_i is approximately the proportion of open sites colonized per unit time. For instance, if we created or found 100 open sites, we could come back in a year and see how many became occupied over that time interval of one year, and that proportion would be a function of c_i. The parameter e is often thought of as the probability that a site becomes unoccupied per unit time. If we found 100 occupied sites in one year, we could revisit them a year later and see how many became *un*occupied over that time interval of one year.

We use the subscript i to remind us that the colonization is coming from within the sites that we are studying (i.e. internal colonization). With internal colonization, we are modeling a closed spatial population of sites, whether "site"

refers to an entire field (as above), or a small patch of ground occupied by an individual plant [202].

The Levins metapopulation model (Fig. 4.5a)

A function for a differential equation requires arguments for time, a vector of the state variables (here we have one state variable, p), and a vector of parameters.

```
> levins <- function(t, y, parms) {
+     p <- y[1]
+     with(as.list(parms), {
+         dp <- ci * p * (1 - p) - e * p
+         return(list(dp))
+     })
+ }
```

By using `with`, we can specify the parameters by their names, as long as `parms` includes names. The function returns a list that contains a value for the derivative, evaluated at each time point, for each state variable (here merely $\mathrm{d}p/\mathrm{d}t$). We then use `levins` in the numerical integration function `ode` in the `deSolve` package.

```
> library(deSolve)
> prms <- c(ci = 0.15, e = 0.05); Initial.p <- 0.01
> out.L <- data.frame(ode(y = Initial.p, times = 1:100, func = levins,
+     parms = prms))
```

We then plot the result (Fig. 4.5a).

```
> plot(out.L[, 2] ~ out.L[, 1], type = "l", ylim = c(0, 1),
+     ylab = "p", xlab = "time")
```

Can we use this model to predict the eventual equilibrium? Sure — we just set eq. 4.6 to zero and solve for p. This model achieves and equilibrium at,

$$0 = c_i p - c_i p^2 - ep$$
$$p^* = \frac{c_i - e}{c_i} = 1 - \frac{e}{c_i}.$$

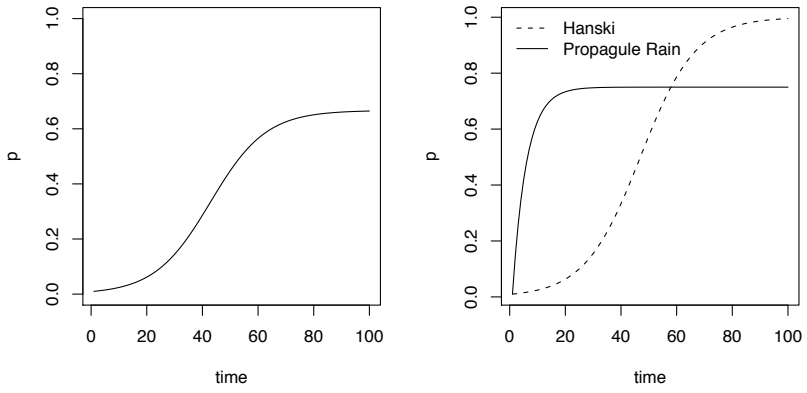
When we do this, we see that $p^* > 0$ as long as $c_i > e$ (e.g., Fig. 4.5a). When is $p^* = 1$, so that all the sites are filled? In principle, all sites cannot be occupied simultaneously unless $e = 0$!

4.3.2 Propagule rain

From where else might propagules come? If a site is not closed off from the rest of the world, propagules could come from outside the collection of sites that we are actually monitoring.

For now, let us assume that our collection of sites is continually showered by propagules from an external source. If only those propagules are important, then we could represent the dynamics as,

$$\frac{\mathrm{d}p}{\mathrm{d}t} = c_e(1 - p) - ep \tag{4.7}$$



(a) Levins (b) Propagule Rain and Hanski

Fig. 4.5: Three metapopulation models, using similar parameters ($c_i = 0.15$, $c_e = 0.15$, $e = 0.05$).

where c_e specifies rate of colonization coming from the external source. Gotelli [63] refers to this model as a metapopulation model with "propagule rain" or the "island–mainland" model. He calls it this because it describes a constant influx of propagules which does not depend on the proportion, p, of sites occupied for propagule production. Extinction here is mediated only by the proportion of sites occupied, and has a constant per site rate.

The propagule rain metapopulation model (Fig. 4.5b)

A function for a differential equation requires arguments for time, a vector of the state variables (here we have one state variable, p), and a vector of parameters.

```
> gotelli <- function(t, y, parms) {
+     p <- y[1]
+     with(as.list(parms), {
+         dp <- ce * (1 - p) - e * p
+         return(list(dp))
+     })
+ }
```

The function returns a list that contains a value for the derivative, evaluated at each time point, for each state variable (here merely $\mathrm{d}p/\mathrm{d}t$.

We can solve for this model's equilibrium by setting eq. 4.7 equal to zero.

$$0 = c_e - c_e p - ep \tag{4.8}$$

$$p^* = \frac{c_e}{c_e + e}. \tag{4.9}$$

Of course, we might also think that both internal and external sources are important, in which case we might want to include both sources in our model,

$$\frac{\mathrm{d}p}{\mathrm{d}t} = (c_i p + c_e)(1 - p) - ep \tag{4.10}$$

$$\tag{4.11}$$

As we have seen before, however, adding more parameters is not something we take lightly. Increasing the number of parameters by, in this case, 50% could require a lot more effort to estimate.

4.3.3 The rescue effect and the core-satellite model

Thus far, we have ignored what happens between census periods. Imagine that we sample site "A" each year on 1 January. It is possible that between 2 January and 31 December the population at site A becomes extinct and then is subsequently recolonized, or "rescued" from extinction. When we sample on 1 January in the next year, we have no way of knowing what has happened in the intervening time period. We would not realize that the population had become extinct and recolonization had occurred.

We can, however, model total extinction rate E with this *rescue effect*,

$$E = -ep(1 - p). \tag{4.12}$$

Note that as $p \to 1$, the total extinction rate approaches zero. Total extinction rate declines because as the proportion of sites occupied increases, it becomes increasingly likely that dispersing propagules will land on all sites. When propagules happen to land on sites that are on the verge of extinction, they can "rescue" that site from extinction.

Brown and Kodric-Brown [17] found that enhanced opportunity for immigration seemed to reduce extinction rates in arthropod communities on thistles. They coined this effect of immigration on extinction as the "rescue effect." MacArthur and Wilson [121] also discussed this idea in the context of island biogeography. We can even vary the strength of this effect by adding yet another parameter q, such that the total extinction rate is $ep(1 - qp)$ (see [62]).

Assuming only internal propagule supply and the simple rescue effect results in what is referred to as the the core-satellite model,

$$\frac{\mathrm{d}p}{\mathrm{d}t} = c_i p(1 - p) - ep(1 - p) \tag{4.13}$$

This model was made famous by Illka Hanski [70]. It is referred to as the *core-satellite* model, for reasons we explore later.

The core-satellite metapopulation model

A function for a differential equation requires arguments for time, a vector of the state variables (here we have one state variable, p), and a vector of parameters.

```
> hanski <- function(t, y, parms) {
+     p <- y[1]
+     with(as.list(parms), {
+         dp <- ci * p * (1 - p) - e * p * (1 - p)
+         return(list(dp))
+     })
+ }
```

The function returns a list that contains a value for the derivative, evaluated at each time point, for each state variable (here merely $\mathrm{d}p/\mathrm{d}t$).

Graphing propagule rain and core-satellite models (Fig. 4.5b)

First, we integrate the models using the same parameters as for the Levins model, and collect the results.

```
> prms <- c(ci <- 0.15, ce <- 0.15, e = 0.05)
> out.IMH <- data.frame(ode(y = Initial.p, times = 1:100,
+     func = gotelli, parms = prms))
> out.IMH[["pH"]] <- ode(y = Initial.p, times = 1:100, func = hanski,
+     parms = prms)[, 2]
```

We then plot the rsult (Fig. 4.5a).

```
> matplot(out.IMH[, 1], out.IMH[, 2:3], type = "l", col = 1,
+     ylab = "p", xlab = "time")
> legend("topleft", c("Hanski", "Propagule Rain"), lty = 2:1,
+     bty = "n")
```

Core-satellite equilibria

What is the equilibrium for the Hanski model (eq. 4.13)? We can rearrange this to further simplify solving for p^*.

$$\frac{\mathrm{d}p}{\mathrm{d}t} = (c_i - e)\, p\, (1 - p) \tag{4.14}$$

This shows us that for any value of p between zero and one, the sign of the growth rate (positive or negative) is determined by c_i and e. If $c_i > e$, the rate of increase will always be positive, and because occupancy cannot exceed 1.0, the metapopulation will go to full occupancy ($p^* = 1$), and stay there. This equilibrium will be a *stable attractor* or stable equilibrium. What happens if for some reason the metapopulation becomes globally extinct, such that $p = 0$, even though $c_i > e$? If $p = 0$, then like logistic growth, the metapopulation stops changing and cannot increase. However, the slightest perturbation away from $p = 0$ will lead to a positive growth rate, and increase toward the stable

attractor, $p^* = 1$. In this case, we refer to $p^* = 0$ as an *unstable equilibrium* and a repellor.

If $c_i < e$, the rate of increase will always be negative, and because occupancy cannot be less than 0, the metapopulation will become extinct ($p^* = 0$), and stay there. Thus $p^* = 0$ would be a stable equilibrium or attractor. What is predicted to happen if, for some odd reason this population achieved full occupancy, $p = 1$, even though $c_i < e$? In that case, $(1 - p) = 0$, and the rate of change goes to zero, and the population is predicted to stay there, even though extinction is greater than colonization. How weird is that? Is this fatal flaw in the model, or an interesting prediction resulting from a thorough examination of the model? How relevant is it? How could we evaluate how relevant it is? We will discuss this a little more below, when we discuss the effects of habitat destruction.

What happens when $c_i = e$? In that case, $c_i - e = 0$, and the population stops changing. What is the value of p when it stops changing? It seems as though it could be any value of p, because if $c_i - e = 0$, the rate change goes to zero. What will happen if the population gets perturbed — will it return to its previous value? Let's return to question in a bit.

To analyze stability in logistic growth, we examined the slope of the partial derivative at the equilibrium, and we can do that here. We find that the partial derivative of eq. 4.13 with respect to p is

$$\frac{\partial \dot{p}}{\partial p} = c - 2cp - e + 2ep \tag{4.15}$$

where $\dot{p}$ is the time derivative (eq. 4.13). A little puzzling and rearranging will show

$$\frac{\partial \dot{p}}{\partial p} = (c_i - e)(1 - 2p) \tag{4.16}$$

and make things simpler. Recall our rules with regard to stability (Chapter 3). If the partial derivative (the slope of the time derivative) is negative at an equilibrium, it means the the growth rate approaches zero following a perturbation, meaning that it is stable. If the partial derivative is positive, it means that the change accelerates away from zero following the perturbation, meaning that the equilibrium is unstable. So, we find the following guidelines:

- $c_i > e$
  - $p = 1$, $\partial \dot{p}/\partial p < 0$, stable equilibrium.
  - $p = 0$, $\partial \dot{p}/\partial p > 0$, unstable equilibrium.
- $c_i < e$
  - $p = 1$, $\partial \dot{p}/\partial p > 0$, unstable equilibrium.
  - $p = 0$, $\partial \dot{p}/\partial p < 0$, stable equilibrium.

What if $c_i = e$? In that case, both the time derivative ($\mathrm{d}p/\mathrm{d}t$) and the partial derivative ($\partial \dot{p}/\partial p$) are zero *for all values of* p. Therefore, if the population gets displaced from any arbitrary point, it will remain unchanged, not recover, and will stay displaced. We call this odd state of affairs a *neutral equilibrium*. We revisit neutral equilibrium when we discuss interspecific competition and predation.

We can also explore the stability of one of these equilibria by plotting the mettapopulation growth rate as a function of p (Fig. 4.6). When we set $c_i > e$, and examine the slope of that line at $p^* = 1$, we see the slope is negative, indicating a stable equilibrium.

An equilibrium for the core-satellite metapopulation model (Fig. 4.6)

We first create an expression for the growth itself, $\mathrm{d}p/\mathrm{d}t$. We then plot it, while we evaluate it, on the fly.

```
> dpdtCS <- expression((ci - e) * p * (1 - p))
> ci <- 0.15; e <- 0.05; p <- seq(0, 1, length = 50)

> plot(p, eval(dpdtCS), type = "l", ylab = "dp/dt")
```

Levins *vs.* Hanski

Why would we use Levins' model instead of Hanski's core-satellite model? To explore this possibility, let's see how the Hanski model might change gradually into the Levins model. First we define the Hanski model with an extra parameter, a,

$$\frac{\mathrm{d}p}{\mathrm{d}t} = c_i p\,(1-p) - e p\,(1-ap)\,. \tag{4.17}$$

Under Hanski's model, $a = 1$ and under Levins' model $a = 0$. If we solve for the equilibrium, we see that

$$p^* = \frac{c-e}{c-ae} \tag{4.18}$$

so that we can derive either result for the two models. In the context of logistic growth, where $K = Hp^*$, this result, eq. 4.18, implies that for the Hanski model, K fills all available habitat, whereas the Levins model implies that K fills only a fraction of the total available habitat. That fraction results from the dynamic balance between c_i and e.

4.4 Parallels with Logistic Growth

It may have already occurred to you that the closed spatial population described here sounds a lot like simple logistic growth. A closed contiguous population, spatial or not, reproduces in proportion to its density, and is limited by its own density. Here we will make the connection a little more clear. It turns out that a simple rearrangement of eq. 4.6 will provide the explicit connection between logistic growth and the spatial population model with internal colonization [181].

Imagine for a moment that you are an avid birder following a population of Song Sparrows in Darrtown, OH, USA (Fig. 3.1a). If Song Sparrows are limited by the number of territories, and males are competing for territories, then you could think about male Song Sparrows as "filling up" some proportion, p, of the available habitat. You have already described this population with

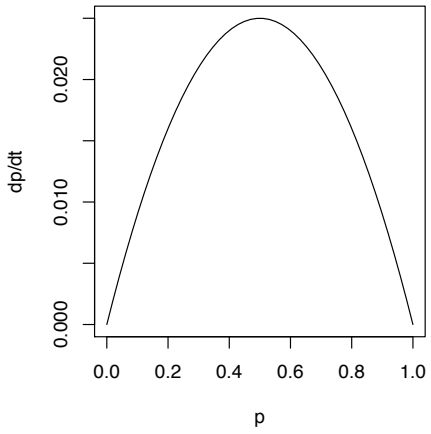


Fig. 4.6: Metapopulation growth rate as a function of p, in the core-satellite model ($c_i = 0.15$, $e = 0.05$). When we plot population growth rate for the core-satellite model, for arbitrary parameter values where $c_i > e$, we see that growth rate falls to zero at full occupancy (i.e., at $p^* = 1$). We also see that the slope is negative, indicating that this equilibrium is stable.

the logistic growth model ($\mathrm{d}N/\mathrm{d}t = rN(1 - \alpha N)$). Lately, however, you have been thinking about how territories, spatially arranged in the landscape, may limit this population. You therefore decide that you would like to use Levins' spatially-implicit metapopulation model instead (eq. 4.6). How will you do it? You do it by *rescaling* logistic growth.

Let us start by defining our logistic model variables in other terms. First we define N as

$$N = pH$$

where N is the number of males defending territories, H is the total number of possible territories, and p is the proportion of possible territories occupied at any one time. At equilibrium, $N^* = K = p^*H$, so $\alpha = 1/(p^*H)$. Recall that for the Levins model, $p^* = (c_i - e)/c_i$, so therefore,

$$\alpha = \frac{c_i}{(c_i - e)\,H}.$$

We now have N, α, and K in terms of p, H, c_i and e, so what about r? Recall that for logistic growth, the per capita growth rate goes to r as $N \to 0$ (Chapter 3). For the Levins metapopulation model, the per patch growth rate is

$$\frac{1}{p}\frac{\mathrm{d}p}{\mathrm{d}t} = c_i\,(1 - p) - e. \tag{4.19}$$

As $p \to 0$ this expression simplifies to $c_i - e$, which is equivalent to r. Summarizing, then, we have,

$$r = c_i - e \tag{4.20}$$

$$N = pH \tag{4.21}$$

$$\alpha = \frac{1}{K} = \frac{1}{p^* H} = \frac{c_i}{H\,(c_i - e)} \tag{4.22}$$

$$\tag{4.23}$$

Substituting into logistic growth ($\dot{N} = rN(1 - \alpha N)$), we now have

$$\frac{\mathrm{d}(pH)}{\mathrm{d}t} = (c_i - e)\,pH\left(1 - \frac{c_i}{H\,(c_i - e)}Hp\right) \tag{4.24}$$

$$= (c_i - e)\,pH - \frac{c_i - e}{c_i - e}c_i p^2 H \tag{4.25}$$

$$= H\,(c_i p\,(1 - p) - ep) \tag{4.26}$$

which is almost the Levins model. If we note that H is a constant, we realize that we can divide both sides by H, ending up with the Levins model eq. 4.6.

4.5 Habitat Destruction

Other researchers have investigated effects of habitat loss on metapopulation dynamics [88, 146, 202]. Taking inspiration from the work of Lande [95, 96], Karieva and Wennergren [88] modeled the effect of habitat destruction, D, on overall immigration probability. They incorporated this into Levins' model as

$$\frac{\mathrm{d}p}{\mathrm{d}t} = c_i p(1 - D - p) - ep \tag{4.27}$$

where D is the amount of habitat destroyed, expressed as a fraction of the original total available habitat.

Habitat destruction model

To turn eq. 4.27 into a function we can use with `ode`, we have,

```
> lande <- function(t, y, parms) {
+     p <- y[1]
+     with(as.list(parms), {
+         dp <- ci * p * (1 - D - p) - e * p
+         return(list(dp))
+     })
+ }
```

Habitat destruction, D, may vary between 0 (= Levins model) to complete habitat loss 1.0; obviously the most interesting results will come for intermediate values of D (Fig. 4.7).

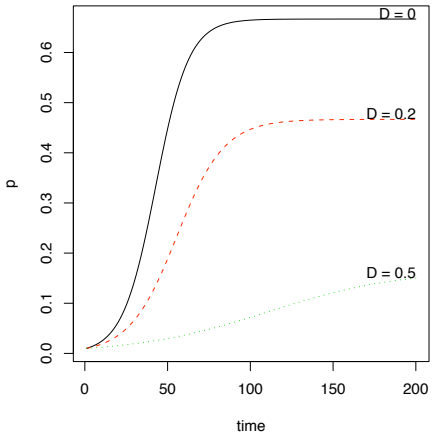


Fig. 4.7: Metapopulation dynamics, combining the Levins model and habitat destruction ($c_i = 0.15$, $e = 0.05$).

Illustrating the effects of habitat destruction (Fig. 4.7)

We can plot the dynamics for three levels of destruction, including none. We first set all the parameters, and time.

```
> library(deSolve)
> prmsD <- c(ci = 0.15, e = 0.05, D = 0)
> Ds <- c(0, 0.2, 0.5)
> Initial.p <- 0.01
> t <- 1:200
```

We then create an empty matrix of the right size to hold our results, and then integate the ODE.

```
> ps <- sapply(Ds, function(d) {
+     prmsD["D"] <- d
+     ode(y = Initial.p, times = t, func = lande, parms = prmsD)[,
+         2]
+ })
```

Last, we plot it and add some useful labels.

```
> matplot(t, ps, type = "l", ylab = "p", xlab = "time")
> text(c(200, 200, 200), ps[200, ], paste("D = ", Ds, sep = ""),
+     adj = c(1, 0))
```

What is the equilibrium under this model? Setting eq. 4.27 to zero, we can then solve for p.

$$0 = c_i - c_i D - c_i p - e \tag{4.28}$$

$$p^* = \frac{c_i - c_i D - e}{c_i} = 1 - \frac{e}{c_i} - D \tag{4.29}$$

Thus we see that habitat destruction has a simple direct effect on the metapopulation.

A core-satellite habitat loss scenario

Let us return now to that odd, but logical, possibility in the core-satellite model where $c_i < e$ and $p = 1$. Recall that in this case, $p = 1$ is an *unstable* equilibrium ($p = 0$ is the stable equilibrium for $c_i < e$). We discuss this in part for greater ecological understanding, but also to illustrate why theory is sometimes useful — because it helps us explore the logical consequences of our assumptions, even when, at first blush, it seems to make little sense.

Imagine that at one time, a metapopulation is regulated by the mechanisms in the core-satellite model, including the rescue effect, and $c_i > e$. We therefore pretend that, the metapopulation occupies virtually every habitable site (let $p = 0.999$). Now imagine that the environment changes, causing $c_i < e$. Perhaps human urbanization reduces colonization rates, or climate change enhances extinction rates. All of a sudden, our metapopulation is poised on an unstable equilibrium. What will happen and how might it differ with and without the rescue effect?

When $c_i > e$, we see that $p^* = 1$ is the stable attractor (Fig. 4.8). However, when $c_i < e$, we see the inevitable march toward extinction predicted by the Hanski model (core-satellite) (Fig. 4.8). Last, when we compare it to the Levins model, we realize something somewhat more interesting. While the Levins model predicts very rapid decline, the Hanski model predicts a much more gradual decline *toward extinction*. Both models predict extinction, but the rescue effect delays the appearance of that extinction. It appears that the rescue effect (which is the difference between the two models) may act a bit like the "extinction debt" [202] wherein deterministic extinction is merely delayed, but not postponed indefinitely. Perhaps populations influenced by the rescue effect might be prone to unexpected collapse, if the only stable equilibria are 1 and 0. Thus simple theory can provide interesting insight, resulting in very different predictions for superficial similar processes.

The unexpected collapse of core populations

Here we plot the dynamics of metapopulations starting at or near equilbrium. The first two use the Hanski model, while the third uses Levins. The second and third use $c_i < e$.

```
> C1 <- ode(y = 0.999, times = t, func = hanski, parms = c(ci = 0.2,
+     e = 0.01))
> C2 <- ode(y = 0.999, times = t, func = hanski, parms = c(ci = 0.2,
+     e = 0.25))
> L2 <- ode(y = 0.95, times = t, func = levins, parms = c(ci = 0.2,
+     e = 0.25))
```

Next, we plot these and add a legend.

```
> matplot(t, cbind(C1[, 2], C2[, 2], L2[, 2]), type = "l",
+     ylab = "p", xlab = "Time", col = 1)
> legend("right", c("c > e", "c < e", "c < e (Levins)"), lty = 1:3,
+     bty = "n")
```

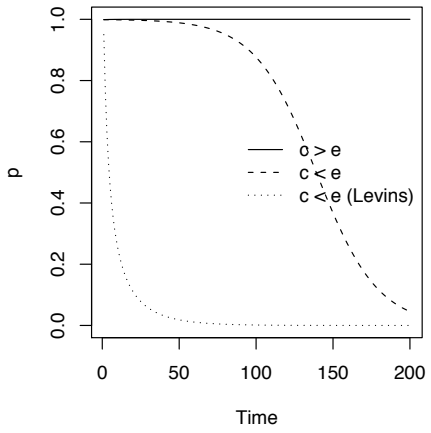


Fig. 4.8: Metapopulation dynamics, starting from near equilibrium for $c_i = 0.20$ and $e = 0.01$. If the environment changes, causing extinction rate to increase until it is greater than colonization rate, we may observe greatly delayed, but inevitable, extinction (e.g., $c_i = 0.20, e = 0.25$).

4.6 Core-Satellite Simulations

Here[1] we explore a simple question that Hanski posed long ago: what would communities look like if all of the populations in the community could be de-

[1] This section relies extensively on code

scribed by their independent own core-satellite model? To answer this question, he created communities as collections of independent (non-interacting) populations that behave according to his metapopulation model with internal colonization and the rescue effect [70]. He found that such simulated communities predict that many species will be in almost all sites ("core species"), and even more species will exist at very few sites ("satellite species"). This seems to be a relatively common phenomenon [35], and an observation we described at the beginning of the chapter (Fig. 4.1).

Hanksi's goal was to simulate simultaneously a substantive number of species, to create a community. Each species is assumed to be governed by internal propagule production only, and the rescue effect. Further, he assumed that the long term average density independent growth rate ($r = c_i - e$) was zero. That is, the populations were not *systematically* increasing or decreasing. However, he allowed for stochastic year-to-year variation in probabilities c_i and e.

In these simulations here, we will select the mean for each parameter, c_i and e, and the proportion, ϕ ("phi") by which they are allowed to vary. The realized values of $c_{i,t}$ and e_t at any one point in time are random draws from a uniform distribution within the ranges $i \pm \phi i$ and $e \pm \phi e$. (This requires that we do numerical integration at each integer time step since there is no obvious analytical solution to an equation in which the parameters vary through time. This will keep these parameters constant for an entire year, and yet also allow years to vary.)

We start by using the `args()` function to find out what arguments (i.e. options) are available in the simulation function, `MetaSim`.

```
> args(MetaSim)

function (Time = 50, NSims = 1, method = "hanski", ci = 0.25,
    e = 0.25, phi = 0.75, p0 = 0.5, D = 0.5)
NULL
```

What options (or arguments) can you vary in `MetSim`? The 'method' may equal `CoreSatellite`, `Levins`, `IslandMainland`, or `HabitatDestruction`. The default is `CoreSatellite`; if an argument has a value to begin with (e.g. `method='CoreSatellite'`), then it will use that value unless you replace it.

Let's start with an initial run of 10 simulations (produces dynamics for 10 populations) to reproduce Hanski's core-satellite pattern by using the rescue effect with equal i and e.

```
> out.CS.10 <- MetaSim(method = "hanski", NSims = 10)
> matplot(out.CS.10$t, out.CS.10$Ns, type = "l", xlab = "Time",
+     ylab = "Occupancy", sub = out.CS.10$method)
```

These dynamics (Fig. 4.9) appear to be completely random. A *random walk* is a dynamic that is a random increase or decrease at each time step. Such a process is not entirely random because the abundance at time t is related to the abundance at time $t-1$, so observations in random walks are correlated through time; they are temporally autocorrelated.

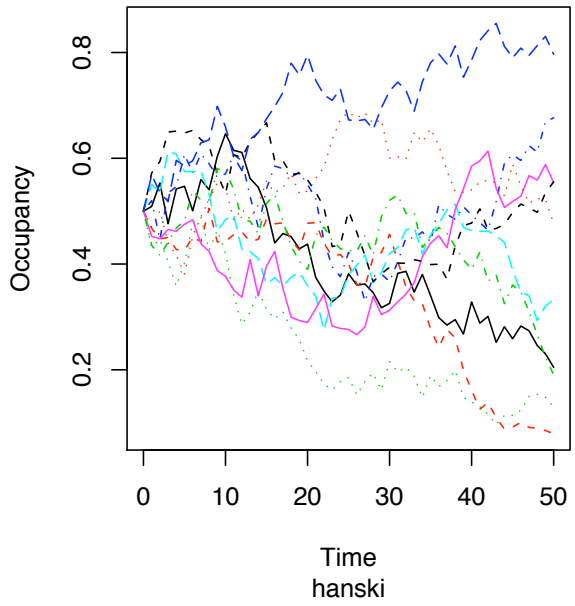


Fig. 4.9: Core-satellite species dynamics with stochasticity ($\bar{i} = \bar{e} = 0.2$).

Does a single metapopulation growth rate appear related to p, the metapopulation size? What would a deterministic dynamic look like if $c_i > e$? It would increase rapidly at first, and then slow down as it approached 1.0. Can you detect that slow-down here? Similarly, as a metapopulation declines toward extinction, its progression toward $p = 0$ slows down. As a result, we tend to accumulate a lot of common and rare species, for which p is close to one or zero.

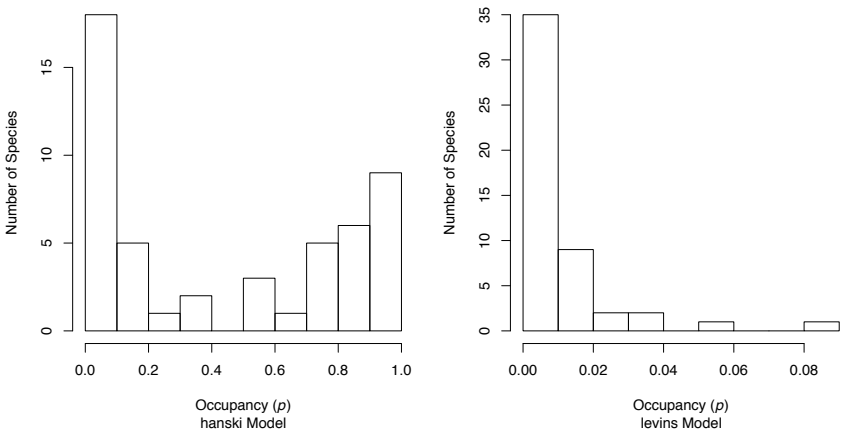
Now we will do more simulations (50 species), and run them for longer (500 time intervals *vs.* 50). Doing many more simulations will take a little longer, so be patient[2].

```
> system.time(out.CS.Lots <- MetaSim(method = "hanski", NSims = 50,
+     Time = 1000))

   user  system elapsed
 48.524   0.072  48.603
```

time series, although this may not tell you much. Alternatively, we can plot a histogram of the 50 species' final abundances, at $t = 500$.

```
> hist(out.CS.Lots$Ns[501, ], breaks = 10, main = NULL,
+    xlab = expression("Occupancy (" * italic("p") * ")"),
+    ylab = "Number of Species",
+    sub = paste(out.CS.Lots$method, " Model", sep = ""))
```

[2] `system.time` merely times the process, in secs.



(a) With Rescue Effect (b) No rescue effect

Fig. 4.10: The species-abundance distribution resulting from dynamics for 50 independent metapopulations with internal colonization. (a) includes the rescue effect (Hanski's model), and note that most species are either common ($p > 0.8$) or rare ($p < 0.2$). Levins model (b) does not include the rescue effect, and there are very few core species ($p > 0.8$).

Our simulations (Fig. 4.10) should be consistent with the core-satellite hypothesis — are they? In Hanski's model, we see that most metapopulations are either core species ($p > 0.8$) or satellite species ($p < 0.2$) (Fig. 4.10a). This is not to imply that there should be hard rules about what constitutes a core and satellite species, but rather merely shows we have a plethora of both common and uncommon species.

What does the Levins model predict? Let's run the simulations and find out.

```
> system.time(out.L.Lots <- MetaSim(NSims = 50, Time = 500,
+     method = "levins"))

   user  system elapsed
 23.280   0.027  23.308
```

Now we plot a histogram of the 50 species' final abundances, at $t = 500$.

```
> hist(out.L.Lots$Ns[501, ], breaks = 10, xlab = expression("Occupancy (" *
+     italic("p") * ")"), ylab = "Number of Species", main = NULL,
+     sub = paste(out.L.Lots$method, " Model", sep = ""))
```

In contrast to the core-satellite model, the Levins model predicts that many fewer species are common (Fig. 4.10b). Thus these two population models make contrasting predictions regarding the structure of communities (i.e. relative species abundances), and provide testable alternatives [35].

4.7 Summary

In this chapter, we have introduced *space* as an important component of population dynamics. We provided a source-sink framework for linked populations, where population size depends on both intrinsic capacities of a habitat patch, and on immigration and emigration rates. We described (i) a population of individuals within a site, and (ii) a population of populations within a region in the "metapopulation" framework. We showed similarities and differences between related metapoopulation models, and between related metapopulation and logistic models. We investigated the response of metapopulations to habitat destruction. Last, we have shown how different population dynamics lead to different community structure.

Problems

4.1. Equilibria
Derive expressions and calculate equilibria for the following metapopulation models, with $c_i = 0.05,\ e = 0.01$. Show your work — start with the differential equations, set to zero, and solve p^*; then substitute in values for $c_i,\ e$.
(a) Levins model.
(b) Propagule rain model (`gotelli`).
(c) Propagule rain model that also includes both external and internal propagule production and dispersal.
(d) Hanski model.
(e) Lande (habitat destruction) model (with `D=0.1`).

4.2. Habitat destruction
Compare different levels of habitat destruction.
(a) Use the habitat destruction model (`lande`) to compare 9 levels of destruction (`ds <- seq(0,.8, by=.1)`), using $c_i = 0.1,\ e = 0.01$. Plot of graph of the dynamics through time, and calculate the equilibria directly.
(b) Write an ODE function for a habitat destruction model with rescue effect. Let the "rescue" have an additional parameter, a, such that extinction rate is $ep(1 - ap)$.
(c) Let $D = 0.5,\ c_i = 0.1,\ e = 0.02$, and vary a over five levels (including $a = 0,\ 1$) to investigate the effects of "relative rescue effect" on the equilibria and dynamics of a metapopulation.

10

Community Composition and Diversity

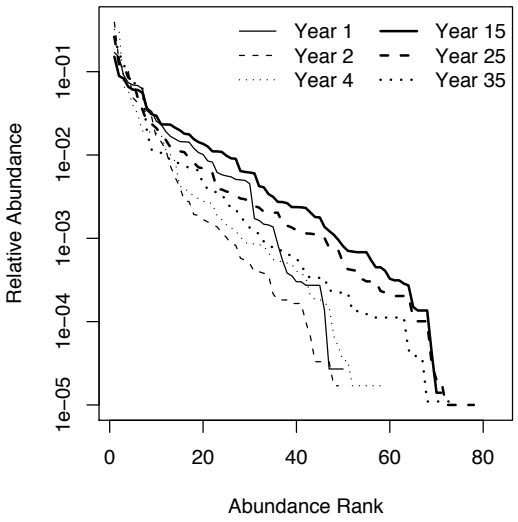


Fig. 10.1: Empirical rank–abundance distributions of successional plant communities (old-fields) within the temperate deciduous forest biome of North America. "Year" indicates the time since abandonment from agriculture. Data from the Buell-Small succession study (http://www.ecostudies.org/bss/)

It seems easy, or at least tractable, to compare the abundance of a single species in two samples. In this chapter, we introduce concepts that ecologists use to compare entire communities in two samples. We focus on two quantities: *species composition*, and *diversity*. We also discuss several issues related to this, including species–abundance distributions, ecological neutral theory, diversity partitioning, and species–area relations. Several packages in R include functions for dealing specifically with these topics. Please see the "Environmetrics" link

within the "Task Views" link at any CRAN website for downloading Rpackages. Perhaps the most comprehensive (including both diversity and composition) is the `vegan` package, but many others include important features as well.

10.1 Species Composition

Species composition is merely the set of species in a site or a sample. Typically this includes some measure of abundance at each site, but it may also simply be a list of species at each site, where "abundance" is either presence or absence. Imagine we have four sites (A–D) from which we collect density data on two species, *Salix whompii* and *Fraxinus virga*. We can enter hypothetical data of the following abundances.

```
> dens <- data.frame(Salwho = c(1, 1, 2, 3), Fravir = c(21,
+     8, 13, 5))
> row.names(dens) <- LETTERS[1:4]
> dens
```

```
  Salwho Fravir
A      1     21
B      1      8
C      2     13
D      3      5
```

Next, we plot the abundances of both species; the plotted points then are the sites (Fig. 10.2).

```
> plot(dens, type = "n")
> text(dens, row.names(dens))
```

In Fig. 10.2, we see that the species composition in site A is most different from the composition of site D. That is, the distance between site A and D is greater than between any other sites. The next question, then, is *how* far apart are any two sites? Clearly, this depends on the scale of the measurement (e.g., the values on the axes), and also on how we measure distance through multivariate space.

10.1.1 Measures of abundance

Above we pretended that the abundances were absolute densities (i.e., 1 = one stem per sample). We could of course represent all the abundances differently. For instance, we could calculate *relative density*, where each species in a sample is represented by the *proportion* of the sample comprised of that species. For site A, we divide each species by the sum of all species.

```
> dens[1, ]/sum(dens[1, ])
```

```
   Salwho Fravir
A 0.04545 0.9545
```

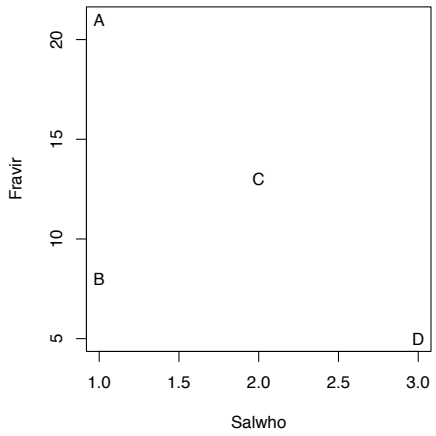


Fig. 10.2: Hypothetical species composition for four sites (A–D).

We see that *Salix* makes up about 5% of the sample for Site A, and *Fraxinus* makes up about 95% of the sample. Once we calculate relative densities for each species at each site, this eliminates differences in total density at each site because all sites then total to 1.

We could also calculate relative measures for any type of data, such as biomass or percent cover.

In most instances, *relative density* refers to the density of a species *relative* to the other species in a sample (above), but it can also be density in a sample relative to other samples. We would thus make each *species* total equal 1, and then its abundance at each site reflects the proportion of a species total abundance comprised by that site. For instance, we can make all *Salix* densities relative to each other.

```
> dens[, 1]/sum(dens[, 1])

[1] 0.1429 0.1429 0.2857 0.4286
```

Here we see that sites A and B both have about 14% of all *Salix* stems, and site D has 43%.

Whether our measures of abundance are absolute or relative, we would like to know how different samples (or sites) are from each other. Perhaps the simplest way to describe the difference among the sites is to calculate the *distances* between each pair of sites.

10.1.2 Distance

There are many ways to calculate a *distance* between a pair of sites. One of the simplest, which we learned in primary school, is *Euclidean distance*. With two species, we have two dimensional space, which is Fig. 10.2. The Euclidean distance between two sites is merely the length of the vector connecting those

sites. We calculate this as $\sqrt{x^2 + y^2}$, where x and y are the (x, y) distances between a pair of sites. The x distance between sites B and C is difference in *Salix* abundance between the two sites,

```
> x <- dens[2, 1] - dens[3, 1]
```

where `dens` is the data frame with sites in rows, and species in different columns. The y distance between sites B and C is difference in *Fraxinus* abundance between the two sites.

```
> y <- dens[2, 2] - dens[3, 2]
```

The Euclidean distance between these is therefore

```
> sqrt(x^2 + y^2)

[1] 5.099
```

Distance is as simple as that. We calculate all pairwise Euclidean distances between sites A–D based on 2 species using built-in functions in R.

```
> (alldists <- dist(dens))

       A      B      C
B 13.000
C  8.062  5.099
D 16.125  3.606  8.062
```

We can generalize this to include any number of species, but it becomes increasingly harder to visualize. We can add a third species, *Mandragora officinarum*, and recalculate pairwise distances between all sites, but now with three species.

```
> dens[["Manoff"]] <- c(11, 3, 7, 5)
> (spp3 <- dist(dens))

       A      B      C
B 15.264
C  9.000  6.481
D 17.205  4.123  8.307
```

We can plot species abundances as we did above, and `pairs(dens)` would give us all the pairwise plots given three species. However, what we really want for species is a 3-D plot. Here we load another package[1] and create a 3-D scatterplot.

```
> pairs(dens)# not shown
> library(scatterplot3d)
> sc1 <- scatterplot3d(dens,  type='h', pch="",
+       xlim=c(0,5), ylim=c(0, 25), zlim=c(0,15))
> text(sc1$xyz.convert(dens), labels=rownames(dens))
```

In three dimensions, Euclidean distances are calculated the same basic way, but we add a third species, and the calculation becomes $\sqrt{x^2 + y^2 + z^2}$. Note that we take the square root (as opposed to the cube root) because we originally *squared* each distance. We can generalize this for two sites for $\boldsymbol{R}$ species as

[1] You can install this package from any R CRAN mirror.

$$D_E = \sqrt{\sum_{i=1}^{R} (x_{ai} - x_{bi})^2} \tag{10.1}$$

Of course, it is difficult (impossible?) to visualize arrangements of sites with more than three axes (i.e., > 3 species), but we can always *calculate* the distances between pairs of sites, regardless of how many species we have.

There are many ways, in addition to Euclidean distances, to calculate distance. Among the most commonly used in ecology is *Bray–Curtis* distance, which goes by other names, including *Søorenson* distance.

$$D_{BC} = \sum_{i=1}^{R} \frac{|x_{ai} - x_{bi}|}{x_{ai} + x_{bi}} \tag{10.2}$$

where R is the number of species in all samples. Bray–Curtis distance is merely the total difference in species abundances between two sites, divided by the total abundances at each site. Bray–Curtis distance (and a couple others) tends to result in more intuitively pleasing distances in which both common and rare species have relatively similar weights, whereas Euclidean distance depends more strongly on the most abundant species. This happens because Euclidean distances are based on squared differences, whereas Bray–Curtis uses absolute differences. Squaring always amplifies the importance of larger values. Fig. 10.3 compares graphs based on Euclidean and Bray–Curtis distances of the same raw data.

Displaying multidimensional distances

A simple way to display distances for three or more species is to create a plot in two dimensions that attempts to arrange all sites so that they are *approximately* the correct distances apart. In general this is impossible to achieve precisely, but distances can be approximately correct. One technique that tries to create an optimal (albiet approximate) arrangement is *non-metric multidimensional scaling*. Here we add a fourth species (*Aconitum lycoctonum*) to our data set before plotting the distances.

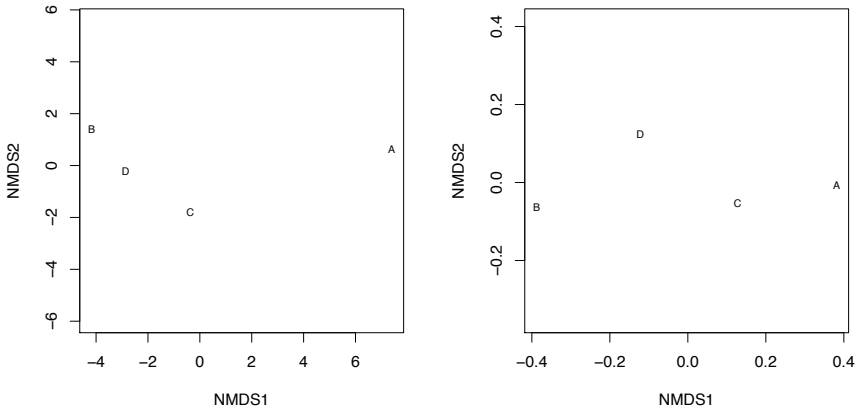
```
> dens$Acolyc <- c(16, 0, 9, 4)
```

The non-metric multidimensional scaling function is in the `vegan` package. It calculates distances for us using the original data. Here we display Euclidean distances among sites (Fig. 10.3a).

```
> library(vegan)
> mdsE <- metaMDS(dens, distance = "euc", autotransform = FALSE,
+     trace = 0)
> plot(mdsE, display = "sites", type = "text")
```

Here we display Bray–Curtis distances among sites (Fig. 10.3b).

```
> mdsB <- metaMDS(dens, distance = "bray", autotransform = FALSE,
+     trace = 0)
> plot(mdsB, display = "sites", type = "text")
```



(a) Euclidean distances (b) Bray–Curits distances

Fig. 10.3: Nonmetric multidimensional (NMDS) plots showing approximate distances between sites. These two figures display the same raw data, but Euclidean distances tend to emphasize differences due to the more abundant species, whereas Bray-Curtis does not. Because NMDS provides iterative optimizations, it will find slightly different arrangements each time you run it.

10.1.3 Similarity

Sometimes, we would like to know how *similar* two communities are. Here we describe two measures of similarity, *percent similarity*, and *Sørensen similarity* [124].

Percent similarity may be the simplest of these; it is simply the sum of the minimum percentages for each species in the community. Here we convert each species to its relative abundance; that is, its proportional abundance at each site. To do this, we treat each site (row) separately, and then divide the abundance of each species by the sum of the abundances at each site.

```
> (dens.RA <- t(apply(dens, 1, function(sp.abun) sp.abun/sum(sp.abun))))
```

```
   Salwho Fravir Manoff Acolyc
A 0.02041 0.4286 0.2245 0.3265
B 0.08333 0.6667 0.2500 0.0000
C 0.06452 0.4194 0.2258 0.2903
D 0.17647 0.2941 0.2941 0.2353
```

Next, to compare two sites, we find the minimum relative abundance for each species. Comparing sites A and B, we have,

```
> (mins <- apply(dens.RA[1:2, ], 2, min))
```

```
 Salwho  Fravir  Manoff  Acolyc
0.02041 0.42857 0.22449 0.00000
```

Finally, we sum these, and multiply by 100 to get percentages.

```
> sum(mins) * 100

[1] 67.35
```

The second measure of similarity we investigate is Sørensen's similarity,

$$S_s = \frac{2C}{A+B} \tag{10.3}$$

where C is the number of species two sites have in common, and A and B are the number of species at each site. This is equivalent to dividing the shared species by the average richness.

To calculate this for sites A and B, we could find the species which have non-zero abundances both sites.

```
> (shared <- apply(dens[1:2, ], 2, function(abuns) all(abuns !=
+     0)))

Salwho Fravir Manoff Acolyc
  TRUE   TRUE   TRUE  FALSE
```

Next we find the richness of each.

```
> (Rs <- apply(dens[1:2, ], 1, function(x) sum(x >
+     0)))

A B
4 3
```

Finally, we divide the shared species by the summed richnesses and multiply by 2.

```
> 2 * sum(shared)/sum(Rs)

[1] 0.8571
```

Sørensen's index has also been used in the development of more sophisticated measures of similarity between sites [144, 164].

10.2 Diversity

To my mind, there is no more urgent or interesting goal in ecology and evolutionary biology than understanding the determinants of biodiversity. Biodiversity is many things to many people, but we typically think of it as a measure of the variety of biological life present, perhaps taking into account the relative abundances. For ecologists, we most often think of *species diversity* as some quantitative measure of the variety or number of different species. This has direct analogues to the genetic diversity within a population [45, 213], and the connections between species and genetic diversity include both shared patterns and shared mechanisms. Here we confine ourselves entirely to a discussion of species diversity, the variety of different species present. Consider this example.

Table 10.1: Four hypothetical stream invertebrate communities. Data are total numbers of individuals collected in ten samples (sums across samples). Diversity indices (Shannon-Wiener, Simpson's) explained below.

Species	Stream 1	Stream 2	Stream 3	Stream 4
Isoperla	20	50	20	0
Ceratopsyche	20	75	20	0
Ephemerella	20	75	20	0
Chironomus	20	0	140	200
Number of species (R)	4	3	4	1
Shannon-Wiener H	1.39	1.08	0.94	0
Simpson's S_D	0.75	0.66	0.48	0

We have four stream insect communities (Table 10.1). Which has the highest "biodiversity"?

We note that one stream has only one species — clearly that can't be the most "diverse" (still undefined). Two streams have four species — are they the most diverse? Stream 3 has more bugs in total (200 *vs.* 80), but stream 1 has a more equal distribution among the four species.

10.2.1 Measurements of variety

So, how shall we define "diversity" and measure this variety? There are many mathematical expressions that try to summarize biodiversity [76, 92]. The inquisitive reader is referred to [124] for a practical and comprehensive text on measures of species diversity. Without defining it precisely (my pay scale precludes such a noble task), let us say that diversity indices attempt to quantify

- the probability of encountering different species at random, or,
- the uncertainty or multiplicity of possible community states (i.e., the entropy of community composition), or,
- the variance in species composition, relative to a multivariate centroid.

For instance, a simple count of species (Table 10.1) shows that we have 4, 3, 4, and 1 species collected from streams 1–4. The larger the number of species, the less certain we could be about the identity of an individual drawn blindly and at random from the community.

To generalize this further, imagine that we have a species pool[2] of R species, and we have a sample comprised of only one species. In a sample with only one species, then we know that the possible *states* that sample can take is limited to one of only R different possible states — the abundance of one species is 100% and all others are zero. On the other hand, if we have two species then the community could take on $R(R-1)$ different states — the first species could be any one of R species, and the second species could be any one of the other species, and all others are zero. Thus increasing diversity means increasing

[2] A *species pool* is the entire, usually hypothetical, set of species from which a sample is drawn; it may be all of the species known to occur in a region.

the possible states that the community could take, and thus increasing our uncertainty about community structure [92]. This increasing lack of information about the system is a form of *entropy*, and increasing diversity (i.e., increasing multiplicity of possible states) is increasing entropy. The jargon and topics of statistical mechanics, such as entropy, appear (in 2009) to be an increasingly important part of community ecology [71, 171].

Below we list three commonly used diversity indices: species richness, Shannon-Wiener index, and Simpson's diversity index.

- Species richness, R, the count of the number of species in a sample or area; the most widely used measure of biodiversity [82].
- Shannon-Wiener diversity.[3]

$$H' = -\sum_{i=1}^{R} p \ln(p) \tag{10.4}$$

  where R is the number of species in the community, and p_i is the relative abundance of species i.
- Simpson's diversity. This index is (i) the probability that two individuals drawn from a community at random will be different species [147], (ii) the initial slope of the species-individuals curve [98] (e.g., Fig. 10.6), and (iii) the expected variance of species composition (Fig. 10.5) [97, 194].

$$S_D = 1 - \sum_{i=1}^{R} p_i^2 \tag{10.5}$$

  The summation $\sum_{i=1}^{R} p_i^2$ is the probability that two individuals drawn at random are the same species, and it is known as Simpson's "dominance." Lande found that this Simpon's index can be more precisely estimated, or estimated accurately with smaller sample sizes, than either richness or Shannon-Wiener [97].

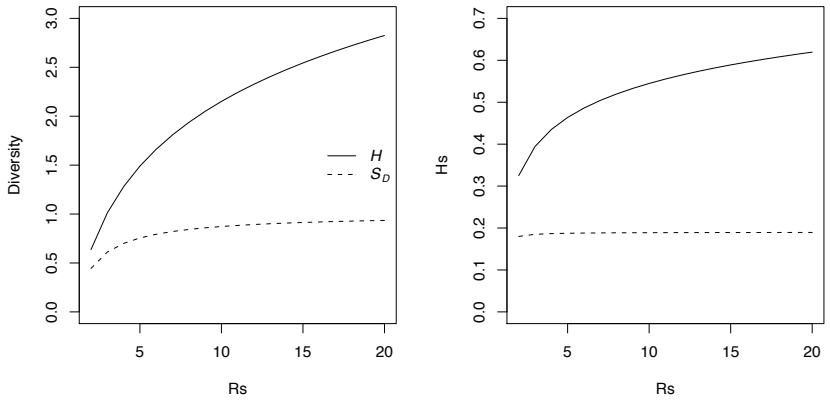
These three indices are actually directly related to each other — they comprise estimates of *entropy*, the amount of disorder or the multiplicity of possible states of a system, that are directly related via a single constant [92]. However, an important consequence of their differences is that richness depends most heavily on rare species, Simpson's depends most heavily on common species, and Shannon-Wiener stands somewhere between the two (Fig. 10.4).

Relations between number of species, relative abundances, and diversity

This section relies heavily on code and merely generates Fig. 10.4.

Here we display diversities for communities with different numbers and relative abundances of species (Fig. 10.4). We first define functions for the diversity indices.

[3] Robert May stated that this index is connected by merely an "ectoplasmic thread" to information theory [135], but there seems be a bit more connection than that.



(a) Equally abundant (b) Most common sp. = 90%

Fig. 10.4: Relations between richness, Shannon-Weiner, and Simpson's diversities (note difference in *y*-axis scale between the figures). Communities of 2–20 species are composed of either equally abundant species (a) or with the most common species equal to 90% of the community (b).

```
> H <- function(x) {
+     x <- x[x > 0]
+     p <- x/sum(x)
+     -sum(p * log(p))
+ }
> Sd <- function(x) {
+     p <- x/sum(x)
+     1 - sum(p^2)
+ }
```

Next we create a *list* of communities with from 1 to 20 *equally* abundant species, and calculate H and S_D for each community.

```
> Rs <- 2:20
> ComsEq <- sapply(Rs, function(R) (1:R)/R)
> Hs <- sapply(ComsEq, H)
> Sds <- sapply(ComsEq, Sd)
> plot(Rs, Hs, type = "l", ylab = "Diversity", ylim = c(0,
+     3))
> lines(Rs, Sds, lty = 2)
> legend("right", c(expression(italic("H")), expression(italic("S"["D"]))),
+     lty = 1:2, bty = "n")
```

Now we create a *list* of communities with from 2 to 25 species, where one species always comprises 90% of the community, and the remainder are equally abundant rare species. We then calculate H and S_D for each community.

```
> Coms90 <- sapply(Rs, function(R) {
+     p <- numeric(R)
```

```
+     p[1] <- 0.9
+     p[2:R] <- 0.1/(R - 1)
+     p
+ })
> Hs <- sapply(Coms90, H)
> Sds <- sapply(Coms90, Sd)
> plot(Rs, Hs, type = "l", ylim = c(0, 0.7))
> lines(Rs, Sds, lty = 2)
```

Simpson's diversity, as a variance of composition

This section relies heavility on code.

Here we show how we would calculate the variance of species composition. First we create a pretend community of six individuals (rows) and 3 species (columns). Somewhat oddly, we identify the degree to which each individual is comprised of each species; in this case, individuals can be only one species.[4] Here we let two individuals be each species.

```
> s1 <- matrix(c(1, 1, 0, 0, 0, 0, 0, 0, 1, 1, 0,
+     0, 0, 0, 0, 0, 1, 1), nr = 6)
> colnames(s1) <- c("Sp.A", "Sp.B", "Sp.C")
> s1
```

```
     Sp.A Sp.B Sp.C
[1,]    1    0    0
[2,]    1    0    0
[3,]    0    1    0
[4,]    0    1    0
[5,]    0    0    1
[6,]    0    0    1
```

We can plot these individuals in community space, if we like (Fig. 10.5a).

```
> library(scatterplot3d)
> s13d <- scatterplot3d(jitter(s1, 0.3), type = "h",
+     angle = 60, pch = c(1, 1, 2, 2, 3, 3), xlim = c(-0.2,
+         1.4), ylim = c(-0.2, 1.4), zlim = c(-0.2,
+         1.4))
> s13d$points3d(x = 1/3, y = 1/3, z = 1/3, type = "h",
+     pch = 19, cex = 2)
```

Next we can calculate a *centroid*, or multivariate mean — it is merely the vector of species means.

```
> (centroid1 <- colMeans(s1))
```

```
  Sp.A   Sp.B   Sp.C
0.3333 0.3333 0.3333
```

[4] Imagine the case where the columns are traits, or genes. In that case, individuals could be characterized by affiliation with multiple columns, whether traits or genes.

Given this centroid, we begin to calculate a variance by (i) substracting each species vector (0s, 1s) from its mean, (ii) squaring each of these deviates, and (3) summing to get the sum of squares.

```
> (SS <- sum(sapply(1:3, function(j) (s1[, j] -
+     centroid1[j]))^2))
```

```
[1] 4
```

We then divide this sum by the number of individuals that were in the community (N)

```
> SS/6
```

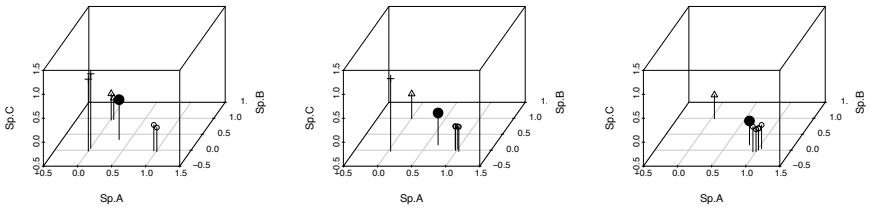
```
[1] 0.6667
```

We find that the calculation given above for Simpson's diversity returns exactly the same number. We would calculate the relative abundances, square them, add them, and subtract that value from 1.

```
> p <- c(2, 2, 2)/6
> 1 - sum(p^2)
```

```
[1] 0.6667
```

In addition to being the variance of species composition, this number is also the probability that two individuals drawn at random are different species. As we mentioned above, there are other motivations than these to derive this and other measures of species diversity, based on entropy and information theory [92] — and they are all typically correlated with each other.



(a) $S_D = 0.67$ (b) $S_D = 0.50$ (c) $S_D = 0.28$

Fig. 10.5: Plotting three examples of species composition. The centroid of each composition is a solid black dot. The third example (on right) has zero abundances of species C. Simpson's diversity is the variance of these points around the centroid. Individual points are not plotted at precisely 0 or 1 — they are plotted with a bit of jitter or noise so that they do not overlap entirely.

10.2.2 Rarefaction and total species richness

Rarefaction is the process of generating the relationship between the number of species *vs.* the number of individuals in one or more samples. It is typically determined by randomly resampling individuals [59], but could also be determined by resampling samples. Rarefaction allows direct comparison of the richness of two samples, corrected for numbers of individuals. This is particularly important because R depends heavily on the number of individuals in a sample. Thus rarefaction finds the general relation between the number(s) of species *vs.* number of individuals (Fig. 10.6), and is limited to less than or equal to the number of species you actually observed. A related curve is the *species-accumulation* curves, but this is simply a useful but haphazard accumulation of new species (a cumulative sum) as the investigator samples new individuals.

Another issue that ecologists face is trying to estimate the *true* number of species in an area, given the samples collected. This number of species would be larger than the number of species you observed, and is often referred to as *total species richness* or the *asymptotic richness*. Samples almost always find only a subset of the species present in an area or region, but we might prefer to know how many species are really there, in the general area we sampled. There are many ways to do this, and while some are better than others, none is perfect. These methods estimate minimum numbers of species, and assume that the unsampled areas are homogeneous and similar to the sampled areas.

Before using these methods seriously, the inquisitive reader should consult [59, 124] and references at http://viceroy.eeb.uconn.edu/EstimateS. Below, we briefly explore an example in R.

An example of rarefaction and total species richness

Let us "sample" a seasonal tropical rainforest on Barro Colorado Island (BCI) http://ctfs.si.edu/datasets/bci/). Our goal will be to provide baseline data for later comparison to other such studies.

We will use some of the data from a 50 ha plot that is included in the `vegan` package [36, 151]. We will pretend that we sampled every tree over 10 cm dbh,[5] in each of 10 plots scattered throughout the 50 ha area. What could we say about the forest within which the plots were located? We have to consider the scale of the sampling. Both the *experimental unit* and the *grain* are the 1 ha plots. Imagine that the plots were scattered throughout the 50 ha plot, so that the extent of the sampling was a full 50. ha[6] First, let's pretend we have sampled 10 1 ha plots by extracting the samples out of the larger dataset.

```
> library(vegan)
> data(BCI)
> bci <- BCI[seq(5, 50, by = 5), ]
```

[5] "dbh" is diameter at 1.37 m above the ground.

[6] See John Wiens' foundational paper on spatial scale in ecology [220] describing the meaning of grain, extent, and other spatial issues.

Next, for each species, I sum all the samples into one, upon which I will base rarefaction and total richness estimation.

Next we combine all the plots into one sample (a single summed count for each species present), select numbers of individuals for which I want rarefied samples (multiples of 500), and then perform rarefaction for each of those numbers.

```
> N <- colSums(bci)
> subs3 <- c(seq(500, 4500, by = 500), sum(N))
> rar3 <- rarefy(N, sample = subs3, se = T, MARG = 2)
```

Next we want to graph it, with a few bells and whistles. We set up the graph of the 10 plot, individual-based rarefaction, and leave room to graph total richness estimators as well (Fig. 10.6).

```
> plot(subs3, rar3[1, ], ylab = "Species Richness",
+     axes = FALSE, xlab = "No. of Individuals",
+     type = "n", ylim = c(0, 250), xlim = c(500,
+         7000))
> axis(1, at = 1:5 * 1000)
> axis(2)
> box()
> text(2500, 200, "Individual-based rarefaction (10 plots)")
```

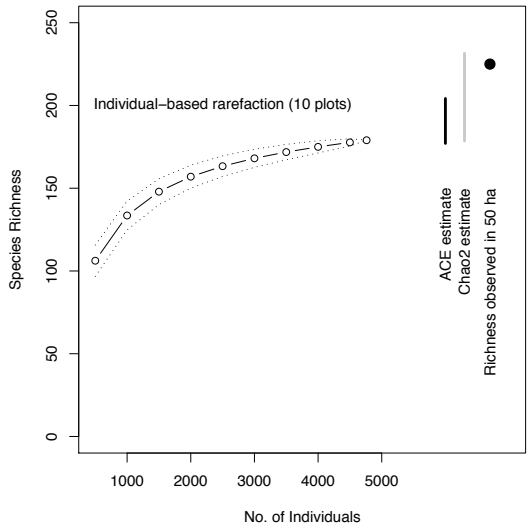


Fig. 10.6: Baseline tree species richness estimation based on ten 1 ha plots, using individual-based rarefaction, and two different total richness estimators, ACE and Chao 2. The true total tree richness in the 50 ha plot is present for comparison.

Here we plot the expected values and also ± 2 SE.

```
> lines(subs3, rar3[1, ], type = "b")
> lines(subs3, rar3[1, ] + 2 * rar3[2, ], lty = 3)
> lines(subs3, rar3[1, ] - 2 * rar3[2, ], lty = 3)
```

Next we hope to estimate the minimum total number of species (asymptotic richness) we might observe in the area around (and in) our 10 plots, if we can assume that the surrounding forest is homogeneous (it would probably be best to extrapolate only to the 50 ha plot). First, we use an *abundance-based coverage estimator*, *ACE*, that appears to give reasonable estimates [124]. We plot intervals, the expected values ± 2 SE (Fig. 10.6).

```
> ace <- estimateR(N)
> segments(6000, ace["S.ACE"] - 2 * ace["se.ACE"],
+     6000, ace["S.ACE"] + 2 * ace["se.ACE"], lwd = 3)
> text(6000, 150, "ACE estimate", srt = 90, adj = c(1,
+     0.5))
```

Next we use a frequency-based estimator, Chao 2, where the data only need to be presence/absence, but for which we also need multiple sample plots.

```
> chaoF <- specpool(bci)
> segments(6300, chaoF[1, "Chao"] - 2 * chaoF[1,
+     "Chao.SE"], 6300, chaoF[1, "Chao"] + 2 * chaoF[1,
+     "Chao.SE"], lwd = 3, col = "grey")
> text(6300, 150, "Chao2 estimate", srt = 90, adj = c(1,
+     0.5))
```

Last we add the observed number of tree species (over 10 cm dbh) found in the entire 50 ha plot.

```
> points(6700, dim(BCI)[2], pch = 19, cex = 1.5)
> text(6700, 150, "Richness observed in 50 ha",
+     srt = 90, adj = c(1, 0.5))
```

This shows us that the total richness estimators did not overestimate the total number of species within the extent of this relatively homogenous sample area (Fig. 10.6).

If we wanted to, we could then use any of these three estimators to compare the richness in this area to the richness of another area.

10.3 Distributions

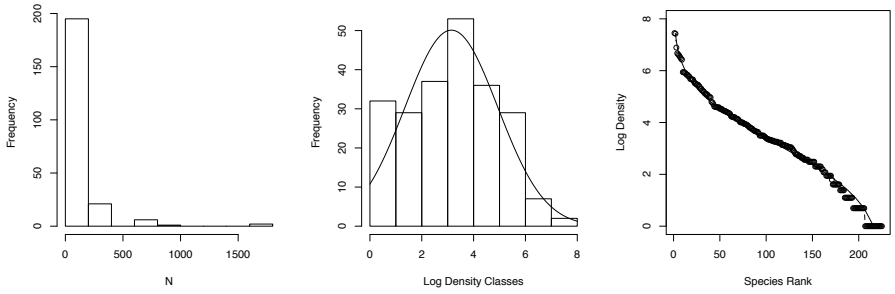
In addition to plotting species in multidimensional space (Fig. 10.3), or estimating a measure of diversity or richness, we can also examine the *distributions* of species abundances.

Like any other vector of numbers, we can make a histogram of species abundances. As an example, here we make a histogram of tree densities, where each species has its own density (Fig. 10.7a). This shows us what is patently true for nearly all ecological communities — *most species are rare*.

10.3.1 Log-normal distribution

Given general empirical patterns, that most species are rare, Frank Preston [168,170] proposed that we describe communities using the logarithms of species abundances (Fig. 10.7).[7] This often reveals that a community can be described approximately with the normal distribution applied to the log-transformed data, or the *log-normal ditribution*. We can also display this as a *rank–abundance distribution* (Fig. 10.7c). To do this, we assign the most abundant species as rank = 1, and the least abundant has rank = R, in a sample of R species, and plot log-abundance *vs.* rank.

May [129] described several ways in which common processes may drive log-normal distributions, and cause them to be common in data sets. Most commonly cited is to note that log-normal distributions arise when each observation (i.e., each random variable) results from the product of independent factors. That is, if each species' density is determined by largely independent factors which act multiplicatively on each species, the resulting densities would be log-normally distributed.



(a) Raw data (b) Species–abundance dist. (c) Rank–abundance dist.

Fig. 10.7: Three related types of distributions of tree species densities from Barro Colorado Island [36]. (a) Histogram of raw data, (b) histogram of log-transformed data; typically referred to as the "species–abundance distribution," accompanied here with the normal probability density function, (c) the "rank–abundance distribution," as typically presented with the log-transformed data, with the complement of the cumulative probability density function (1-pdf) [129]. Normal distributions were applied using the mean and standard deviation from the log-transformed data, times the total number of species.

[7] Preston used base 2 logs to make his histogram bins, and his practice remains standard; we use the natural log.

Log-normal abundance distributions (Fig. 10.7)

We can plot tree species densities from Barro Colorado Island [36], which is aailable online, or in the `vegan` package. First we make the data available to us, then make a simple histogram.

```
> data(BCI)
> N <- sort(colSums(BCI), decr = TRUE)
> hist(N, main = NULL)
```

Next we make a *species–abundance distribution*, which is merely a histogram of the log-abundances (classically, base 2 logs, but we use base e). In addition, we add the normal probability density function, getting the mean and standard deviation from the data, and plotting the expected number of species by multiplying the densities by the total number of species.

```
> hist(log(N), xlab = "Log Density Classes", main = NULL)
> m.spp <- mean(log(N))
> sd.spp <- sd(log(N))
> R <- length(N)
> curve(dnorm(x, m.spp, sd.spp) * R, 0, 8, add = T)
```

Next we create the *rank–abundance distribution*, which is just a plot of log-abundances *vs.* ranks.

```
> plot(log(N), type = "b", ylim = c(0, 8), main = NULL,
+     xlab = "Species Rank", ylab = "Log Density")
> ranks.lognormal <- R * (1 - pnorm(log(N), m.spp,
+     sd.spp))
> lines(ranks.lognormal, log(N))
```

We can think of the rank of species i as the total number of species that are more abundant than species i. This is essentially the opposite (or complement) of the integral of the species–abundance distribution [129]. That means that if we can describe the species abundance distribution with the normal density function, then 1-cumulative probability function is the rank.

10.3.2 Other distributions

Well over a dozen other types of abundance distributions exist to describe abundance patterns, other than the log-normal [124]. They can all be represented as *rank–abundance distributions*.

The *geometric distribution*[8] (or pre-emption niche distribution) reflects a simple idea, where each species pre-empts a constant fraction of the remaining niche space [129, 145]. For instance, if the first species uses 20% of the niche space, the second species uses 20% of the remaining 80%, etc. The frequency of the ith most abundant species is

[8] This probability mass function, $\mathrm{P}_i = d\,(1-d)^{i-1}$, is the probability distribution of the number of attempts, i, needed for one success, if the independent probability of success on one trial is d.

$$N_i = \frac{N_T}{C} d\,(1-d)^{i-1} \tag{10.6}$$

where d is the abundance of the most common species, and C is just a constant to make $\sum N_i = N_T$, where $C = 1 - (1-d)^{S_T}$. Thus this describes the geometric rank–abundance distribution.

The *log-series distribution* [55] describes the frequency of species with n individuals,

$$F\,(S_n) = \frac{\alpha x^n}{n} \tag{10.7}$$

where α is a constant that represents diversity (greater α means greater diversity); the α for a diverse rainforest might be 30–100. The constant x is a fitted, and it is always true that $0.9 < x < 1.0$ and x increases toward 0.99 as $N/S \rightarrow 20$ [124]. x can be estimated from $S/N = [(1-x)/x] \cdot [-\ln(1-x)]$. Note that this is not described as a rank–abundance distribution, but species abundances can nonetheless be plotted in that manner [129].

The log-series rank–abundance distribution is a bit of a pain, relying on the standard exponential integral [129], $E_1(s) = \int_s^\infty exp(-t)/t\,dt$. Given a range of N, we calculate ranks as

$$F\,(N) = \alpha \int_s^\infty exp(-t)/t\,dt \tag{10.8}$$

where we can let $t = 1$ and $s = N \log\,(1 + \alpha/N_T)$.

The log-series distribution has the interesting property that the total number of species in a sample of N individuals would be $S_T = \alpha \log(1 + N/\alpha)$. The parameter α is sometimes used as a measure of diversity. If your data are log-series distributed, then α is approximately the number of species for which you expect 1 individual, because $x \approx 1$. Two very general theories predict a log-series distribution, including neutral theory, and maximum entropy. Oddly, these two theories both predict a log-series distirbution, but make opposite assumptions about niches and individuals (see next section).

MacArthur's *broken stick distribution* is a classic distribution that results in a very even distribution of species abundances [118]. The number of individuals of each species i is

$$N_i = \frac{N_T}{S_T} \sum_{n=i}^{S_T} \frac{1}{n} \tag{10.9}$$

where N_T and S_T are the total number of individuals and species in the sample, respectively. MacArthur described this as resulting from the simultaneous breakage of a stick at random points along the stick. The resulting size fragments are the N_i above. MacArthur's broken stick model is thus both a *stochastic* and a *deterministic* model. It has a simulation (stick breakage) that is the direct analogue of the deterministic analytical expression.

Other similarly tactile stick-breaking distributions create a host of different rank–abundance patterns [124, 206]. In particular, the stick can be broken *sequentially*, first at one random point, then at a random point along one of two newly broken fragments, then at an additional point along any one of the *three* broken fragments, *etc.*, with $S_T - 1$ breaks creating S_T species. The critical

difference between the models then becomes *how each subsequent fragment is selected*. If the probability of selecting each fragment is related directly to its size, then this becomes identical to MacArthur's broken stick model. On the other hand, if each subsequent piece is selected randomly, regardless of its size, then this results in something very similar to the log-normal distribution [196, 205]. Other variations on fragment selection generate other patterns [206].

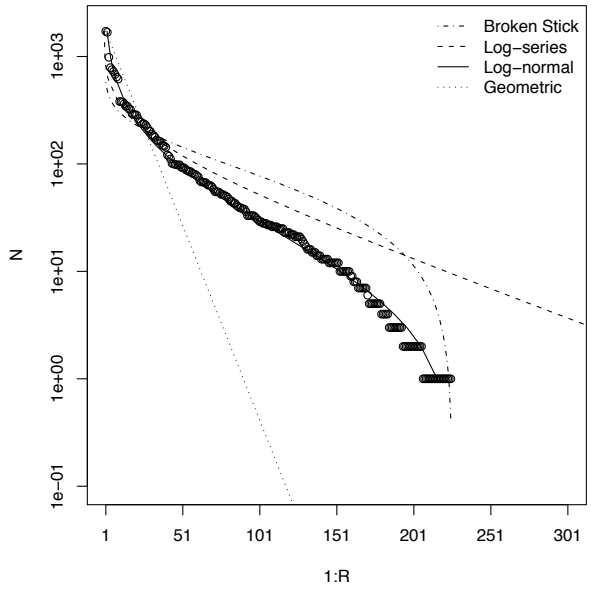


Fig. 10.8: A few common rank–abundance distributions, along with the BCI data [36]. The log-normal curve is fit to the data, and the broken stick distribution is always determined by the number of species. Here we let the geometric distribution be determined by the abundance of the most common species. The log-series was plotted so that it matched qualitatively the most abundant species.

Generating other rank–abundance distributions)

We can illustrate the above rank–abundance distributions as they might relate to the BCI tree data (see previous code). We start with MacArthur's broken stick model. We use cumulative summation backwards to add all $1/n_i$, and then re-sort it by rank (cf. eq. 10.9).

```
> N <- sort(colSums(BCI), decr = TRUE)
> f1 <- sort(cumsum(1/(R:1)), decr = TRUE)
> Nt <- sum(N)
> NMac <- Nt * f1/sum(f1)
```

Next, we create the geomtric rank–abundance distribution, where we let the BCI data tell us d, the density of the most abundant species; therefore we can multiply these by N_T to get expected abundances.

```
> d <- N[1]/Nt
> Ngeo.f <- d * (1 - d)^(0:(R - 1))
> Ngeo <- Nt * Ngeo.f
```

Last, we generate a log-series relevant to the BCI data. First, we use the `optimal.theta` function in the `untb` package to find a relevant value for Fisher's α. (See more and θ and α below under neutral theory).

```
> library(untb)
> alpha <- optimal.theta(N)
```

To calculate the rank abundance distribution for the log-series, we first need a function for the "standard exponential integral" which we then integrate for each population size.

```
> sei <- function(t = 1) exp(-t)/t
> alpha <- optimal.theta(N)
> ranks.logseries <- sapply(N, function(x) {
+     n <- x * log(1 + alpha/Nt)
+     f <- integrate(sei, n, Inf)
+     fv <- f[["value"]]
+     alpha * fv
+ })
```

Plotting other rank–abundance distributions (Fig. 10.8)

Now we can plot the BCI data, and all the distributions, which we generated above. Note that for the log-normal and the log-series, we calculated ranks, based on the species–abundance distributions, whereas in the standard form of the geometric and broken stick distributions, the expected abundances are calculated, in part, from the ranks.

```
> plot(1:R, N, ylim = c(0.1, 2000), xlim = c(1,
+     301), axes = FALSE, log = "y")
> axis(2)
> axis(1, 1 + seq(0, 300, by = 50))
> box()
> lines(1:R, NMac, lty = 4)
> lines(1:R, Ngeo, lty = 3)
> lines(ranks.logseries, N, lty = 2)
> lines(ranks.lognormal, N, lty = 1)
> legend("topright", c("Broken Stick", "Log-series",
+     "Log-normal", "Geometric"), lty = c(4, 2,
+     1, 3), bty = "n")
```

Note that we have not *fit* the the log-series or geometric distributions to the data, but rather, placed them in for comparison. Properly fitting curves to distributions is a picky business [124, 151], especially when it comes to species abundance distributions.

10.3.3 Pattern *vs.* process

Note the resemblance between stick-breaking and niche evolution — if we envision the whole stick as all of the available niche space, or energy, or limiting resources, then each fragment represents a portion of the total occupied by each species. Thus, various patterns of breakage act as models for niche partitioning and relative abundance patterns. Other biological and stochastic processes create specific distributions. For instance, completely random births, deaths, migration, and speciation will create the log-series distribution and the log-normal-like distributions (see *neutral theory* below). We noted above that independent, multiplicatively interacting factors can create the log-normal distribution.

On the other hand, all of these abundance distributions should probably be used primarily to describe *patterns* of commonness, rarity, and not to infer anything about the processes creating the patterns. These graphical descriptions are merely attractive and transparent ways to illustrate the abundances of the species in your sample/site/experiment.

The crux of the issue is that different *processes* can cause the same abundance distribution, and so, sadly, *we cannot usually infer the underlying processes from the patterns we observe*. That is, correlation is different than causation. Abundances of real species, in nature and in the lab, are the result of *mechanistic processes*, including as those described in models of abundance

distributions. However, we cannot say that a particular pattern was the result of a particular process, based on the pattern alone. Nonetheless, they are good summaries of ecological communities, and they show us, in a glance, a lot about *diversity*.

Nonetheless, interesting questions remain about the relations between patterns and processes. Many ecologists would argue that describing patterns and predicting them are the most important goals of ecology [156], while others argue that process is all important. However, both of these camps would agree about the fallacy of drawing conclusions about processes based on pattern — nowhere in ecology has this fallacy been more prevalent than with abundance distributions [66].

10.4 Neutral Theory of Biodiversity and Biogeography

One model of abundance distributions is particularly important, and we elaborate on it here. It is referred to variously as the *unified neutral theory of biodiversity and biogeography* [81], or often "neutral theory" for short. Neutral theory is important because it does much more than most other models of abundance distributions. It is a testable theory that makes quantitative predictions across several levels of organizations, for both evolutionary and ecological processes.

Just as evolutionary biology has neutral theories of gene and allele frequencies, ecology has neutral theories of population and community structure and dynamics [9, 22, 81]. Neutral communities of species are computationally and mathematically related and often identical to models of genes and alleles [53] (Table 10.2). Thus, lessons you have learned about genetic drift often apply to neutral models of communities. Indeed, neutral ecological dynamics at the local scale are often referred to as *ecological drift*, and populations change via demographic stochasticity.[9]

Stephen Hubbell proposed his "unified neutral theory of biodiversity and biogeography" (hereafter NTBB, [81]) as a null model of the dynamics of individuals, to help explain relative abundances of tropical trees. Hubbell describes it as a direct descendant of MacArthur and Wilson's theory of island biogeography [121, 122] (see below, species–area relations). Hubbell proposed it both as a null hypothesis and also — and this is the controversial part — as a model of community dynamics that closely approximates reality.

The relevant "world" of the NTBB is a metacommunity (Fig. 10.9), that is, a collection of similar local communities connected by dispersal [102].[10] The metacommunity is populated entirely by individuals that are functionally identical. The NTBB is a theory of the *dynamics of individuals*, modeling individual

[9] Some have called neutral theory a *null model*, but others disagree [61], describing distinctions between dynamical, process-based neutral models with fitted parameters, *vs.* static data-based null models [60]. Both can be used as benchmarks against which to measure other phenomena. Under those circumstances, I suppose they might both be null hypotheses.

[10] The metacommunity concept is quite general, and a neutral metacommunity is but one caricature [102].

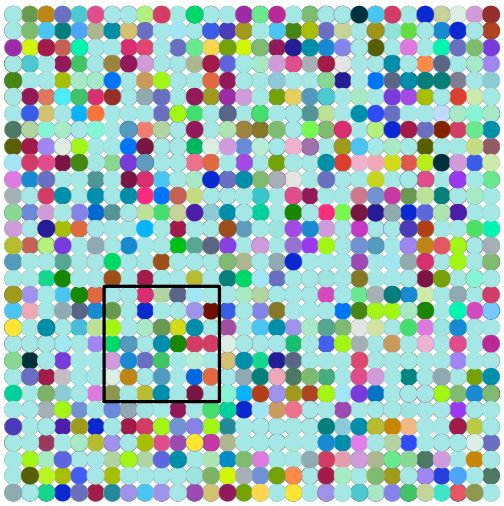


Fig. 10.9: A cartoon of a local community of forest canopy trees (small box) nested inside part of the metacommunity of a tropical forest. The true metaccommunity would extend far beyond the boundaries of this figure to include the true potential source of propagules. Shades of grey indicate different species. The local community is a sample of the larger community (such as the 50 ha forest dynamics plot on BCI) and receives migrants from the metacommunity. Mutation gives rise to new species in the metacommunity. For a particular local community, such as a 50 ha plot on an island in the Panama canal, the metacommunity will include not only the surrounding forest on the island, but also Panama, and perhaps much of the neotropics [86].

births, deaths, migration and mutation. It assumes that within a guild, such as late successional tropical trees, species are essentially neutral with respect to their fitness, that is, they exhibit *fitness equivalence*. This means that the probabilities of birth, death, mutation and migration are *identical for all individuals*. Therefore, changes in population sizes occur via *random walks*, that is, via stochastic increases and decreases with time step (Fig. 10.10). Random walks do not imply an absence of competition or other indirect enemy mediated negative density dependence. Rather, competition is thought to be diffuse, and equal among individuals. We discuss details of this in the context of simulation. Negative density dependence arises either through a specific constraint on the total number of individuals in a community [81], or as traits of individuals related to the probabilities of births, deaths, and speciation [214].

A basic paradox of the NTBB, is that in the absence of migration or mutation, diversity gradually declines to zero, or monodominance. A random walk due to fitness equivalence will eventually result in the loss of all species except one.[11] However, the loss of diversity in any single local community is predicted to be very, very slow, and is countered by immigration and speciation (we dis-

[11] This is identical to allele frequency in genetic drift.

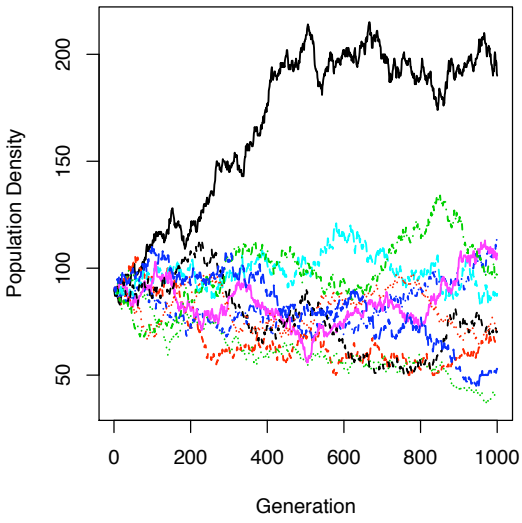


Fig. 10.10: Neutral ecological drift. Here we start with 10 species, each with 90 individuals, and let their abundances undergo random walks within a finite local community, with no immigration. Here, one generation is equal to nine deaths and nine births. Note the slow decline in unevennes — after 1000 deaths, no species has become extinct.

cuss more details below). Thus, species do not increase deterministically when rare — this makes the concept of coexistence different than the stable coexistance criteria discussed in previous chapters. Coexistance here *is not stable* but rather only a stochastic event with a limited time horizon which is balanced by the emergence of new species.

If all communities are thought to undergo random walks toward monodominance, how is diversity maintained in any particular region? Two factors maintain species in any local community. First, immigration into the local community from the metacommunity can bring in new species. Even though each local community is undergoing a random walk toward monodominance, each local community may become dominated by any one of the species in the pool because all species have equal fitness. Thus separate local communities are predicted to become dominated by different species, and these differences among local communities help maintain diversity in the metacommunity landscape.[12] Second, at longer time scales and larger spatial scales, speciation (i.e., mutation and lineage-splitting) within the entire metacommunity maintains biodiversity. Mutation and the consequent speciation provide the ultimate source of variation. Random walks toward extinction in large communities are *so* lengthy that the extinctions are balanced by speciation.

[12] This is the same as how genetic drift operates across subpopulations connected by low rates of gene exchange.

Introducing new symbols and jargon for ecological neutral theory, we state that the diversity of the metacommunity, θ, is a function of the number of individuals in the metacommunity, J_M, and the per capita rate at which new species arise (via mutation) ν ($\theta = 2J_M\nu$; Table 10.2). A local community undergoes ecological drift; drift causes the slow loss of diversity, which is balanced by a per capita (J_L) immigration rate m.

Table 10.2: Comparison of properties and jargon used in ecological and population genetic neutral theory (after Alonso et al. [1]). Here x is a continuous variable for relative abundance of a species or allele ($0 \leq x \geq 1$, $x = n/J$). Note this is potentially confused with Fisher's log-series $\langle \phi_n \rangle = \theta x^n/n$, which is a discrete species abundance distribution in terms of numbers of individuals, n (not relative abundance), and where $x = b/d$.

Property	**Ecology**	**Population Genetics**
Entire System (size)	Metacommunity (J_M)	Population (N)
Nested subsystem (size)	Local community (J_L)	Subpop. or Deme (N)
Smallest neutral system unit	Individual organism	Individual gene
Diversity unit	Species	Allele
Stochastic process	Ecological drift	Genetic drift
Generator of diversity (rate symbol)	Speciation (ν)	Mutation (μ)
Fundamental diversity number	$\theta \approx 2J_M\nu$	$\theta \approx 4N\mu$
Fundamental dispersal number	$I \approx 2J_L m$	$\theta \approx 4Nm$
Relative abundance distribution ($\Phi(x)$)	$\frac{\theta}{x}(1-x)^{\theta-1}$	$\frac{\theta}{x}(1-x)^{\theta-1}$
Time to common ancestor	$\frac{-J_M x}{1-x}\log x$	$\frac{-Nx}{1-x}\log x$

It turns out that the abundance distribution of an infinitely large metacommunity is Fisher's log-series distribution, and that θ of neutral theory is α of Fisher's log-series [2, 105, 215]. However, in any one local community, random walks of rare species are likely to include zero, and thus become extinct in the local community by chance alone. This causes a deviation from Fisher's log-series in any *local community* by reducing the number of the the rarest species below that predicted by Fisher's log-series. In particular, it tends to create a log-normal–like distribution, much as we often see in real data (Fig. 10.8). These theoretical findings and their match with observed data are thus consistent with the hypothesis that communities may be governed, in some substantive part, by neutral drift and migration.

Both theory and empirical data show that species which coexist may be more similar than predicted by chance alone [103], and that similarity (i.e., fitness equality) among species helps maintain higher diversity than would otherwise be possible [27]. Chesson makes an important distinction between *stabilizing mechanisms*, which create attractors, and *equalizing mechanisms*, which reduce differences among species, slow competitive exclusion and facilitate stabilization [27].

The NTBB spurred tremendous debate about the roles of chance and determinism, of dispersal-assembly and niche-assembly, of evolutionary processes in ecology, and how we go about "doing community ecology" (see, e.g., articles in *Functional Ecology*, 19(1), 2005; *Ecology*, 87(6), 2006). This theory in its narrowest sense has been falsified with a few field experiments and observation studies [32,223]. However, the *degree* to which stochasticity and dispersal *versus* niche partitioning structure communities remains generally unknown. Continued refinements and elaboration (e.g., [2, 52, 64, 86, 164, 216]) seem promising, continuing to intrigue scientists with different perspectives on natural communities [86,99]. Even if communities turn out to be completely non-neutral, NTBB provides a baseline for community dynamics and patterns that has increased the rigor of evidence required to demonstrate mechanisms controlling coexistence and diversity. As Alonso *et al.* state, "... good theory has more predictions per free parameter than bad theory. By this yeardstick, neutral theory fares fairly well" [1].

10.4.1 Different flavors of neutral communities

Neutral dynamics in a local community can be simulated in slightly different ways, but they are all envisioned some type of *random walk*. A random walk occurs when individuals reproduce or die at random, with the consequence that each population increases or decreases by chance.

The simplest version of a random walk assumes that births and deaths and consequent increases and decreases in population size are equally likely and equal in magnitude. A problem with this type of random walk is that a community can increase in size (number of individuals) without upper bound, or can disappear entirely, by chance. We know this doesn't happen, but it is a critically important first step in conceptualizing a neutral dynamic [22].

Hubbell added another level of biological reality by fixing the total number of individuals in a local community, J_L, as constant. When an individual dies, it is replaced with another individual, thus keeping the population size constant. Replacements come from within the local community with probability $1-m$, and replacements come from the greater metacommunity with probability m. The dynamics of the metacommunity are so slow compared to the local community that we can safely pretend that it is fixed, unchanging.

Volkov *et al.* [215] took a different approach by assuming that each species undergoes independent *biased random walks*. We imagine that each species undergoes its own completely independent random walk, *as if* it is not interacting with any other species. The key is that the birth rate, b, is slightly *less* than the death rate, d — this bias toward death gives us the name *biased random walk*. In a deterministic model with no immigration or speciation, this would result in a slow population decline to zero. In a stochastic model, however, some populations will increase in abundance by chance alone. Slow random walks toward extinctions are balanced by speciation in the metacommunity (with probability ν).

If species all undergo independent biased random walks, does this mean species don't compete and otherwise struggle for existance? No. The *reason* that

$b < d$ is precisely because all species struggle for existance, and only those with sufficiently high fitness, *and which are lucky*, survive. Neutral theory predicts that it is these species that we observe in nature — those which are lucky, and also have sufficiently high fitness.

In the metacommunity, the average number of species, $\langle \phi_n^M \rangle$, with population size n is

$$\langle \phi_n^M \rangle = \theta \frac{x^n}{x} \tag{10.10}$$

where $x = b/d$, and $\theta = 2J_M\nu$ [215]. The M superscript refers to the metacommunity, and the angle brackets indicate merely that this is the average. Here b/d is barely less than one, because it a biased random walk which is then offset by speciation, ν. Now we see that this is exactly Fisher's log-series distribution (eq. 10.7), where that $x = b/d$ and $\theta = \alpha$. Volkov *et al.* thus show that in a neutral metacommunity, x has a biological interpretation.

The expected size of the entire metacommunity is simply the sum of all of the average species' n [215].

$$J_M = \sum_{n=1}^{\infty} n\langle \phi_n^M \rangle = \theta \frac{x}{1-x} \tag{10.11}$$

Thus the size of the metacommunity is an emergent property of the dynamics, rather than an external constraint. To my mind, it seems that the number of individuals in the metacommunity must result from responses of individuals *to* their external environment.

Volkov *et al.* went on to derive expressions for births, deaths, and average relative abundances in the local community [139, 215]. Given that each individual has an equal probability of dying and reproducing, and that replacements can also come from the metacommunity with a probability proportional to their abundance in the metacommunity, one can specify rules for populations in local communities of a fixed size. These are the probability of increase, $b_{n,k}$, or decrease, $d_{n,k}$, in a species, k, of population size n.

$$b_{n,k} = (1-m)\frac{n}{J_L}\frac{J_L - n}{J_L - 1} + m\frac{\mu_k}{J_M}\left(1 - \frac{n}{J_L}\right) \tag{10.12}$$

$$d_{n,k} = (1-m)\frac{n}{J_L}\frac{J_L - n}{J_L - 1} + m\left(1 - \frac{mu_k}{J_M}\right)\frac{n}{J_L} \tag{10.13}$$

These expressions are the sum of two joint probabilities, each of which is comprised of several independent events. These events are immigration, and birth and death of individuals of different species. Here we describe these probabilities. For a population of size n of species k, we can indicate per capita, per death probabilities including

- m, the probability that a replacement is immigrant from the metacommunity, and $1 - m$, the probability that the replacement is from the local community.

- n/J_L, the probability that an individual selected randomly from the local community belongs to species k, and $1-n/J_L$ or $(J-n)/(J_L)$, the probability that an individual selected randomly from the local community belongs to any species other than k.
- $(J-n)/(J_L-1)$, the conditional probability that, given that an individual of species k has already been drawn from the population, an individual selected randomly from the local community belongs to any species other than to species k.
- μ_k/J_M, the probability that an individual randomly selected from the metacommunity belongs to species k, and $1-n/J_M$, the probability that an individual randomly selected from the metacommunity belongs to any species other than k.

Each of these probabilities is the probability of some unspecified event — that event might be birth, death, or immigration.

Before we translate eqs. 10.12, 10.13 literally, we note that b and d each have two terms. The first term is for dynamics related to the local community, which happen with probability $1-m$. The second is related to immigration from the metacommunity which occurs with probability m. Consider also that if a death is replaced by a birth of the same species, or a birth is countered by a death of the same species, they cancel each other out, as if nothing ever happened. Therefore each term requires a probability related to species k and to non-k.

Eq. 10.12, $b_{n,k}$, is the probability that an individual will be added to the population of species k. The first term is the joint probability that an addition to the population comes from within the local community $(1-m)$ *and* the birth comes from species k (n/J_L) *and* there is a death of an individual of any other species $((J_L-n)/(J_L-1))$.[13] The second term is the joint probability that the addition to the population comes from the metacommunity via immigration (m) *and* that the immigrant is of species k (μ_k/J_M) *and* is not accompanied by a death of an individual of its species (n/J_L).

An individual may be subtracted from the population following similar logic. Eq. 10.13, $d_{n.k}$, is the probability that a death will remove an individual from the population of species k. The first term is the joint probability that the death occurs in species k (n/J_L) and the replacement comes from the local community $(1-m)$ and is some species other than k $((J_L-n)/(J_L-1))$. The second term is the joint probability that the death occurs in species k (n/J_L), and that it is replaced by an immigrant (m) *and* the immigrant is any species in the metacommunity other than k $(1-\mu_k/J_M)$.

10.4.2 Investigating neutral communities

Here we exxplore netural communities using the `untb` package, which contains a variety of functions for teaching and research on neutral theory.

[13] The denominator of the death probability is J_L-1 instead of J_L because we have already selected the first individual who will do the reproduction, so the total number of remaining individuals is J_L-1 rather than J_L; the number of non-k individuals remains J_L-n

Pure drift

After loading the package and setting graphical parameters, let's run a simulation of drift. Recall that drift results in the slow extinction of all but one species (Fig. 10.10). We start with a local community with 20 species, each with 25 individuals[14] The simulation then runs for 1000 generations (where 9/900 individuals die per generation).[15]

```
> library(untb)
> a <- untb(start = rep(1:20, 25), prob = 0, gens = 2500,
+     D = 9, keep = TRUE)
```

We `keep`, in a matrix, all 450 trees from each of the 1000 time steps so that we can investigate the properties of the community. The output is a matrix where each element is an integer whose value represents a species ID. Rows are time steps and we can think of columns as spots on the ground occupied by trees. Thus a particular spot of ground may be occupied by an individual of one species for a long time, and suddenly switch identity, because it dies and is replaced by an individual of another species. Thus the community always has 450 individuals (columns), but the identities of those 450 change through time, according to the rules laid out in eqs. 10.12, 10.13. Each different species is represented by a different integer; here we show the identitites of ten individuals (columns) for generations 901–3.

```
> (a2 <- a[901:903, 1:10])
```

```
     [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
[1,]    7   15    4   15   11    7   15   14   20    20
[2,]    7   15    4   15   11    7   15   14   20    20
[3,]    7   15    4   15   11    7   15   14   20    20
```

Thus, in generation 901, tree no. 1 is species 7 and in generation 902, tree no. 3 is species 4.

We can make pretty pictures of the communities at time steps 1, 100, and 2000, by having a single point for each tree, and coding species identity by shades of grey.[16]

```
> times <- c(1, 50, 2500)
> sppcolors <- sapply(times, function(i) grey((a[i,
+     ] - 1)/20))
```

This function applies to the community, at each time step, the `grey` function. Recall that species identity is an integer; we use that integer to characterize each species' shade of grey.

Next we create the three graphs at three time points, with the appropriate data, colors, and titles.

---

[14] This happens to be the approximate tree density (450 trees ha^{-1}, for trees > 10 cm DBH) on BCI.

[15] Note that `display.untb` is great for pretty pictures, whereas `untb` is better for more serious simulations.

[16] We could use a nice color palette, `hcl`, based on hue, chroma, and luminance, for instance `hcl(a[i,]*30+50)`

```
> layout(matrix(1:3, nr = 1))
> par(mar = c(1, 1, 3, 1))
> for (j in 1:3) {
+     plot(c(1, 20), c(1, 25), type = "n", axes = FALSE)
+     points(rep(1:20, 25), rep(1:25, each = 20),
+         pch = 19, cex = 2, col = sppcolors[, j])
+     title(paste("Time = ", times[j], sep = ""))
+ }
```

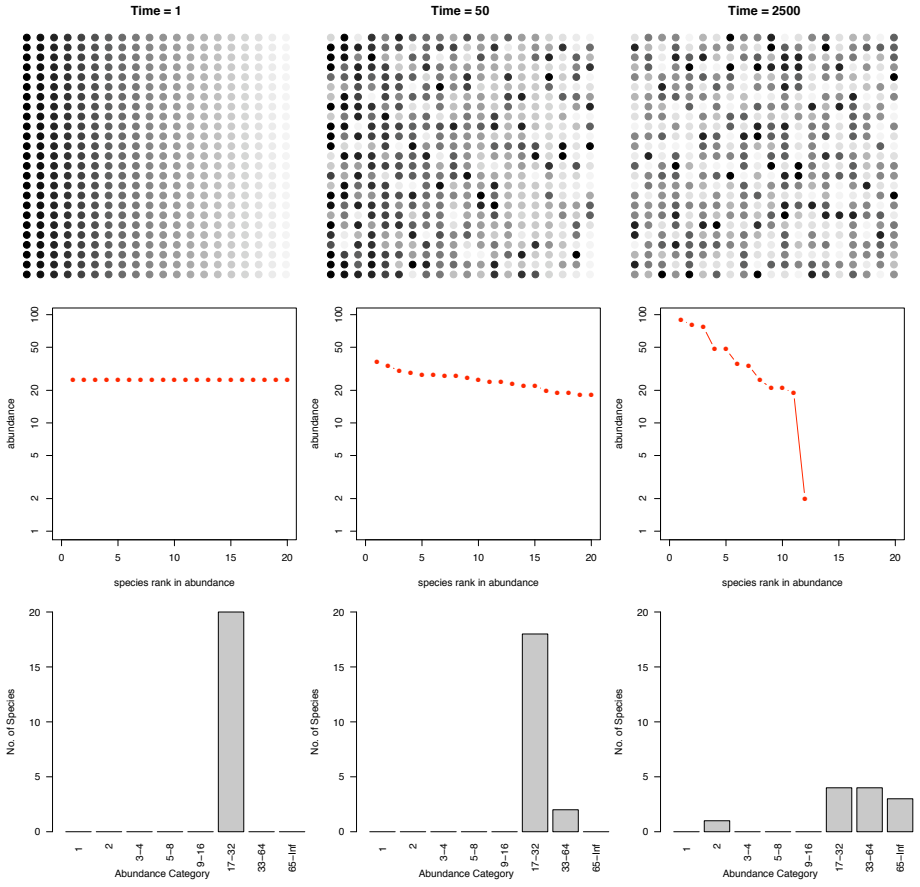


Fig. 10.11: Three snapshots of one community, drifting through time. Shades of grey represent different species. Second row contains rank abundance distributions; third row contains species abundance distributions. Drift results in the slow loss of diversity.

From these graphs (Fig. 10.11), we see that, indeed, the species identity of the 450 trees changes through time. Remember that this is *not spatially explicit* — we are not stating that individual *A* is next, or far away from, individual *B*.

Rather, this is a spatially implicit representation — all individuals are characterized by the same probaibllities.

Next let's graph the rank abundance distribution of these communities (Fig. 10.11). For each time point of interest, we first coerce each "ecosystem" into a `count` object, and then plot it. Note that we plot them all on a common scale to facilitate comparison.

```
> layout(matrix(1:3, nr = 1))
> for (i in times) {
+     plot(as.count(a[i, ]), ylim = c(1, max(as.count(a[times[3],
+         ]))), xlim = c(0, 20))
+     title(paste("Time = ", i, sep = ""))
+ }
```

Next we create species abundance distributions, which are histograms of species' abundances (Fig. 10.11). If we want to plot them on a common scale, it takes a tad bit more effort. We first create a matrix of zeroes, with enough columns for the last community, use `preston` to create the counts of species whose abundances fall into logarithmic bins, plug those into the matrix, and label the matrix columns with the logarithmic bins.

```
> out <- lapply(times, function(i) preston(a[i,
+     ]))
> bins <- matrix(0, nrow = 3, ncol = length(out[[3]]))
> for (i in 1:3) bins[i, 1:length(out[[i]])] <- out[[i]]
> bins
```

```
     [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8]
[1,]    0    0    0    0    0   20    0    0
[2,]    0    0    0    0    0   18    2    0
[3,]    0    1    0    0    0    4    4    3
```

```
> colnames(bins) <- names(preston(a[times[3], ]))
```

Finally, we plot the species–abundance distributions.

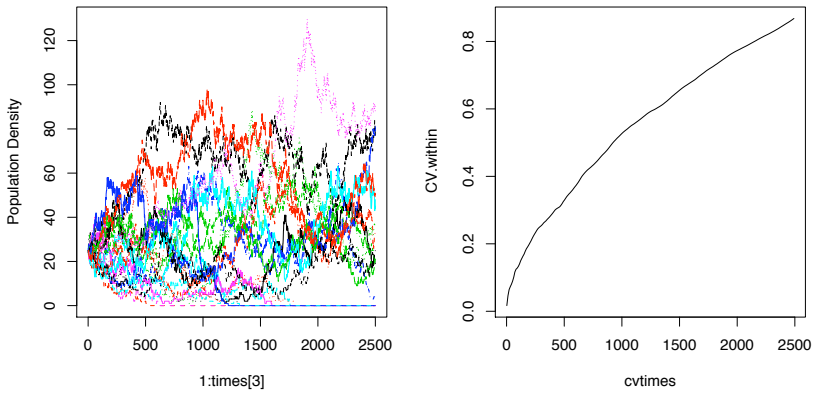
```
> layout(matrix(1:3, nr = 1))
> for (i in 1:3) {
+     par(las = 2)
+     barplot(bins[i, ], ylim = c(0, 20), xlim = c(0,
+         8), ylab = "No. of Species", xlab = "Abundance Category")
+ }
```

Bottom line: *drift causes the slow loss of species from local communities* (Fig. 10.11). What is not illustrated here is that without dispersal, drift will cause different species to become abundant in different places because the variation is random. In that way, drift maintains diversity at large scales, in the metacommunity. Last, low rates of dispersal among local communities maintains some diversity in local communities without changing the entire metacommunity into a single large local community. Thus dispersal limitiation, but not its absence, maintains diversity.

Next, let's examine the dynamics through time. We will plot individual species trajectories (Fig. 10.12a).

```
> sppmat <- species.table(a)
> matplot(1:times[3], sppmat[1:times[3], ], type = "l",
+     ylab = "Population Density")
```

The trajectories all start at the same abundance, but they need not have. The trajectories would still have the same drifting quality.



(a) 20 populations (b) Average variability

Fig. 10.12: Dynamics and average variation within populations. In random walks, average variation (measured with the coefficient of variation) increases with time.

For random walks in general, the observed variance and coefficient of variation ($CV = \hat{\sigma}/\bar{x}$) of a population will grow over time [32]. Here we calculate the average population CV of cumulative observations (note the convenient use of nested `(s)apply` functions). Let's calculate the CV's for every tenth time step.

```
> cvtimes <- seq(2, 2500, by = 10)
> CV.within <- sapply(cvtimes, function(i) {
+     cvs <- apply(sppmat[1:i, ], 2, function(x) sd(x)/mean(x))
+     mean(cvs)
+ })
```

Now plot the average CV through time. The average observed CV should increase (Fig. 10.12b).

```
> plot(cvtimes, CV.within, type = "l")
```

This shows us that the populations never approach an equilibrium, but wander aimlessly.

Real data

Last, we examine a BCI data set [36]. We load the data (data from 50 1 ha plots × 225 species, from the `vegan` package), and sum species abundances to get each species total for the entire 50 h plot (Fig. 10.13a).

```
> library(vegan)
> data(BCI)
> n <- colSums(BCI)
> par(las = 2, mar = c(7, 4, 1, 1))
> xs <- plot(preston(rep(1:225, n), n = 12, original = TRUE))
```

We would like to estimate θ and m from these data, but that requires specialized software for any data set with a realistic number of individuals. Specialized software would provide maximum likelihood estimates (in a reasonable amount of computing time) for m and θ for large communities [51,69,86]. The BCI data have been used repeatedly, so we rely on estimates from the literature ($\theta \approx 48$, $m \approx 0.1$) [51,215]. We use the approach of Volkov et al. [215] to generate expected species abundances (Fig. 10.13a).

```
> v1 <- volkov(sum(n), c(48, 0.1), bins = TRUE)
> points(xs, v1[1:12], type = "b")
```

More recently, Jabot and Chave [86] arrived at estimates that differed from previous estimates by orders of magnitude (Fig. 10.13b). Their novel approach estimated θ and m were derived from *both* species abundance data and from phylogenetic data ($\theta \approx 571, 724$, $m \approx 0.002$). This is possible because neutral theory makes a rich array of predictions, based on both ecological and evolutionary processes. Their data were only a little bit different (due to a later census), but their analyses revealed radically different estimates, with a much greater diversity and larger metacommunity (greater θ), and much lower immigration rates (smaller m).

Here we derive expected species abundance distributions. The first is based soley on census data, and is similar to previous expections (Fig. 10.13b).

```
> v2 <- volkov(sum(n), c(48, 0.14), bins = TRUE)
> xs <- plot(preston(rep(1:225, n), n = 12, original = TRUE),
+     col = 0, border = 0)
> axis(1, at = xs, labels = FALSE)
> points(xs, v2[1:12], type = "b", lty = 2, col = 2)
```

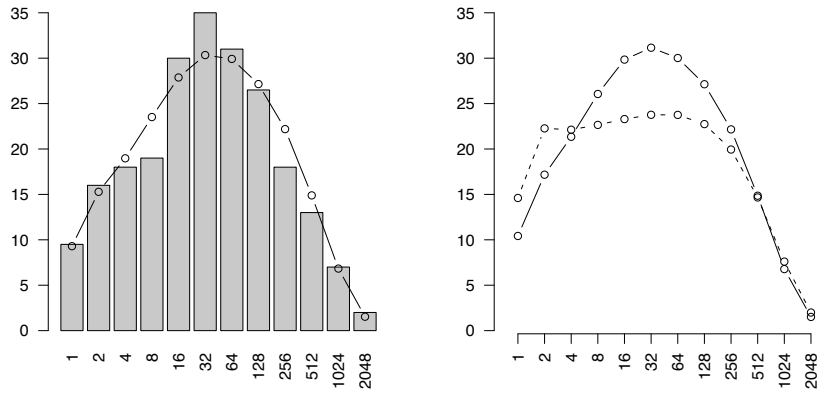
However, when they also included phylogenetic data, they found very different expected species abundance distributions (Fig. 10.13b).

```
> v4 <- volkov(sum(n), c(571, 0.002), bins = TRUE)
> points(xs, v4[1:12], type = "b", lty = 4, col = 2)
```

If nothing else, these illustrate the effects of increasing θ and reducing m.

10.4.3 Symmetry and the rare species advantage

An important advance in neutral theory is the quantitfication of a *symmetric rare species advantage*. The symmetry hypothesis posits that all species have a *symmetrical rare species advantage* [214, 216]. That is, all species increase when rare to the same degree (equal negative density dependence). In a strict sense, all individuals remain the same in that their birth and death probabilities change with population size in the same manner for all species. This obviously



(a) Original estimates (b) Jabot and Chave estimates

Fig. 10.13: Species abundance distributions for BCI trees. (a) Data for histogram from [36], overlain with expected abundances with θ and m values fitted to the data [51,215]. (b) Jabot and Chave found that when they used only species abundances (as did previous investigators) their pattern was similar to previous findings (solid line). However, adding phylogenetic information led to very different expectations (dashed line).

reduces the chance of random walks to extinction, but is nonetheless the same among all species. Estimation of the magnitude of the rare species advantage is interesting addition to stepwise increasing complexity.

To sum up: it is safe to say that neutral theory has already made our thinking about community structure and dynamics more sophisticated and subtle, by extending island biogeography to individuals. The theory is providing quantitative, pattern-generating models, that are analogous to null hypotheses. With the advent of specialized software, theory is now becoming more useful in our analysis of data [51, 69, 86].

10.5 Diversity Partitioning

We frequently refer to biodiversity (i.e., richness, Simpson's, and Shannon-Wiener diversity) at different spatial scales as α, β, and γ diversity (Fig. 10.14).

- Alpha diversity, α, is the diversity of a point location or of a single sample.
- Beta diversity, β, is the diversity due to multiple localities; β diversity is sometimes thought of as *turnover* in species composition among sites, or alternatively as the number of species in a region that are *not* observed in a sample.
- Gamma diversity, γ, is the diversity of a region, or at least the diversity of all the species in a set of samples collected over a large area (with large extent relatve to a single sample).

Diversity across spatial scales can be further be *partitioned* in one of two ways, either using *additive* or *multiplicative* partitioning.

Additive partitioning [42, 43, 97] is represented as

$$\bar{\alpha} + \beta = \gamma \tag{10.14}$$

where $\bar{\alpha}$ is the average diversity of a sample, γ is typically the diversity of the pooled samples, and β is found by difference ($\beta = \gamma - \bar{\alpha}$). We can think of β as the *average* number of species *not* found in a sample, but which we know to be in the region. Additive partitioning allows direct comparison of average richness among samples at any hierarchical level of organization because all three measures of diversity (α, β, and γ) are *expressed in the same units*. This makes it analogous to partitioning variance in ANOVA. This is not the case for multiplicative partitioning diversity.

Partitioning can also be *multiplicative* [219],

$$\bar{\alpha}\beta = \gamma \tag{10.15}$$

where β is a conversion factor that describes the relative change in species composition among samples. Sometimes this type of β diversity is thought of as the number of different community types in a set of samples. However, one must use this interpretation with great caution, as either meaning of β diversity depends completely on the sizes or extent of the samples used for α diversity.

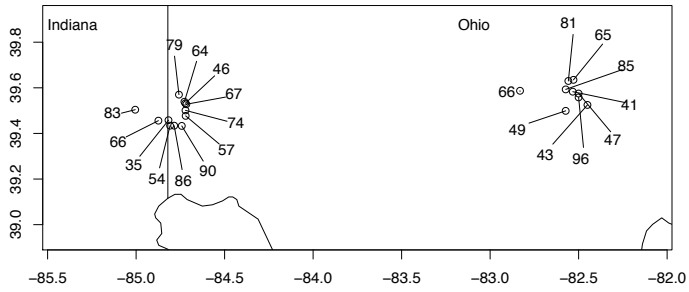


Fig. 10.14: Hierarchical sampling of moth species richness in forest patches in Indiana and Ohio, USA [198]. α-diversity is the diversity of a single site (richness indicated by numbers). γ-diversity is the total number of species found in any of the samples (here $\gamma = 230$ spp.). Additive β-diversity is the difference, $\gamma - \bar{\alpha}$, or the average number of species *not* observed in a single sample. Diversity partitioning can be done at two levels, sites within ecoregions and ecoregions within the geographic region (see example in text for details).

Let us examine the the limits of β diversity in extremely small and extremely large samples. Imagine that our sample units each contain, on average, one individual (and therefore one species) and we have 10^6 samples. If richness is our measure of diversity, then $\bar{\alpha} = 1$. Now imagine that in all of our samples we find a total of 100 species, or $\gamma = 100$. Our additive and multiplicative partitions would then be $\beta_A = 99$, and $\beta_M = 100$, respectively. If the size of the sample unit increases, each sample will include more and more individuals and therefore more of the diversity, and by definition, β will decline. If each sample gets large enough, then each sample will capture more and more of the species until a sample gets so large that it includes all of the species (i.e., $\bar{\alpha} \to \gamma$). At this point, $\beta_A \to 0$ and $\beta_M \to 1$.

Note that β_A and β_M do not change at the same rates (Fig. 10.15). When we increase sample size so that each sample includes an average of two species ($\bar{\alpha} = 2$), then $\beta_A = 98$ and $\beta_M = 50$. If each sample were big enough to have on average 50 species ($\bar{\alpha} = 50$), then $\beta_A = 50$ and $\beta_M = 2$. So, the β's do not change at the same rate (Fig. 10.15).

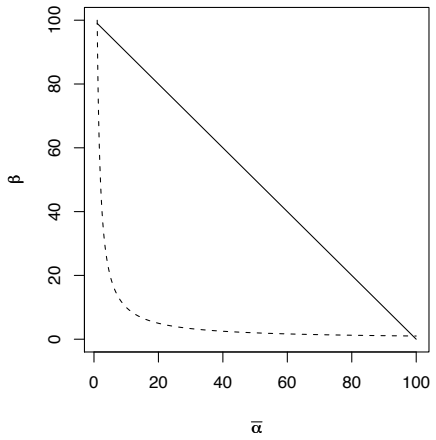


Fig. 10.15: Relations of β_A (with additive partitioning) and β_M (with multiplicative partitioning) to $\bar{\alpha}$, for a fixed $\gamma = 500$ species. In our example, we defined diversity as species richness, so the units of β_A and α are number of species per sample, and $\bar{\alpha}$ is the mean number of species in a sample.

Multiplicative β_M is sometimes thought of as the number of independent "communities" in a set of samples. This would make sense if our sampling regime were designed to capture representative parts of different communities. For example, if we sampled an elevational gradient, or a productivity gradient, and our smallest sample was sufficiently large so as to be representative of that point along the gradient[17] then β_M could provide a measure of the relative turnover in

[17] One type of sample that attempts to do this is a relevé.

composition or "number of different communities." However, we know that composition is also predicted to vary randomly across the landscape [36]. Therefore, if each sample is small, and not really representative of a "community," then the small size of the samples will inflate β_M and change the interpretation.

As an example, consider the BCI data, which consists of 50 contiguous 1 ha plots. First, we find γ (all species in the data set, or the number of columns), and $\bar{\alpha}$ (mean species number per 1 ha plot).

```
> (gamma <- dim(BCI)[2])

[1] 225

> (alpha.bar <- mean(specnumber(BCI)))

[1] 90.78
```

Next we find additive β-diversity and multiplicative β-diversity.

```
> (beta.A <- gamma - alpha.bar)

[1] 134.2

> (beta.M <- gamma/alpha.bar)

[1] 2.479
```

Now we interpret them. These plots are located in a relatively uniform tropical rainforest. Therefore, they each are samples drawn from a single community type. However, the samples are small. Therefore, each 1 ha plot (10^4 m^2 in size) misses more species than it finds, on average ($\beta_A > \bar{\alpha}$). In addition, $\beta_M = 2.48$, indicating a great deal of turnover in species composition. We could mistakenly interpret this as indicating something like ~ 2.5 independent community types in our samples. Here, however, we have a single community type — additive partitioning is a little simpler and transparent in its interpretation.

For other meanings of β-diversity, linked to neutral theory, see [36, 144].

10.5.1 An example of diversity partitioning

Let us consider a study of moth diversity by Keith Summerville and Thomas Crist [197, 198]. The subset of their data presented here consists of woody plant feeding moths collected in southwest Ohio, USA. Thousands of individuals were trapped in 21 forest patches, distributed in two adjacent ecoregions (12 sites - North Central Tillplain [NCT], and 9 sites - Western Allegheny Plateau [WAP], Fig. 10.14). This data set includes a total of 230 species, with 179 species present in the NCT ecoregion and 173 species present in the WAP ecoregion. From these subtotals, we can already see that each ecoregion had most of the combined total species (γ).

We will partition richness at three spatial scales: sites within ecoregions ($\bar{\alpha}_1$), ecoregions ($\bar{\alpha}_2$), and overall (γ). This will result in two β-diversities: β_1 among sites within each ecoregion, and β_2 between ecoregions. The relations among these are straightforward.

$$\bar{\alpha}_2 = \bar{\alpha}_1 + \beta_1 \tag{10.16}$$

$$\gamma = \bar{\alpha}_2 + \beta_2 \tag{10.17}$$

$$\gamma = \bar{\alpha}_1 + \beta_1 + \beta_2 \tag{10.18}$$

To do this in R, we merely implement the above equations using the data in Fig. 10.14 [198]. First, we get the average site richness, $\bar{\alpha}_1$. Because we have different numbers of individuals from different site, and richness depends strongly on the number of individuals in our sample, we may want to weight the sites by the number of individuals. However, I will make the perhaps questionable argument for the moment that because trapping effort was similar at all sites, we will not adjust for numbers of individuals. We will assume that different numbers of individuals reflect different population sizes, and let number of individuals be one of the local determinants of richness.

```
> data(moths)

> a1 <- mean(moths[["spp"]])
```

Next we calculate average richness richness for the ecoregions. Because we had 12 sites in NCT, and only nine sites in WAP for what might be argued are landscape constraints, we will use the weighted average richness, adjusted for the number of sites.[18] We also create an object for $\gamma = 230$.

```
> a2 <- sum(c(NCT = 179, WAP = 173) * c(12, 9)/21)
> g <- 230
```

Next, we get the remaining quantities of interest, and show that the partition is consistent.

```
> b1 <- a2 - a1
> b2 <- g - a2
> abg <- c(a1 = a1, b1 = b1, a2 = a2, b2 = b2, g = g)
> abg

    a1     b1     a2     b2      g
 65.43 111.00 176.43  53.57 230.00

> a1 + b1 + b2 == g

[1] TRUE
```

The partitioning reveals that β_1 is the largest fraction of overall γ-richness (Fig. 10.16). This indicates that in spite of the large distance between sampling areas located in different ecoregions, and the different soil types and associated flora, most of the variation occurs *among sites within regions*. If there had been a greater difference in moth community composition among ecoregions, then β_2-richness would have made up a greater proportion of the total.

[18] The arithmetic mean is $\sum a_i Y_i$, where all $a_i = 1/n$, and n is the total number of observations. A weighted average is the case where the a_i represent unequal *weights*, often the fraction of n on which each Y_i is based. In both cases, $\sum a = 1$.

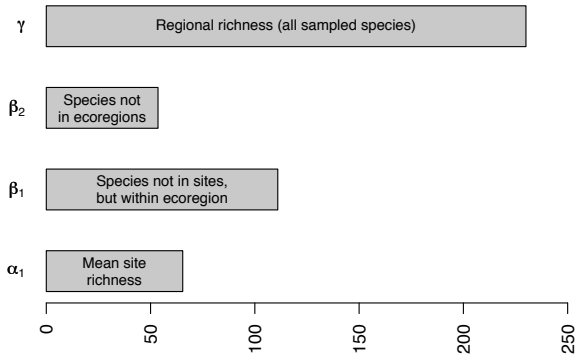


Fig. 10.16: Hierarchical partitioning of moth species richness in forest patches [198]. See Fig. 10.14 for geographical locations.

These calculations show us how simple this additive partition can be, although more complicated analyses are certainly possible. It can be very important to weight appropriately the various measures of diversity (e.g., the number of individuals in each sample, or number of samples per hierarchical level). The number of individuals in particular has a tremendous influence on richness, but has less influence on Simpson's diversity partitioning. The freely available PARTITION software will perform this additive partitioning (with sample sizes weights) and perform statistical tests [212].

10.5.2 Species–area relations

The relation between the number of species found in samples of different area has a long tradition [5, 10, 120–122, 169, 178], and is now an important part of the metastasizing subdiscipline of *macroecology* [15, 71].

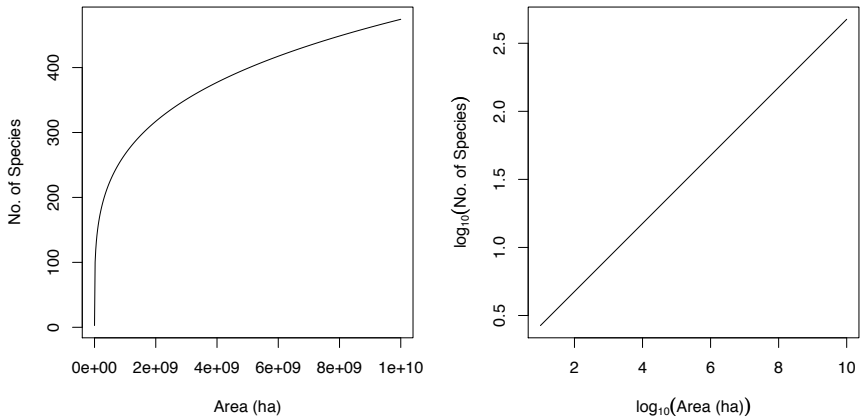
Most generally, the species–area relation (SAR) is simply an empirical pattern of the number of species found in patches of different size, plotted as a function of the sizes of the respective patches (Fig. 10.17). These patches may be isolated from each other, as islands in the South Pacific [121], or mountaintops covered in coniferous forest surrounded by a sea of desert [16], or calcareous grasslands in an agricultural landscape [68]. On the other hand, these patches might be nested sets, where each larger patch contains all others [41, 163].

Quantitatively, the relation is most often proposed as a simple power law,

$$R = cA^z \tag{10.19}$$

where R is the number of species in a patch of size A, and c and z are fitted constants. This is most often plotted as a log–log relation, which makes it linear.

$$\log(R) = b + zA \tag{10.20}$$



(a) Arithmetic scale (b) Logarithmic scale

Fig. 10.17: Power law species–area relations.

where b is the intercept (equal to $\log c$) and z is the slope.

Drawing power law species–area relations (Fig. 10.17)

Here we simply draw some species area curves.

```
> A <- 10^1:10
> c <- 1.5
> z <- 0.25
> curve(c * x^z, 10, 10^10, n = 500, ylab = "No. of Species",
+     xlab = "Area (ha)")
```

```
> A <- 10^1:10
> c <- 1.5
> z <- 0.25
> curve(log(c, 10) + z * x, 1, 10,
+     ylab = quote(log[10]("No. of Species")),
+     xlab = quote(log[10]("Area (ha)")))
```

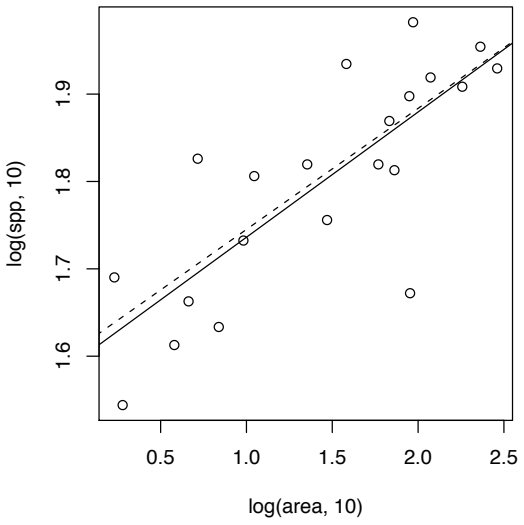


Fig. 10.18: Fitted power law species–area relations.

Fitting a species–area relation (Fig. 10.18)

Here we fit a species–area curve to data, and examine the slope. We could fit a nonlinear power relation ($y = cA^z$); this would be appropriate, for instance, if the residual noise around the line were of the same magnitude for all patch areas. We could use reduced major axis regression, which is appropriate when there is equivalent uncertainty or error on both x and y. Last (and most often), we could use a simple linear regression on the log-transformed data, which is appropriate when we know x to a high degree of accuracy, but measure y with some error, and the transformation causes the residual errors to be of similar magnitude at all areas. We start with the last (log-transformed). Here we plot the data, and fit a linear model to the common log-transformed data.

```
> plot(log(spp, 10) ~ log(area, 10), moths)
> mod <- lm(log(spp, 10) ~ log(area, 10), data = moths)
> abline(mod)
```

Next we fit the nonlinear curve to the raw data, and overlay that fit (on the log scale).

```
> mod.nonlin <- nls(spp ~ a * area^z, start = list(a = 1,
+     z = 0.2), data = moths)
> curve(log(coef(mod.nonlin)[1], 10) + x * coef(mod.nonlin)[2],
+     0, 3, add = TRUE, lty = 2)
```

Assessing species–area relations

Note that in Figure 10.18, the fits differ slightly between the two methods. Let's compare the estimates of the slope — we certainly expect them to be similar, given the picture we just drew.

```
> confint(mod)

                2.5 % 97.5 %
(Intercept)   1.50843 1.6773
log(area, 10) 0.09026 0.1964

> confint(mod.nonlin)

      2.5%   97.5%
a 31.61609 50.1918
z  0.08492  0.1958
```

We note that the estimates of the slopes are quite similar. Determining the better of the two methods (or others) is beyond the scope of this book, but be aware that methods can matter.

The major impetus for the species–area relation came from (i) Preston's work on connections between the species–area relation and the log-normal species abundance distribution [169,170], and (ii) MacArthur and Wilson's theory of island biogeography[19] [122].

Preston posited that, given the log-normal species abundance distributions (see above), then increasingly large samples should accumulate species at particular rates. Direct extensions of this work, linking neutral theory and maximum entropy theory to species abundances and species–area relations continues today [10, 71, 81]

Island biogeography

MacArthur and Wilson proposed a simple theory wherein the number of species on an oceanic island was a function of the immigration rate of new species, and extinction rate of existing species (Fig. 10.19). The number of species at any one time was *a dynamic equilibrium*, resulting from both slow inevitable extinction and slow continual arrival of replacements. Thus species composition on the island was predicted to change over time, that is, to undergo turnover.

Let us imagine immigration rate, y, as a function of the number of species already on an island, x (Fig. 10.19). This relation will have a negative slope, because as the number of species rises, that chance that a new individual actually represents a new species will decline. The immigration rate will be highest when there are no species on the island, $x = 0$, and will fall to zero when every conceivable species is already there. In addition, the slope should be decelerating

[19] (Originally proposed in a paper entitled "An Equilibrium Theory of Insular Zoogeography" [121])

(concave up) because some species will be much more likely than others to immigrate. This means that the immigration rate drops steeply as the most likely immigrants arrive, and only the unlikely immigrants are missing. Immigrants may colonize quickly for two reasons. First, as Preston noted, some species are simply much more common than others. Second, some species are much better dispersers than others.

Now let us imagine extinction rate, y, as a function of the number of species, x (Fig. 10.19). This relation will have a positive slope, such that the probability of extinction increases with the number of species. This is predicted to have an accelerating slope (concave-up), for essentially the same sorts of reasons governing the shape of the immigration curve: Some species are less common than others, and therefore more likely to become extinct do to demographic and environmental stochasticity, and second, some species will have lower fitness for any number of reasons. As the number of species accumulates, the more likely it will become that these extinction-prone species (rare and/or lower fitness) will be present, and therefore able to become extinct.

The *rate of change* of the number of species on the island, ΔR, will be the difference between immimigration, I, and extinction, E, or

$$\Delta R = I - E. \tag{10.21}$$

When $\Delta R = 0$, we have an equilibrium. If we knew the quantitative form of immigration and extinction, we could solve for the equilibrium. That equilibrium would be the point on the x axis, R, where the two rates cross (Fig. 10.19).

In MacArthur and Wilson's theory of island biogeography, these rates could be driven by the *sizes* of the islands, where

- larger islands had lower extinction rates because of larger average population sizes.
- larger islands had higher colonization rates because they were larger targets for dispersing species.

The distance between an island and sources of propagules was also predicted to influence these rates.

- Islands closer to mainlands had higher colonization rates of new species because more propagules would be more likely to arrive there,
- The effect of area would be more important for islands far from mainlands than for islands close to mainlands.

Like much good theory, these were simple ideas, but had profound effects on the way ecologists thought about communities. Now these ideas, of dispersal mediated coexistence and landscape structure, continue to influence community ecologists [15, 81, 102].

Drawing immigration and extinction curves

It would be fun to derive a model of immigration and extinction rates from first principles [121], but here we can illustrate these relations with some simple

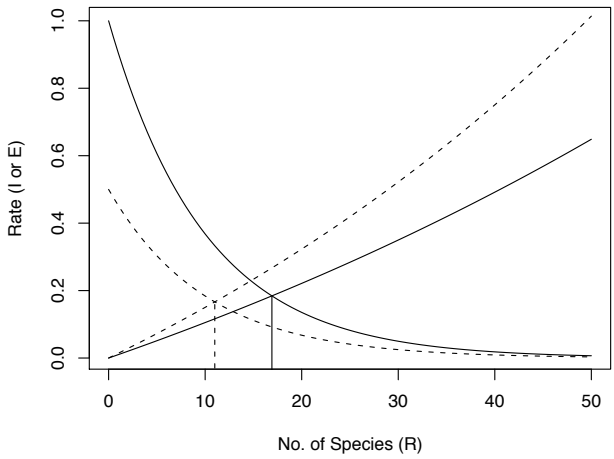


Fig. 10.19: Immigration and extinction curves for the theory of island biogeography. The declining curves represent immigration rates as functions of the number of species present on an island. The increasing curves represent extinction rates, also as functions of island richness. See text for discussion of the heights of these curves, i.e., controls on these rates. Here the dashed lines represent an island that is shrinking in size.

phenomenological graphs. We will assume that immmigration rate, I, can be represented as a simple negative exponential function $\exp(I_0 - iR)$, where I_0 is the rate of immigration to an empty island, and i is the per species negative effect on immigration rate.

```
> I0 <- log(1)
> b <- 0.1
> curve(exp(I0 - b * x), 0, 50, xlab = "No. of Species (R)",
+     ylab = "Rate (I or E)")
```

Note that extinction rate, E, must be zero if there are no species present. Imagine that extinction rate is a function of density and that average density declines as the number of species increases, or $\bar{N} = 1/R$.[20].

```
> d <- 0.01
> curve(exp(d * x) - 1, 0, 50, add = TRUE)
```

We subtract 1 merely to make sure that $E = 0$ when $R = 0$.

The number of species, R, will result from $\Delta R = 0 = I - E$, the point at which the lines cross.

$$I = e^{I_0 - bR} \tag{10.22}$$

$$E = e^{dR} - 1 \tag{10.23}$$

$$\delta R = 0 = I - E \tag{10.24}$$

[20] Why would this make sense ecologically?

Here we find this empricially by creating a function of R to minimize — we will minimize $(I-E)^2$; squaring the difference gives the quantity the convenient property that the minimum will be approached from either positive or negative values.

```
> deltaR <- function(R) {
+     (exp(I0 - b * R) - (exp(d * R) - 1))^2
+ }
```

We feed this into an optimizer for one-parameter functions, and specify that we know the optimum will be achieved somewhere in the interval between 1 and 50.

```
> optimize(f = deltaR, interval = c(1, 50))[["minimum"]]
```

```
[1] 16.91
```

The output tells us that the minimum was achieved when $R \approx 16.9$.

Now imagine that rising sea level causes island area to shrink. What is this predicted to do? It could

1. reduce the base immigration rate because the island is a smaller target,
2. increase extinction rate because of reduced densities.[21]

Let us represent reduced immigration rate by reducing I_0.

```
> I0 <- log(1/2)
> curve(exp(I0 - b * x), 0, 50, add = TRUE, lty = 2)
```

Next we increase extinction rate by increasing the per species rate.

```
> d <- 0.014
> curve(exp(d * x) - 1, 0, 50, add = TRUE, lty = 2)
```

If we note where the rates cross, we find that the number of predicted species has declined. With these new immigration and extinciton rates the predicted number of species is

```
> optimize(f = deltaR, interval = c(1, 50))[["minimum"]]
```

```
[1] 11.00
```

or 11 species, roughly a 35% decline ($(17 - 11)/17 = 0.35$).

The beauty of this theory is that it focuses our attention on *landscape level processes*, often outside the spatial and temporal limits of our sampling regimes. It specifies that any factor which helps determine the immigration rate or extinction rate, including island area or proximity to a source of propagules, is predicted to alter the equilibrium number of species at any point in time. We should further emphasize that the *identity* of species should change over time, that is, undergo turnover, because new species arrive and old species become extinct. The rate of turnover, however, is likely to be slow, because the species that are most likely to immigrate and least likely to become extinct will be the same species from year to year.

[21] We might also predict an increased extinction rate because of reduced rescue effect (Chapter 4).

10.5.3 Partitioning species–area relations

You may already be wondering if there is a link island biogeography and β-diversity. After all, as we move from island to island, and as we move from small islands to large islands, we typically encounter additional species, and that is what we mean by β-diversity. Sure enough, there are connections [42].

Let us consider the moth data we used above (Fig. 10.14). The total number of species in all of the patches is, as before, γ. The average richness of these patches is $\bar{\alpha}$, and also note that *part of what determines that average is the area of the patch*. That is, when a species is missing from a patch, part of the reason might be that the patch is smaller than it could be. We will therefore partition β into yet one more category: species missing due to patch size, β_{area}. This new quantity is the average difference between $\bar{\alpha}$ and the diversity *predicted* for the largest patch (Fig. 10.20). In general then,

$$\beta = \beta_{area} + \beta_{replace} \tag{10.25}$$

where $\beta_{replace}$ is the average number of species missing that are *not* explained by patch size.

In the context of these data (Fig. 10.20), we now realize that $\beta_1 = \beta_{area} + \beta_{ecoregion}$, so the full partition becomes

$$\gamma = \bar{\alpha}_1 + \beta_{area} + \beta_{ecoregion} + \beta_{geogr.region} \tag{10.26}$$

where $\beta_{replace} = \beta_{ecoregion} + \beta_{geogr.region}$. Note that earlier in the chapter, we did not explore the effect of area. In that case, $\beta_{ecoregion}$ included both the effect of area and the effect of ecoregion; here we have further partitioned this variation into variation due to patch size, as well as variation due to ecoregion. This reduces the amount of unexplained variation among sites within each ecoregion.

Let's calculate those differences now. We will use quantities we calculated above for $\bar{\alpha}_1$, $\bar{\alpha}_2$, γ, and a nonlinear species–area model from above. We can start to create a graph similar to Fig. 10.20.

```
> plot(spp ~ area, data = moths, ylim = c(30, 230),
+     xlab = "Area (ha)", ylab = "No. of Species (R)")
> curve(coef(mod.nonlin)[1] * x^coef(mod.nonlin)[2],
+     0, max(moths[["area"]]), add = TRUE, lty = 2,
+     lwd = 2)
> abline(h = g, lty = 3)
> text(275, g, quote(gamma), adj = c(0.5, 1.5),
+     cex = 1.5)
```

Next we need to find the *predicted richness* for the maximum area. We use our statistical model to find that.

```
> (MaxR <- predict(mod.nonlin, list(area = max(moths[["area"]]))))

[1] 88.62
```

We can now find β_{area}, β_{eco} and β_{geo}.

```
> b.area <- MaxR - a1
> b.eco <- a2 - (b.area + a1)
> b.geo <- g - a2
```

Now we have partitioned γ a little bit more finely with a beastiary of β's, where

- $\bar{\alpha}_1$ is the average site richness.
- β_{area} is the average number of species not observed, due to different patch sizes.
- β_{eco} is the average number of species not observed at a site, is not missing due to patch size, but is in the ecoregion.
- β_{geo} is the average number of species not found in the samples from different ecoregions.

Finally, we add lines to our graph to show the partitions.

```
> abline(h = g, lty = 3)
> abline(h = b.eco + b.area + a1, lty = 3)
> abline(h = b.area + a1, lty = 3)
> abline(h = a1, lty = 3)
```

Now we have further quantified how forest fragment area explains moth species richness. Such understanding of the spatial distribution of biodiversity provides a way to better quantify patterns governed by both dispersal and habitat preference, and allows us to better describe and manage biodiversity in human-dominated landscapes.

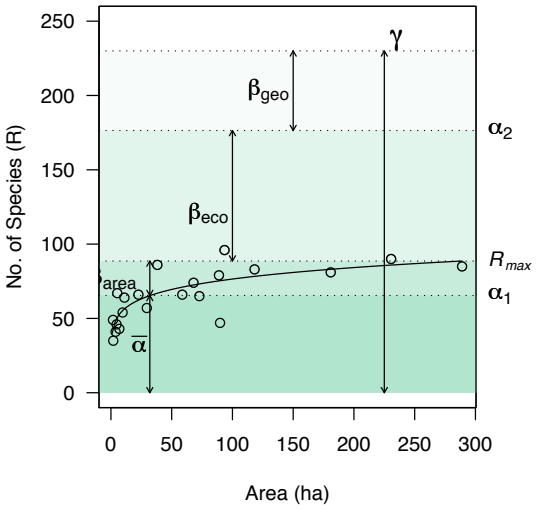


Fig. 10.20: Combining species–area relations with additive diversity partitioning. Forest fragment area explains relatively little of the diversity which accumulates in isolated patches distributed in space. However, it is likely that area associated with the collection of samples (i.e., the distances among fragments) contributes to β_{eco} and β_{geo}.

10.6 Summary

We have examined communities as multivariate entities which we can describe and compare in a variety of ways.

- Composition includes all species (multivariate data), whereas species diversity is a univariate description of the variety of species present.
- There are many ways to quantify species diversity, and they tend to be correlated. The simplest of these is richness (the number of species present), whereas other statistics take species' relative abundances into account.
- Species abundance distributions and rank abundance distributions are analogous to probability distributions, and provide more thorough ways to describe the patterns of abundance and variety in communities. These all illustrate a basic law of community ecology: most species are rare. Null models of community structure and processes make predictions about the shape of these distributions.
- Ecological neutral theory provides a dynamical model, not unlike a null model, which allows quantitative predictions relating demographic, immigration, and speciation rates, species abundance distributions, and patterns of variation in space and time.
- Another law of community ecology is that the number of species increases with sample area and appears to be influenced by immigration and extinction rates.
- We can partition diversity at different spatial scales to understand the structure of communities in landscapes.

Problems

Table 10.3: Hypothetical data for Problem 1.

Site	Sp. A	Sp. B	Sp. C
Site 1	0	1	10
Site 2	5	9	10
Site 3	25	20	10

10.1. How different are the communities in Table 10.3?
(a) Convert all data to relative abundance, where the relative abundance of each site sum to 1.
(b) Calculate the Euclidean and Bray-Curtis (Sørensen) distances between each pair of sites for both relative and absolute abundances.
(c) Calculate richness, Simpson's and Shannon-Wiener diversity for each site.

10.2. Use rarefaction to compare the tree richness in two 1 ha plots from the BCI data in the `vegan` package. Provide code, and a single graph of the expectations

for different numbers of individuals; include in the graph some indication of the uncertainty.

10.3. Select one of the 1 ha BCI plots (from the `vegan` package), and fit three different rank abundance distributions to the data. Compare and contrast their fits.

10.4. Simulate a neutral community of 1000 individuals, selecting the various criteria on yur own. Describe the change through time. Relate the species abundance distributions that you observe through time to the parameters you choose for the simulation.

10.5. Using the `dune` species data (`vegan` package), partition species richness into $\bar{\alpha}, \beta$, and γ richness, where rows are separate sites. Do the same thing using Simpson's diversity.

A

A Brief Introduction to R

R is a language. Use it every day, and you will learn it quickly.

§, the precursor to R, is a quantitative programming environment developed at AT&T Bell Labs in the 1970s. S-Plus is a commercial, "value-added" version and was begun in the 1980s, and R was begun in the 1990s by Robert Gentleman and Ross Ihaka of the Statistics Department of the University of Auckland. Nearly 20 senior statisticians provide the core development group of the R language, including the primary developer of the original § language, John Chambers, of Bell Labs.

R is an official part of the Free Software Foundation's GNU project[1] (http://www.fsf.org/). It is free to all, and licensed to stay that way.

R is a language and environment for dynamical and statistical computing and graphics. R is similar to the S language, different implementation of §. Technically speaking, R is a "dialect" of §. R provides a very wide variety of statistical (linear and nonlinear modelling, classical statistical tests, time-series analysis, classification, clustering, ...) and graphical techniques, and is highly extensible. R has become the lingua franca of academic statistics, and is very useful for a wide variety of computational fields, such as theoretical ecology.

A.1 Strengths of R/S

- Simple and compact syntax of the language. You can learn R quickly, and can accomplish a lot with very little code.
- Extensibility. Anyone can extend the functionality of R by writing code. This may be a simple function for personal use, or a whole new family of statistical procedures in a new package.
- A huge variety of statistical and computing procedures. This derives from the ease with which R/§ can be extended and shared by users around the world.
- Rapid updates.

[1] Pronounced "g-noo" — it is a recursive acronym standing for "GNU's Not Unix."

- Replicability and validation. All data analyses should be well documented, and this only happens reliably when the analyses are performed with scripts or programs, as in R or SAS. Point-and-click procedures cannot be validated by supervisors, reviewers, or auditors.
- Getting help from others is easy. Any scripted language can be quickly and easily shared with someone who can help you. I cannot help someone who says "first I clicked on this, and then I clicked on that ...."
- Repetitive tasks simplified. Writing code allows you to do anything you want a huge number of times. It also allows very simple updates with new data.
- High quality graphics. Well-designed publication-quality plots can be produced with ease, including mathematical symbols and formulae where needed. Great care has been taken over the defaults for the minor design choices in graphics, but the user retains full control.
- R is available as Free Software under the terms of the Free Software Foundation's GNU General Public License in source code form. It compiles and runs out of the box on a wide variety of UNIX platforms and similar systems (including FreeBSD and Linux). It also compiles and runs on Windows 9x/NT/2000 and Mac OS.
- Accessibility. Go now to www.r-project.org. Type "R" into Google. The R Project page is typically the first hit.

There is a tremendous amount of free documentation for R. Rcomes with a selection of manuals under the "Help" menu — explore these first. At the main R project web site, see the "Documentation" on the left sidebar. The FAQ's are very helpful. The "Other" category includes a huge variety of items; search in particular for "Contributed documentation."[2] There you will find long (100+ pages) and short tutorials. You will also find two different "RReference Cards," which are useful lists of commonly used functions.[3]

A.2 The R Graphical User Interface (GUI)

R has a very simple but useful graphical user interface (GUI; Fig. A.1). A few points regarding the GUI:

- "You call this a graphical user interface?" Just kidding — the GUI is *not* designed for point-and-click modeling.
- The R GUI is designed for package management and updates.
- The R GUI is designed for use with scripts.

The R GUI does not provide a "statistics package." R is a language and programming environment. You can download an R package called `Rcmdr` that provides a point-and-click interface for introductory statistics, if you really, really want to. In my experience, students who plan to use statistics in their

[2] I find this when I google "r 'contributed documentation'."
[3] Try googling 'R Reference Card' in quotes.

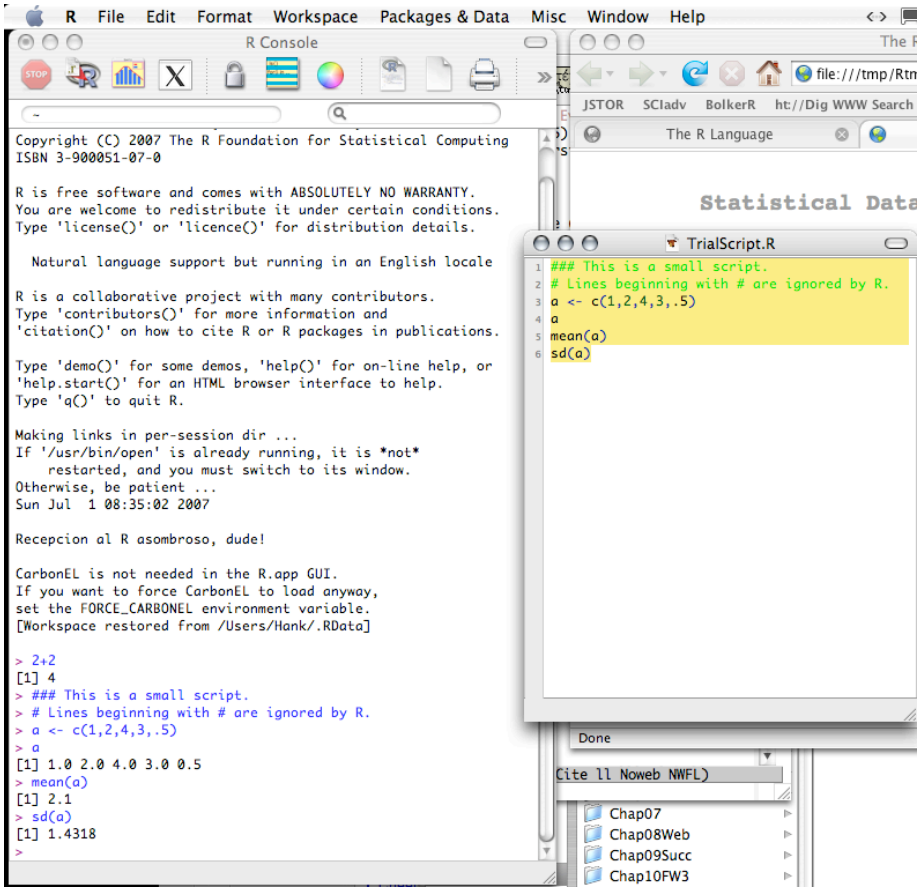


Fig. A.1: The Mac OS X R GUI. Color coded syntax not visible in this figure.

research find it more frustrating to learn this interface than to learn to take advantage of the language.

The R GUI *is* designed for maintenance. With the R GUI you can check for updates, and download any of the hundreds of small packages that extend R in hundreds of ways. (A package is not unlike a "PROC," for SAS users — first you tell call it, then you use it).

The R GUI *is* designed for using scripts. Scripts are text files that contain your analysis; that is, they contain both code to do stuff, and comments about what you are doing and why. These are opened within R and allow you to do work and save the code.

- Scripts are NOT Microsoft Word documents that require Microsoft Word to open them, but rather, simple text files, such as one could open in Notepad, or SimpleText.
- Scripts are a written record of everything you do along the way to achieving your results.

- Scripts are the core of data analysis, and provide many of the benefits of using a command-driven system, whether R, Matlab, or some other program or environment.
- Scripts are interactive. I find that because scripts allow me to do anything and record what I do, they are very interactive. They let me try a great variety of different things very quickly. This will be true for you too, once you begin to master the language.

The R GUI can be used for simple command line use. At the command line, you can add 2+2 to get 4. You could also do `ar(lake.phosphorus)` to perform autoregressive time series analysis on a variable called `lake.phosphorus`, but you would probably want to do that in a script that you can save and edit, to keep track of the analysis that you are doing.

A.3 Where is R?

As an Open Source project, R is distributed across the web. People all around the world continue to develop it, and much of it is stored on large computer servers ("mirrors") across the world. Therefore, when you download R, you download only a portion of it — the language and a few base packages that help everything run. Hundreds of "value-added" packages are available to make particular tasks and groups of tasks easier. We will download one or more of these.

It is useful to have a clear conception of where different parts of R reside (Fig. A.2). Computer servers around the world store identical copies of everything (hence the terms archive and "mirrors"). When you open R, you load into your computer's virtual, temporary RAM more than just the R language — you automatically load several useful packages including "base," "stat," and others. Many more packages exist (about a dozen come with the normal download) and hundreds are available at each mirror. These are easily downloaded through the R GUI.

A.4 Starting at the Very Beginning

To begin with, we will go through a series of steps that will get you up and running using a script file with which to interact with R, and using the proper working directory. Start here.

1. Create a new directory (i.e., a folder) in "Documents" (Mac) or "My Documents" (Windows). *and call it* "Rwork." For now, calling it the same thing as everyone else will just simplify your life. If you put "Rwork" elsewhere, adjust the relevant code as you go. For now, keep all of your script files and output files into that directory.
2. Open the R GUI in a manner that is appropriate for your operating system and setup (e.g., double-click the desktop icon).

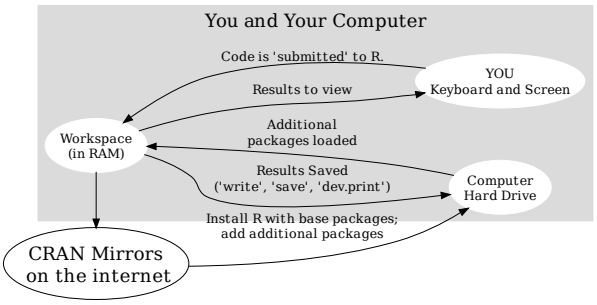


Fig. A.2: A conceptual representation of where R exists. "CRAN" stands for "Comprehensive R Archive Network." "RAM" (random access memory) is your computer's active brain; R keeps some stuff floating in this active memory and this "stuff" is the *workspace*.

3. Set the *working directory.* You can do this via Misc directory in Mac OS X or in the File menu in Windows using "Change dir...." Set the working directory to "Rwork." (If you have not already made an `Rwork` directory, do so now — put it in "Documents" or "My Documents.")
4. Open a new R script ("New Document") by using the File menu in the R GUI. On the first line, type `# My first script` with the pound sign. On the next line, type `setwd('~/Documents/Rwork')` if you are on a Mac, or `setwd('C:/Documents and Settings/Users/Jane/My Documents/Rwork')` on Windows, assuming you are named "Jane;" if not, use the appropriate pathname. Save this file in "Rwork;" save it as "RIntro.R." Windows may hide the fact that it is saving it as a ".txt" file. I will assume you can prevent this.

You should submit code to R directly from the script. *Use the script to store your ideas as comments (beginning with #) and your code, and submit code directly from the script file within R (see below for how to do that).* You do *not* need to cut-and-paste. There are slight differences between operating systems in how to submit code from the script file to the command line.

- In Microsoft Windows, place the cursor on a line in the script file *or* highlight a section of code, and then hit `Ctrl-R` to submit the code.
- In Apple Mac OS X, highlight a section of code and then hit `Command-return` to submit the code (see the Edit menu).

From this point on, enter all of the code (indicated in typewriter font, and beginning with ">") in your script file, save the file with Command-S (Mac) or Ctrl-S (Windows), and then submit the code as described above. Where a line begins with "+," ignore the plus sign, because this is just used by R to indicate continued lines of code. You may enter a single line of code on more than one line. R will simply continue as if you wrote it all on one line.

You can start the help interface with this command.

```
> help.start()
```

This starts the HTML interface to on-line help (using a web browser available at your machine). You can use help within the R GUI, or with this HTML help system.

Find out where you are using `getwd()` (*get* the *w*orking *d*irectory). Use a comment (beginning with #) to remind yourself what this does.

```
> # Here I Get my Working Directory; that is,
> # I find out which folder R is currently operating from.
> getwd()
```

The precise output will depend on your computer.

You can also *set* the *w*orking *d*irectory using `setwd()`; if you created a directory called `Rwork` as specified above, one of the following these should work, depending on your operating system. If these both fail, keep trying, or use the menu in the R GUI to set the working directory.

```
> setwd("~/Documents/Rwork")
```

or

```
> setwd("C:/Documents and Settings/Users/Jane/My Documents/Rwork")
```

On the Mac, a UNIX environment, the tilde-slash (~ /) represents your home directory.

I urge you to use `setwd` *at the beginning of each script file you write so that this script always sets you up to work in a particular, specified directory.* As you write your script file, remember,

- Text that begins with "#" will be ignored by R.
- Text that does not make sense to R will cause R to return an error message, but will not otherwise mess things up.

Remember that a strength of R is that you can provide, to yourself and others, comments that explain every step and every object. To add a comment, simply type one or more "#," followed by your comment.

Finally, have fun, be amused. Now you are ready for programming in R.

R is a language. Use it every day, and you will learn it quickly.

B

Programming in R

This material assumes you have completed Appendix A, the overview of R. Do the rest of this Appendix in a script. Make comments of your own throughout your script.

Open and save a new file (or *script*) in R. Think of your script as a pad of paper, on which you program and *also on which you write notes to yourself.* See Appendix A for instructions.

You will want to take copious notes about what you do. Make these notes in the script file, using the pound sign #. Here is an example:

```
> # This will calculate the mean of 10 random standard normal variables.
> mean( rnorm( 10 ) )

[1] 0.1053
```

You submit this (as described above) from the script directly to the Console with (select) Command-r on a Mac, or Ctrl-r on Windows.

B.1 Help

You cannot possibly use R without using its help facilities. R comes with a lot of documentation (see the relevant menu items), and people are writing more all the time (this document is an example!).

After working through this tutorial, you could go through the document "An Introduction to R" that comes with R. You can also browse "Keywords by Topic" which is found under "Search Engine & Keywords" in the Help menu.

To access help for a specific function, try

```
> `?`(mean)

Help for 'mean' is shown in browser /usr/bin/open ...
Use
        help("mean", htmlhelp = FALSE)
or
        options(htmlhelp = FALSE)
to revert.
```

or

```
> help(mean)
```

```
Help for 'mean' is shown in browser /usr/bin/open ...
Use
        help("mean", htmlhelp = FALSE)
or
        options(htmlhelp = FALSE)
to revert.
```

The help pages provide a very regular structure. There is a *name*, a brief *description*, its *usage*, its *arguments*, *details* about particular aspects of its use, the *value* (what you get when you use the function), *references*, *links* to other functions, and last, *examples*.

If you don't know the exact name of the R function you want help with, you can try

```
> help.search("mean")
> apropos("mean")
```

These will provide lists of places you can look for functions related to this keyword.

Last, a great deal of R resides in *packages* online (on duplicate servers around the world). If you are on line, help for functions that you have not yet downloaded can be retrieved with

```
> RSiteSearch("violin")
> RSiteSearch("violin", restrict = c("functions"))
```

To learn more about help functions, combine them!

```
> help(RSiteSearch)
```

B.2 Assignment

In general in R, we perform an action, and take the results of that action and *assign* the results to a new object, thereby creating a new object. Here I add two numbers and assign the result to an new object I call a.

```
> a <- 2 + 3
> a
```

```
[1] 5
```

Note that I use an arrow to make the assignment — I make the arrow with a less-than sign, <, and a dash. Note also that to reveal the contents of the object, I can type the name of the object.

I can then use the new object to perform another action, and assign

```
> b <- a + a
```

I can perform two actions on one line by separating them with a semicolon.

```
> a + a; a + b

[1] 10
[2] 15
```

Sometimes the semicolon is referred to as an "end of line" operator.

B.3 Data Structures

We refer to a single number as a scalar; a scalar is a single real number. Most objects in R are more complex. Here we describe some of these other objects: vectors, matrices, data frames, lists, and functions.

B.3.1 Vectors

Perhaps the fundamental unit of R is the *vector*, and most operations in R are performed on vectors. A vector may be just a column of scalars, for instance; this would be a column vector.

Here we create a vector called `Y`.

To enter data directly into R, we will use `c()` and create an R object, in particular a vector. A vector is, in this case, simply a group of numbers arranged in a row or column. Type into your script

```
> Y <- c(8.3, 8.6, 10.7, 10.8, 11, 11, 11.1, 11.2,
+     11.3, 11.4)
```

where the arrow is a less-than sign, `<`, and a dash, `-`. Similarly, you could use

```
> Y = c(8.3, 8.6, 10.7, 10.8, 11, 11, 11.1, 11.2, 11.3,
+     11.4)
```

These are equivalent.

R operates (does stuff) to *objects*. Those objects may be *vectors*, *matrices*, *lists*, or some other class of object.

Sequences

I frequently want to create ordered sequences of numbers. R has a shortcut for sequences of integers, and a slightly longer method that is completely flexible. First, integers:

```
> 1:4

[1] 1 2 3 4

> 4:1

[1] 4 3 2 1

> -1:3
```

```
[1] -1  0  1  2  3
```

```
> -(1:3)
```

```
[1] -1 -2 -3
```

Now more complex stuff, specifying either the units of the sequence, or the total length of the sequence.

```
> seq(from = 1, to = 3, by = 0.2)
```

```
 [1] 1.0 1.2 1.4 1.6 1.8 2.0 2.2 2.4 2.6 2.8 3.0
```

```
> seq(1, 3, by = 0.2)
```

```
 [1] 1.0 1.2 1.4 1.6 1.8 2.0 2.2 2.4 2.6 2.8 3.0
```

```
> seq(1, 3, length = 7)
```

```
[1] 1.000 1.333 1.667 2.000 2.333 2.667 3.000
```

I can also fill in with repetitive sequences. Compare carefully these examples.

```
> rep(1, 3)
```

```
[1] 1 1 1
```

```
> rep(1:3, 2)
```

```
[1] 1 2 3 1 2 3
```

```
> rep(1:3, each = 2)
```

```
[1] 1 1 2 2 3 3
```

B.3.2 Getting information about vectors

Here we can ask R to tell us about `Y`, getting the length (the number of elements), the mean, the maximum, and a six number summary.

```
> sum(Y)
```

```
[1] 105.4
```

```
> mean(Y)
```

```
[1] 10.54
```

```
> max(Y)
```

```
[1] 11.4
```

```
> length(Y)
```

```
[1] 10
```

```
> summary(Y)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
    8.3    10.7    11.0    10.5    11.2    11.4
```

A vector could be character, or logical as well, for instance

```
> Names <- c("Sarah", "Yunluan")
> Names

[1] "Sarah"   "Yunluan"

> b <- c(TRUE, FALSE)
> b

[1]  TRUE FALSE
```

Vectors can also be dates, complex numbers, real numbers, integers, or factors. For factors, such as experimental treatments, see section B.3.5. We can also ask R what classes of data these belong to.

```
> class(Y)

[1] "numeric"

> class(b)

[1] "logical"
```

Here we test whether each element of a vector is greater than a particular value or greater than its mean. *When we test an object, we get a logical vector back that tells us, for each element, whether the condition was true or false.*

```
> Y > 10

 [1] FALSE FALSE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE

> Y > mean(Y)

 [1] FALSE FALSE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
```

We test using >, <, >=, <=, ==, ! = and other conditions. Here we test whether a vector is equal to a number.

```
> Y == 11

 [1] FALSE FALSE FALSE FALSE  TRUE  TRUE FALSE FALSE FALSE FALSE
```

A test of "not equal to"

```
> Y != 11

 [1]  TRUE  TRUE  TRUE  TRUE FALSE FALSE  TRUE  TRUE  TRUE  TRUE
```

This result turns out to be quite useful, including when we want to *extract* subsets of data.

Algebra with vectors

In R, we can add, subtract, multiply and divide vectors. When we do this, we are really *operating on the elements in the vectors*. Here we add vectors `a` and `b`.

```
> a <- 1:3
> b <- 4:6
> a + b

[1] 5 7 9
```

Similarly, when we multiply or divide, we also operate on each pair of elements in the pair of vectors.

```
> a * b

[1]  4 10 18

> a/b

[1] 0.25 0.40 0.50
```

We can also use scalars to operate on vectors.

```
> a + 1

[1] 2 3 4

> a * 2

[1] 2 4 6

> 1/a

[1] 1.0000 0.5000 0.3333
```

What R is doing is *recycling* the scalar (the 1 or 2) as many times as it needs to in order to match the length of the vector. Note that if we try to multiply vectors of unequal length, R performs the operation but may or may not give a warning. Above, we got no warningmessage. However, if we multiply a vector of length 3 by a vector of length 2, R returns a warning.

```
> a * 1:2

[1] 1 4 3
```

R *recycles the shorter vector* just enough to match the length of the longer vector. The above is the same as

```
> a * c(1, 2, 1)

[1] 1 4 3
```

On the other hand, if we multiply vectors of length 4 and 2, we get no error, because four is a multple of 2.

```
> 1:4 * 1:2

[1] 1 4 3 8
```

Recycling makes the above the same as the following.

```
> 1:4 * c(1, 2, 1, 2)

[1] 1 4 3 8
```

B.3.3 Extraction and missing values

We can extract or subset elements of the vector.

I extract subsets of data in two basic ways, by

- identifying which rows (or columns) I want (i.e. the first row), or
- providing a *logical* vector (of TRUE's and FALSE's) of the same length as the vector I am subsetting.

Here I use the first method, using a single integer, and a sequence of integers.

```
> Y[1]

[1] 8.3

> Y[1:3]

[1]  8.3  8.6 10.7
```

Now I want to extract all girths greater than the average girth. Although I don't have to, I remind myself what the logical vector looks like, and then I use it.

```
> Y > mean(Y)

 [1] FALSE FALSE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE

> Y[Y > mean(Y)]

[1] 10.7 10.8 11.0 11.0 11.1 11.2 11.3 11.4
```

Note that I get back all the values of the vector where the condition was TRUE.

In R, *missing data* are of the type "NA." This means "not available," and R takes this appellation seriously. thus is you try to calculate the mean of a vector with missing data, R resists doing it, because if there are numbers missing from the set, how could it possibly calculate a mean? If you ask it to do something with missing data, the answer will be missing too.

Given that R treats missing data as missing data (and not something to be casually tossed aside), there are special methods to deal with such data. For instance, we can test which elements are missing with a special function, `is.na`.

```
> a <- c(5, 3, 6, NA)
> a

[1]  5  3  6 NA

> is.na(a)
```

```
[1] FALSE FALSE FALSE  TRUE
```

```
> !is.na(a)
```

```
[1]  TRUE  TRUE  TRUE FALSE
```

```
> a[!is.na(a)]
```

```
[1] 5 3 6
```

```
> na.exclude(a)
```

```
[1] 5 3 6
attr(,"na.action")
[1] 4
attr(,"class")
[1] "exclude"
```

Some functions allow you to remove missing elements on the fly. Here we let a function fail with missing data, and then provide three different ways to get the same thing.

```
> mean(a)
```

```
[1] NA
```

```
> mean(a, na.rm = TRUE)
```

```
[1] 4.667
```

```
> d <- na.exclude(a)
> mean(d)
```

```
[1] 4.667
```

Note that R takes missing data seriously. If the fourth element of the set really is missing, I cannot calculate a mean because I don't know what the vector is.

B.3.4 Matrices

A matrix is a two dimensional set of elements, for which *all elements are of the same type*. Here is a character matrix.

```
> matrix(letters[1:4], ncol = 2)
```

```
     [,1] [,2]
[1,] "a"  "c"
[2,] "b"  "d"
```

Here we make a numeric matrix.

```
> M <- matrix(1:4, nrow = 2)
> M
```

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

Note that the matrix is filled in by columns, or *column major order*. We could also do it by rows.

```
> M2 <- matrix(1:4, nrow = 2, byrow = TRUE)
> M2
     [,1] [,2]
[1,]    1    2
[2,]    3    4
```

Here is a matrix with 1s on the diagonal.

```
> I <- diag(1, nrow = 2)
> I
     [,1] [,2]
[1,]    1    0
[2,]    0    1
```

The identity matrix plays a special role in matrix algebra; in many ways it is equivalent to the scalar 1. For instance, the inverse of a matrix, $\mathbf{M}$, is $\mathbf{M}^{-1}$, which is the matrix which satisfies the equality $\mathbf{M}\mathbf{M}^{-1} = \mathbf{I}$, where $\mathbf{I}$ is the identity matrix. We solve for the inverse using a few different methods, including

```
> Minv <- solve(M)
> M %*% Minv
     [,1] [,2]
[1,]    1    0
[2,]    0    1
```

QR decomposition is available (e.g., `qr.solve()`).

Note that R recycles the "1" until the specified number of rows and columns are filled. If we do not specify the number of rows and columns, R fills in the matrix with what you give it (as it did above).

Extraction in matrices

I extract elements of matrices in the same fashion as vectors, but specify both rows and columns.

```
> M[1, 2]
[1] 3
> M[1, 1:2]
[1] 1 3
```

If I leave either rows or columns blank, R returns all rows (or columns).

```
> M[, 2]
[1] 3 4
> M[, ]
     [,1] [,2]
[1,]    1    3
[2,]    2    4
```

Simple matrix algebra

Basic matrix algebra is similar to algebra with scalars, but with a few very important differences. Let us define another matrix.

```
> N <- matrix(0:3, nrow = 2)
> N
```

```
     [,1] [,2]
[1,]    0    2
[2,]    1    3
```

To perform scalar, or element-wsie operations, we have

$$\mathbf{A} = \begin{pmatrix} a & b \\ c & d \end{pmatrix};\ \mathbf{B} = \begin{pmatrix} m & o \\ n & p \end{pmatrix} \tag{B.1}$$

$$\mathbf{AB} = \begin{pmatrix} am & bo \\ cn & dp \end{pmatrix} \tag{B.2}$$

The element-wise operation on these two is the default in R,

```
> M * N
```

```
     [,1] [,2]
[1,]    0    6
[2,]    2   12
```

where the element in row 1, column 1 in `M` is multiplied by the element in the same position in `N`.

To perform *matrix mulitplication*, recall from Chap. 2 that,

$$\mathbf{A} = \begin{pmatrix} a & b \\ c & d \end{pmatrix};\ \mathbf{B} = \begin{pmatrix} m & o \\ n & p \end{pmatrix} \tag{B.3}$$

$$\mathbf{AB} = \begin{pmatrix} (am + bn) & (ao + bp) \\ (cm + dn) & (co + dp) \end{pmatrix} \tag{B.4}$$

To perform *matrix mulitplication* in R, we use **%*%**,

```
> M %*% N
```

```
     [,1] [,2]
[1,]    3   11
[2,]    4   16
```

Refer to Chapter 2 (or "matrix algebra" at Wikipedia) for why this is so.

Note that matrix multiplication is not commutative, that is, $\mathbf{NM} \neq \mathbf{MN}$. Compare the previous result to

```
> N %*% M
```

```
     [,1] [,2]
[1,]    4    8
[2,]    7   15
```

Note that a *vector* in R is not defined *a priori* as a column matrix or a row matrix. Rather, it is used as either depending on the circumstances. Thus, we can either left multiply or right multiply a vector of length 2 and `M`.

```
> 1:2 %*% M

     [,1] [,2]
[1,]    5   11

> M %*% 1:2

     [,1]
[1,]    7
[2,]   10
```

If you want to be very, very clear that your vector is really a matrix with one column (a column vector), you can make it thus.

```
> V <- matrix(1:2, ncol = 1)
```

Now when you multiply `M` by `V`, you will get the expected sucesses and failure, according to the rules of matrix algebra.

```
> M %*% V

     [,1]
[1,]    7
[2,]   10

> try(V %*% M)
```

R has formal rules about how it converts vectors to matrices on-the-fly, but it is good to be clear on your own.

Other matrix operations are available. Whenever we add or subtract matrices together, or add a matrix and a scalar, it is always element-wise.

```
> M + N

     [,1] [,2]
[1,]    1    5
[2,]    3    7

> M + 2

     [,1] [,2]
[1,]    3    5
[2,]    4    6
```

The transpose of a matrix is the matrix we get when we substitute rows for columns, and columns for rows. To transpose matrices, we use `t()`.

```
> t(M)

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

More advanced matrix operations are available as well, for singular value decomposition (`svd`), eigenanalysis (`eigen`), finding determinants (`det`), QR decomposition (`qr`), Choleski factorization (`chol`), and related functions. The `Matrix` package was designed to handle with aplomb large sparse matrices.

B.3.5 Data frames

Data frames are two dimensional, a little like spreadsheets and matrices. All columns having exactly the same number of rows. Unlike matrices, each column can be a different data type (e.g., numeric, integer, charactor, complex, imaginary). For instance, the columns of a data frame could contain the names of species, the experimental treatment used, and the dimensions of species traits, as character, factor, and numeric variables, respectively.

```
> dat <- data.frame(species = c("S.altissima", "S.rugosa",
+     "E.graminifolia", "A. pilosus"), treatment = factor(c("Control",
+     "Water", "Control", "Water")), height = c(1.1,
+     0.8, 0.9, 1), width = c(1, 1.7, 0.6, 0.2))
> dat

         species treatment height width
1    S.altissima   Control    1.1   1.0
2       S.rugosa     Water    0.8   1.7
3 E.graminifolia   Control    0.9   0.6
4     A. pilosus     Water    1.0   0.2
```

We can extract data from data frames just the way we can with matrices.

```
> dat[2, ]

   species treatment height width
2 S.rugosa     Water    0.8   1.7

> dat[3, 4]

[1] 0.6
```

We can test elements in data frames, as here where I test whether each element column 2 is "Water." I then use that to extract rows of data that are associated with this criterion.

```
> dat[, 2] == "Water"

[1] FALSE  TRUE FALSE  TRUE

> dat[dat[, 2] == "Water", ]

     species treatment height width
2   S.rugosa     Water    0.8   1.7
4 A. pilosus     Water    1.0   0.2
```

I could also use the subset function

```
> subset(dat, treatment == "Water")
```

```
     species treatment height width
2   S.rugosa     Water    0.8   1.7
4 A. pilosus     Water    1.0   0.2
```

There are advantages to using data frames which will become apparent.

Factors

Factors are a class of data; as such they could belong above with our discussion of character and logical and numeric vectors. I tend, however, to use them in data frames almost exclusively, because I have a data set that includes a bunch of response variables, and *the factors imposed by my experiment*.

When defining a factor, R by default orders the factor levels in alphabetic order — we can reorder them as we like. Here I demonstrate each piece of code and then use the pieces to make a factor in one line of code.

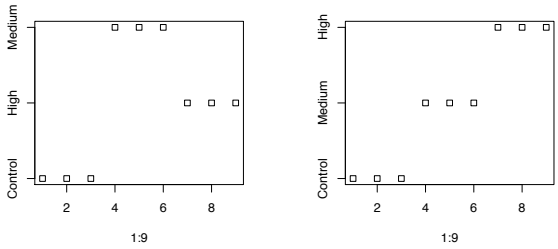
```
> c("Control", "Medium", "High")

[1] "Control" "Medium"  "High"

> rep(c("Control", "Medium", "High"), each = 3)

[1] "Control" "Control" "Control" "Medium"  "Medium"  "Medium"
[7] "High"    "High"    "High"

> Treatment <- factor(rep(c("Control", "Medium", "High"),
+     each = 3))
> Treatment

[1] Control Control Control Medium  Medium  Medium  High
[8] High    High
Levels: Control High Medium
```

Note that R orders the factor alphabetically. This may be relevant if we do something with the factor, such as when we plot it (Fig. B.1a).

```
> levels(Treatment)

[1] "Control" "High"    "Medium"

> stripchart(1:9 ~ Treatment)
```

Now we can re-specify the factor, telling R the order of the levels we want, taking care to remember that R can tell the difference between upper and lower case (Fig. B.1b). See also the function `relevel`.

```
> Treatment <- factor(rep(c("Control", "Medium", "High"),
+     each = 3), levels = c("Control", "Medium", "High"))
> levels(Treatment)

[1] "Control" "Medium"  "High"

> stripchart(1:9 ~ Treatment)
```



(a) Before (b) After

Fig. B.1: Graphics before and after the factor was releveled to place the factor levels in a logical order.

B.3.6 Lists

An amazing data structure that R boasts is the *list*. A *list* is simply a collection of other objects kept together in a hierarchical structure. Each component of the list can be a complete different class of object. Let's build one.

```
> my.list <- list(My.Y = Y, b = b, Names, Weed.data = dat,
+     My.matrix = M2, my.no = 4)
> my.list

$My.Y
 [1]  8.3  8.6 10.7 10.8 11.0 11.0 11.1 11.2 11.3 11.4

$b
[1] 4 5 6

[[3]]
[1] "Sarah"   "Yunluan"

$Weed.data
         species treatment height width
1    S.altissima   Control    1.1   1.0
2       S.rugosa     Water    0.8   1.7
3 E.graminifolia   Control    0.9   0.6
4     A. pilosus     Water    1.0   0.2

$My.matrix
     [,1] [,2]
[1,]    1    2
[2,]    3    4

$my.no
[1] 4
```

We see that this list is a set of objects: a numeric vector, a logical vector, a character vector, a data frame, a matrix, and a scalar (a number). Lists can be nested within other lists.

Note that if we do not specify a name for a component, we can still extract it using the number of the component.

I extract list components in several ways, including by name, and by number (see `?'['` for more information).

```
> my.list[["b"]]

[1] 4 5 6

> my.list[[2]]

[1] 4 5 6
```

If I use a name, there are a few ways, including

```
> my.list[["b"]]

[1] 4 5 6

> my.list$b

[1] 4 5 6
```

If by number, that are two ways, with one or two brackets. In addition to two brackets, as above, we can use one bracket. This allows for extraction of more than one component of the list.

```
> my.list[1:2]

$My.Y
 [1]  8.3  8.6 10.7 10.8 11.0 11.0 11.1 11.2 11.3 11.4

$b
[1] 4 5 6
```

Note that I can extract a subset of one component.

```
> my.list[["b"]][1]

[1] 4
```

If one way of extraction is working for you, experiment with others.

B.3.7 Data frames are also lists

You can also think of a data frame as a list of columns of identical length. I like to extract columns the same way — by name.

```
> mean(dat$height)

[1] 0.95
```

B.4 Functions

A function is a command that does something. You have already been using functions, throughout this document. Let's examine functions more closely.

Among other things, a function has a name, arguments, and values. For instance,

```
> help(mean)
```

This will open the help page (again), showing us the *arguments*. The first argument `x` is the object for which a mean will be calculated. The second argument is `trim=0`. If we read about this argument, we find that it will "trim" a specified fraction of the most extreme observations of `x`. *The fact that the argument* ***trim*** *is already set equal to zero means that is the default.* If you do not use `trim`, then the function will use `trim=0`. Thus, these two are equivalent.

```
> mean(1:4)

[1] 2.5

> mean(1:4, trim = 0)

[1] 2.5
```

R is an "object-oriented" language. A consequence of this is that the same function name will perform different actions, depending on the *class* of the object.[1]

```
> class(1:10)

[1] "integer"

> class(warpbreaks)

[1] "data.frame"

> summary(1:10)

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   1.00    3.25    5.50    5.50    7.75   10.00

> summary(warpbreaks)

     breaks     wool   tension
 Min.   :10.0   A:27   L:18
 1st Qu.:18.2   B:27   M:18
 Median :26.0          H:18
 Mean   :28.1
 3rd Qu.:34.0
 Max.   :70.0
```

In the `warpbreaks` data frame, summary provides the six number summary for each numeric or integer column, but provides "tables" of the factors, that is, it counts the occurrences of each level of a factor and sorts the levels. When we use summary on a linear model, we get output of the regression,

[1] R has hundreds of built-in data sets for demonstrating things. We use one here called 'warpbreaks.' You can find some others by typing `data()`.

```
> summary(lm(breaks ~ wool, data = warpbreaks))

Call:
lm(formula = breaks ~ wool, data = warpbreaks)

Residuals:
   Min     1Q Median     3Q    Max
-21.04  -9.26  -3.65   4.71  38.96

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)    31.04       2.50   12.41   <2e-16
woolB          -5.78       3.54   -1.63     0.11

Residual standard error: 13 on 52 degrees of freedom
Multiple R-squared: 0.0488,        Adjusted R-squared: 0.0305
F-statistic: 2.67 on 1 and 52 DF,  p-value: 0.108
```

B.4.1 Writing your own functions

One very cool thing in R is that you can write your own functions. Indeed it is the extensibility of R that makes it the home of cutting edge working, because edge cutters (i.e., leading scientists) can write code that we all can use. People actually write entire *packages*, which are integrated collections of functions, and R has been extended with hundreds of such packages available for download at all the R mirrors.

Let's make our own function to calculate a mean. Let's further pretend you work for an unethical boss who wants you to show that average sales are higher than they really are. Therefore your function should provide a mean plus 5%.

```
> MyBogusMean <- function(x, cheat = 0.05) {
+     SumOfX <- sum(x)
+     n <- length(x)
+     trueMean <- SumOfX/n
+     (1 + cheat) * trueMean
+ }
> RealSales <- c(100, 200, 300)
> MyBogusMean(RealSales)

[1] 210
```

Thus a function can take any input, do stuff, including produce graphics, or interact with the operating system, or manipulated numbers. You decide on the arguments of the function, in this case, `x` and `cheat`. Note that we supplied a number for `cheat`; this results in the `cheat` argument having a *default* value, and we do not have to supply it. If an argument does not have a default, we have to supply it. If there is a default value, we can change it. Now try these.

```
> MyBogusMean(RealSales, cheat = 0.1)

[1] 220
```

```
> MyBogusMean(RealSales, cheat = 0)

[1] 200
```

B.5 Sorting

We often like to sort our numbers and our data sets; a single vector is easy. To do something else is only a little more difficult.

```
> e <- c(5, 4, 2, 1, 3)
> e

[1] 5 4 2 1 3

> sort(e)

[1] 1 2 3 4 5

> sort(e, decreasing = TRUE)

[1] 5 4 3 2 1
```

If we want to sort all the rows of a data frame, keeping records (rows) intact, we can use `order`. This function is a little tricky, so we explore its use in a vector.

```
> e

[1] 5 4 2 1 3

> order(e)

[1] 4 3 5 2 1

> e[order(e)]

[1] 1 2 3 4 5
```

Here `order` generates an *index* to properly *order* something. Above, this index is used to tell R to select the 4th element of `e` first — `order` puts the number '4' into the first spot, indicating that R should put the 4th element of `e` first. Next, it places '3' in the second spot because the 3rd element of `e` belongs in the 2nd spot of an ordered vector, and so on.

We can use `order` to sort the rows of a data frame. Here I order the rows of the data frame according to increasing order of plant heights.

```
> dat

         species treatment height width
1    S.altissima   Control    1.1   1.0
2       S.rugosa     Water    0.8   1.7
3 E.graminifolia   Control    0.9   0.6
4     A. pilosus     Water    1.0   0.2

> order.nos <- order(dat$height)
> order.nos
```

```
[1] 2 3 4 1
```

This tells us that to order the rows, we have to use the 2nd row of the original data frame as the first row in the ordered data frame, the 3rd row as the new second row, etc. Now we use this index to select the rows of the original data frame in the correct order to sort the whole data frame.

```
> dat[order.nos, ]
```

```
         species treatment height width
2       S.rugosa     Water    0.8   1.7
3 E.graminifolia   Control    0.9   0.6
4     A. pilosus     Water    1.0   0.2
1    S.altissima   Control    1.1   1.0
```

We can reverse this too, of course.

```
> dat[rev(order.nos), ]
```

```
         species treatment height width
1    S.altissima   Control    1.1   1.0
4     A. pilosus     Water    1.0   0.2
3 E.graminifolia   Control    0.9   0.6
2       S.rugosa     Water    0.8   1.7
```

B.6 Iterated Actions: the apply Family and Loops

We often want to perform an action again and again and again..., perhaps thousands or millions of times. In some cases, each action is independent — we just want to do it a lot. In these cases, we have a choice of methods. Other times, each action depends on the previous action. In this case, I always use for-loops.[2] Here I discuss first methods that work only for independent actions.

B.6.1 Iterations of independent actions

Imagine that we have a matrix or data frame and we want to do the same thing to each column (or row). For this we use `apply`, to "apply" a function to each column (or row). We tell `apply` what data we want to use, we tell it the "margin" we want to focus on, and then we tell it the function. The *margin* is the *side* of the matrix. We describe matrices by their number of rows, then columns, as in "a 2 by 5 matrix," so rows constitute the first margin, and columns constitute the second margin. Here we create a 2×5 matrix, and take the mean of rows, for the first margin. Then we sum the columns for the second margin.

```
> m <- matrix(1:10, nrow = 2)
> m
```

[2] There are other methods we could use. These are discussed by others, under various topics, including "flow control." We use ODE solvers for continuous ordinary differential equations.

```
     [,1] [,2] [,3] [,4] [,5]
[1,]    1    3    5    7    9
[2,]    2    4    6    8   10

> apply(m, MARGIN = 1, mean)

[1] 5 6

> apply(m, MARGIN = 2, sum)

[1]  3  7 11 15 19
```

See `?rowMeans` for simple, and even faster, operations.

Similarly, `lapply` will "apply" a function to each element of a list, or each column of a data frame, and always returns a list. `sapply` does something similar, but will simplify the result, to a less complex data structure if possible.

Here we do an independent operation 10 times using `sapply`, defining a function on-the-fly to calculate the mean of a random draw of five observations from the standard normal distribution.

```
> sapply(1:10, function(i) mean(rnorm(5)))

 [1] -0.5612 -0.4815 -0.4646  0.7636  0.1416 -0.5003 -0.1171
 [8]  0.2647  0.6404 -0.1563
```

B.6.2 Dependent iterations

Often the repeated actions depend on previous outcomes, as with population growth. Here we provide a couple of examples where we accomplish this with *for loops*.

One thing to keep in mind for *for loops* in R: the computation of this is fastest if we first make a holder for the output. Here I simulate a random walk, where, for instance, we start with 25 individuals at time = 0, and increase or decrease by some amount that is drawn randomly from a normal distribution, with a mean of zero and a standard deviation 2. We will round the "amount" to the nearest integer (the zero-th decimal place). Your output will differ because it is a random process.

```
> gens <- 10
> output <- numeric(gens + 1)
> output[1] <- 25
> for (t in 1:gens) output[t + 1] <- output[t] + round(rnorm(n = 1,
+     mean = 0, sd = 2), 0)
> output

 [1] 25 29 25 26 28 29 30 32 33 29 30
```

B.7 Rearranging and Aggregating Data Frames

B.7.1 Rearranging or reshaping data

We often need to rearrange our data. A common example in ecology is to collect repeated measurements of an experimental unit and enter the data into multiple columns of a spreadsheet, creating a *wide* format. R prefers to analyze data in a single column, in a *long* format. Here we use `reshape` to rearrange this.

These data are carbon dioxide uptake in 12 individual plants. They are currently structured as longitudinal data; here we rearrange them in the wide format, as if we record uptake seven sequential observations on each plant in different columns. See `?reshape` for details. Here `v.names` refers to the column name of the response variable, `idvar` refers to the column name for the variable that identifies an individual on which we have repeated measurements, and `timevar` refers to the column name which identifies different observations of *the same individual plant*.

```
> summary(CO2)
```

```
    Plant             Type         Treatment       conc
 Qn1    : 7   Quebec     :42   nonchilled:42   Min.   :  95
 Qn2    : 7   Mississippi:42   chilled   :42   1st Qu.: 175
 Qn3    : 7                                   Median : 350
 Qc1    : 7                                   Mean   : 435
 Qc3    : 7                                   3rd Qu.: 675
 Qc2    : 7                                   Max.   :1000
 (Other):42
    uptake
 Min.   : 7.7
 1st Qu.:17.9
 Median :28.3
 Mean   :27.2
 3rd Qu.:37.1
 Max.   :45.5
```

```
> CO2.wide <- reshape(CO2, v.names = "uptake", idvar = "Plant",
+     timevar = "conc", direction = "wide")
> names(CO2.wide)
```

```
 [1] "Plant"       "Type"        "Treatment"   "uptake.95"
 [5] "uptake.175"  "uptake.250"  "uptake.350"  "uptake.500"
 [9] "uptake.675"  "uptake.1000"
```

This is often how we might record data, with an experimental unit (individual, or plot) occupying a single row. If we import the data in this format, we would typically like to reorganize it in the long format, because most analyses we want to do may require this. Here, `v.names` and `timevar` are the names we want to use for some new columns, for the response variable and the identifier of the repeated measurement (typically the latter may be a time interval, but here it is a CO_2 concentration). `times` supplies the identifier for each repeated observation.

```
> CO2.long <- reshape(CO2.wide, v.names = "Uptake",
+     varying = list(4:10), timevar = "Concentration",
+     times = c(95, 175, 250, 350, 500, 675, 1000))
> head(CO2.long)
```

```
     Plant   Type  Treatment Concentration Uptake id
1.95   Qn1 Quebec nonchilled            95   16.0  1
2.95   Qn2 Quebec nonchilled            95   13.6  2
3.95   Qn3 Quebec nonchilled            95   16.2  3
4.95   Qc1 Quebec    chilled            95   14.2  4
5.95   Qc2 Quebec    chilled            95    9.3  5
6.95   Qc3 Quebec    chilled            95   15.1  6
```

If we wanted to, we could use `order()` to re-sort the data frame, for instance to match the original.

```
> CO2.long2 <- with(CO2.long, CO2.long[order(Plant,
+     Concentration), ])
> head(CO2.long2)
```

```
      Plant   Type  Treatment Concentration Uptake id
1.95    Qn1 Quebec nonchilled            95   16.0  1
1.175   Qn1 Quebec nonchilled           175   30.4  1
1.250   Qn1 Quebec nonchilled           250   34.8  1
1.350   Qn1 Quebec nonchilled           350   37.2  1
1.500   Qn1 Quebec nonchilled           500   35.3  1
1.675   Qn1 Quebec nonchilled           675   39.2  1
```

See also the very simple functions `stack` and `unstack`.

B.7.2 Summarizing by groups

We often want to summarize a column of data *by groups* identified in another column. Here I summarize CO_2 uptake by the means of each experimental treatment, chilling. The code below provides the column to be summarized (`uptake`), a vector (or list of vectors) containing the group id's, and the function to use to summarize each subset (means). We calculate the mean CO_2 uptake for each group.

```
> tapply(CO2[["uptake"]], list(CO2[["Treatment"]]),
+     mean)
```

```
nonchilled    chilled
     30.64      23.78
```

We can get fancier, as well, with combinations of groups, for each combination of Type and Treatment.

```
> tapply(CO2[["uptake"]], list(CO2[["Treatment"]],
+     CO2[["Type"]]), sd)
```

```
           Quebec Mississippi
nonchilled  9.596       7.402
chilled     9.645       4.059
```

We can also define a function on-the-fly to calculate both mean and standard deviation of Type and Treatment combination. We will need, however, to define groups differently, by creating the interaction of the two factors.

```
> tapply(CO2[["uptake"]], list(CO2[["Treatment"]],
+     CO2[["Type"]]), function(x) c(mean(x), sd(x)))
```

```
           Quebec    Mississippi
nonchilled Numeric,2 Numeric,2
chilled    Numeric,2 Numeric,2
```

See also `by` that actually uses `tapply` to operate on data frames.

When we summarize data, as in `tapply`, we often want the result in a nice neat data frame. The function `aggregate` does this. Its use is a bit like `tapply` — you provide (i) the numeric columns of a data frame, or a matrix, (ii) a list of named factors by which to organize the responses, and then (iii) the function to summarize (or aggregate) the data. Here we summarize both concentration and uptake.

```
> aggregate(CO2[, 4:5], list(Plant = CO2[["Plant"]]),
+     mean)
```

```
   Plant conc uptake
1    Qn1  435  33.23
2    Qn2  435  35.16
3    Qn3  435  37.61
4    Qc1  435  29.97
5    Qc3  435  32.59
6    Qc2  435  32.70
7    Mn3  435  24.11
8    Mn2  435  27.34
9    Mn1  435  26.40
10   Mc2  435  12.14
11   Mc3  435  17.30
12   Mc1  435  18.00
```

A separate package entitled `reshape` supplies some very elegant and intuitive approaches to the sorting, reshaping and aggregating of data frames. I typically use the `reshape` *package* (with functions `melt` and `cast`), rather than the `reshape` function supplied in the `stat`. I do so merely because I find it a little more intuitive. R also has strong connections to relational database systems such as MySQL.

B.8 Getting Data out of and into the Workspace

We often want to get data into R, and we sometimes want to get it out, as well. Here we start with the latter (referred to as *writing* data), and finish with the former (referred to as *reading* data).

Here I create a data frame of numbers, and write it to a text file in two different formats. The first is a file where the observations in each row are separated by tabs, and the second separates them by commas.

```
> dat <- data.frame(Name = rep(c("Control", "Treatment"),
+     each = 5), First = runif(10), Second = rnorm(1))
> write.table(dat, file = "dat.txt")
> write.csv(dat, file = "dat.csv")
```

Open these in a spreadsheet such as Calc (in OpenOffice and NeoOffice). We can then read these into R using the `read.*` family of functions.

```
> dat.new <- read.csv("dat.csv")
> dat.new2 <- read.table("dat.txt", header = TRUE)
```

These objects will both be data frames.

Now let's get a statistical summary and export that.

```
> mod.out <- summary(aov(First ~ Name, data = dat))
> mod.out[[1]]
```

```
            Df Sum Sq Mean Sq F value Pr(>F)
Name         1 0.1562  0.1562    4.44  0.068
Residuals    8 0.2814  0.0352
```

```
> write.csv(mod.out[[1]], "ModelANOVA.csv")
```

Open this in a spreadsheet, such as Calc, in OpenOffice, or in any other application.

See also the `xtable` package for making tables in LaTeX or HTML formats.

B.9 Probability Distributions and Randomization

R has a variety of probability distributions built-in. For the normal distribution, for instance, there are four functions:

`dnorm` The probability density function, that creates the widely observed bell-shaped curve.

`pnorm` The cumulative probability function that we usually use to describe the probability that a test statistic is greater than or equal to a critical value.

`qnorm` The quantile function that takes probabilities as input.

`rnorm` A random number generator which draws values (quantiles) from a distribution with a specified mean and standard deviation.

For each of these, default parameter values return the standard normal distribution ($\mu = 0$, $\sigma = 1$), but these parameters can be changed.

Here we have the 95% confidence intervals.

```
> qnorm(p = c(0.025, 0.975))
```

```
[1] -1.96  1.96
```

Next we create a histogram using 20 random draws from a normal distribution with a mean of 11 and a standard deviation of 6; we overlay this with the probability density function (Fig. B.2).

```
> myplot <- hist(rnorm(20, m = 11, sd = 6), probability = TRUE)
> myplot

$breaks
[1]  0  5 10 15 20 25

$counts
[1] 1 8 6 4 1

$intensities
[1] 0.01 0.08 0.06 0.04 0.01

$density
[1] 0.01 0.08 0.06 0.04 0.01

$mids
[1]  2.5  7.5 12.5 17.5 22.5

$xname
[1] "rnorm(20, m = 11, sd = 6)"

$equidist
[1] TRUE

attr(,"class")
[1] "histogram"

> lines(myplot$mids, dnorm(myplot$mids, m = 11, sd = 6))
```

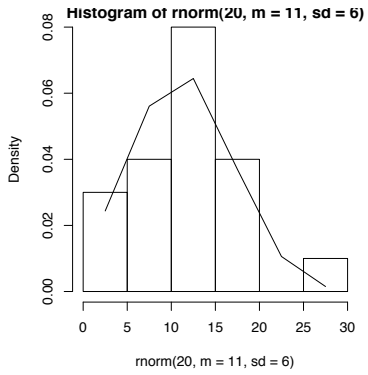


Fig. B.2: Histogram of random numbers drawn from a normal distribution with $\mu = 11$ and $\sigma = 6$. The normal probability density function is drawn as well.

B.10 Numerical integration of ordinary differential equations

In order to study continuous population dynamics, we often would like to integrate complex nonlinear functions of population dynamics. To do this, we need to use numerical techniques that turn the infinitely small steps of calculus, $\mathrm{d}x$, into very small, but finite steps, in order to approximate the change in y, given the change in x, or $\mathrm{d}y/\mathrm{d}x$. Mathematicians and computer scientists have devised very clever ways of doing this very accurately and precisely. In R, the best package for this is `deSolve`, which contains several *solvers* for differential equations that perform numerical integration. We will access these solvers (i.e. numerical integraters) using the `ode` function in the `deSolve` package. This function, `ode`, is a "wrapper" for the underlying suite of functions that do the work. That is, it provides a simple way to use any one of the small suite of functions.

When we have an ordinary differential equation (ODE) such as logistic growth,[3] we say that we "solve" the equation for a particular time interval given a set of parameters and initial conditions or initial population size. For instance, we say that we solve the logistic growth model for time at $t = 0, 1 \ldots 20$, with parameters $r = 1$, $\alpha = 0.001$, and $N_0 = 10$.

Let's do an example with `ode`, using logistic growth. We first have to define a function in a particular way. The arguments for the function must be time, a vector of populations, and a vector or list of model parameters.

```
> logGrowth <- function(t, y, p) {
+     N <- y[1]
+     with(as.list(p), {
+         dN.dt <- r * N * (1 - a * N)
+         return(list(dN.dt))
+     })
+ }
```

Note that I like to convert y into a readable or transparent state variable (N in this case). I also like to use `with` which allows me to use the names of my parameters [157]; this works only is `p` is a vector with named paramters (see below). Finally, we return the derivative as a list of one component.

The following is equivalent, but slightly less readable or transparent.

```
> logGrowth <- function(t, y, p) {
+     dN.dt <- p[1] * y[1] * (1 - p[2] * y[1])
+     return(list(dN.dt))
+ }
```

To solve the ODE, we will need to specify parameters, and initial conditions. Because we are using a vector of named parameters, we need to make sure we name them! We also need to supply the time steps we want.

```
> p <- c(r = 1, a = 0.001)
> y0 <- c(N = 10)
> t <- 1:20
```

[3] *e.g.* $dN/dt = rN(1 - \alpha N)$

Now you put it all into `ode`, with the correct arguments. The output is a matrix, with the first column being the time steps, and the remaining being your state variables. First we load the `deSolve` package.

```
> library(deSolve)
> out <- ode(y = y0, times = t, func = logGrowth, parms = p)
> out[1:5, ]
```

```
     time      N
[1,]    1  10.00
[2,]    2  26.72
[3,]    3  69.45
[4,]    4 168.66
[5,]    5 355.46
```

If you are going to model more than two species, y becomes a vector of length 2. Here we create a function for Lotka-Volterra competition, where

$$\frac{\mathrm{d}N_1}{\mathrm{d}t} = r_1 N_1 (1 - \alpha_{11} N_1 - \alpha_{12} N_2) \tag{B.5}$$

$$\frac{\mathrm{d}N_2}{\mathrm{d}t} = r_2 N_2 (1 - \alpha_{22} N_2 - \alpha_{21} N_1) \tag{B.6}$$

$$\tag{B.7}$$

```
> LVComp <- function(t, y, p) {
+     N <- y
+     with(as.list(p), {
+         dN1.dt <- r[1] * N[1] * (1 - a[1, 1] * N[1] -
+             a[1, 2] * N[2])
+         dN2.dt <- r[2] * N[2] * (1 - a[2, 1] * N[1] -
+             a[2, 2] * N[2])
+         return(list(c(dN1.dt, dN2.dt)))
+     })
+ }
```

Note that `LVComp` assumes that N and r are vectors, and the competition coefficients are in a matrix. For instance, the function extracts the the first element of `r` for the first species (`r[1]`); for the intraspecific competition coefficient for species 1, it uses the element of `a` that is in the first column and first row (`a[1,1]`). The vector of population sizes, N, contains one value for each population *at one time point*. Thus here, the vector contains only two elements (one for each of the two species); it holds only these values, but will do so repeatedly, at each time point. Only the output will contain all of the population sizxes through time.

To integrate these populations, we need to specify new initial conditions, and new parameters for the two-species model.

```
> a <- matrix(c(0.02, 0.01, 0.01, 0.03), nrow = 2)
> r <- c(1, 1)
> p2 <- list(r, a)
> N0 <- c(10, 10)
```

```
> t2 <- c(1, 5, 10, 20)
> out <- ode(y = N0, times = t2, func = LVComp, parms = p2)
> out[1:4, ]

     time     1     2
[1,]    1 10.00 10.00
[2,]    5 35.54 21.80
[3,]   10 39.61 20.36
[4,]   20 39.99 20.01
```

The `ode` function uses a superb ODE solver, `lsoda`, which is a very powerful, well tested tool, superior to many other such solvers. In addition, it has several bells and whistles that we will not need to take advantage of here, although I will mention one, `hmax`. This tells `lsoda` the largest step it can take. Once in a great while, with a very *stiff* ODE (a very wiggly complex dynamic), ODE assumes it can take a bigger step than it should. Setting `hmax` to a smallish number will limit the size of the step to ensure that the integration proceeds as it should.

One of the other solvers in the `deSolve`, `lsodar`, will also return roots (or equilibria), for a system of ODEs, if they exist. Here we find the roots (i.e. the solutions, or equilibria) for a two species enemy-victim model.

```
> EV <- function(t, y, p) {
+     with(as.list(p), {
+         dv.dt <- b * y[1] * (1 - 0.005 * y[1]) -
+             a * y[1] * y[2]
+         de.dt <- a * e * y[1] * y[2] - s * y[2]
+         return(list(c(dv.dt, de.dt)))
+     })
+ }
```

To use `lsodar` to find equilibria, we need to specify a root finding function whose inputs are are the sme of the ODE function, and which returns a scalar (a single number) that determines whether the rate of change ($\mathrm{d}y/\mathrm{d}x$) is sufficiently close to zero that we can say that the system has stopped changed, that is, has reached a steady state or equilibrium. Here we sum the absolute rates of change of each species, and then subtract 10^{-10}; if that difference is zero, we decide that, for all pratcial purposes, the system has stopped changing.

```
> rootfun <- function(t, y, p) {
+     dstate <- unlist(EV(t, y, p))
+     return(sum(abs(dstate)) - 1e-10)
+ }
```

Note that `unlist` changes the `list` returned by `EV` into a simple vector, which can then be summed.

Next we specify parameters, and time. Here all we want is the root, so we specify that we want the value of y after a really long time ($t = 10^{10}$). The `lsodar` function will stop sooner than that, and return the equilibrium it finds, and the time step at which it occurred.

```
> p <- c(b = 0.5, a = 0.02, e = 0.1, s = 0.2)
> t <- c(0, 1e+10)
```

Now we run the function.

```
> out <- ode(y = c(45, 200), t, EV, parms = p, rootfun = rootfun,
+     method = "lsodar")
> out[, ]

      time   1     2
[1,]   0.0  45 200.0
[2,] 500.8 100  12.5
```

Here we see that the steady state population sizes are $V = 100$ and $E = 12.5$, and that given our starting point, this steady state was achieved at $t = 500.8$. Other information is available; see `?lsodar` after loading the `deSolve package`.

B.11 Numerical Optimization

We frequently have a function or a model that we think can describe a pattern or process, but we need to "play around with" the numerical values of the constants in order to make the right shape with our function/model. That is, we need to find the value of the constant (or constants) that create the "best" representation of our data. This problem is known as *optimization*.

Optimization is an entire scientific discipline (or two). It boils down to quickly and efficiently finding parameters (i.e. constants) that meet our criteria. This is what we are doing when we "do" statistics. We fit models to data by telling the computer the structure of the model, and asking it to find values of the constants that minimize the residual error.

Once you have a model of the reality you want to describe, the basic steps toward optimization we consider are (i) create an *objective function*, (ii) use a routine to *minimize* (or *maximize*) the objective function through optimal choice of parameter values, and (iii) see if the "optimal" parameters values make sense, and perhaps refine and interpret them.

An *objective function* compares the data to the predicted values from the model, and returns a quantitative measure of their difference. One widely used objective function the *least-squares criterion*, that is, the objective function is the average or the sum of the squared deviations between the model values and the data — just like a simple ANOVA might. An optimization routine then tries to find model parameters that minimize this criterion.

Another widely used objective function is the likelihood function, or *maximum likelihood*. The likelihood function uses a probability distribution of our choice (often the normal distribution). The objective function then calculates the collective probability of observing those data, given the parameters and fit of the model. In other words, we pretend that the model and the predicted values are true, measure how far off each datum is from the predicted value, and then use a probability distribution to calculate the probability of seeing each datum. It then multiplies all those probabilities to get the *likelihood* of

observing those data, given the selected parameters. An optimization routine then tries to find model parameters that maximize this likelihood. In practice, it is more computationally stable to calculate the negative of the sum of the logarithms of the probabilities, and try to minimize that quantity, rather than maximize the likelihood — but in principle they are they same thing.

Once we have an objective function, an optimization routine makes educated guesses regarding good values for the parameters, until it finds the best values it can, those which minimize the objective function. There are a great variety of optimization routines, and they all have their strengths. One important tradeoff they exhibit is that the fastest and most accurate methods are sometimes the least able to handle difficult data [13]. Below, we rely on a combination to take advantage of the strengths of each type.

Here we introduce two of R's general purpose functions in the `base` package, `optimize` and `optim`, and another, in the `bbmle` package, `mle2` [13]. The function `optimize` should be used where we are in search of one parameter; use others when more than one parameter is being optimized. There are many other optimization routines in R, but we start here[4].

Here we start with one of R's general optimization functions, the one designed for finding a single parameter. Let us find the mean ($\bar{x}$) of some data through optimization. We will start with data, `y`, and let our conceptual model of that data be μ, the mean. We then create a *objective function* whose output will get smaller as the parameter of our model approaches the value we want. Poughly speaking, the mean is the value that minimizes the total difference between all the data and the itself. We will use the least-squares criterion, where the sum of all the squared deviations reaches a minimum when μ approaches the mean.

```
> y <- c(1, 0:10)
> f <- function(y, mu) {
+     sum((y - mu)^2)
+ }
```

Our function, `f`, subtracts μ from each value of y, squares each of these differences, and then sums these squared differences, to get the sum of squares. Our goal is to minimize this. If we guessed at it by hand, we would get this (Fig. B.3).

```
> guesses <- seq(4, 6, by = 0.05)
> LS.criterion <- sapply(guesses, function(mu) f(mu = mu,
+     y = y))
> plot(guesses, LS.criterion, type = "l")
```

Fig. B.3 shows us that the minimum of the objective function occurs when `mu` is a little over 4.5. Now let's let R minimize our least squared deviations. With `optimize`, we provide the function first, we then provide a range of possible values for the parameter of interest, and then give it the values of parameters or data used by the function, other than the parameter we want to fit.

[4] Indeed, all statistical models are fancy optimization routines.

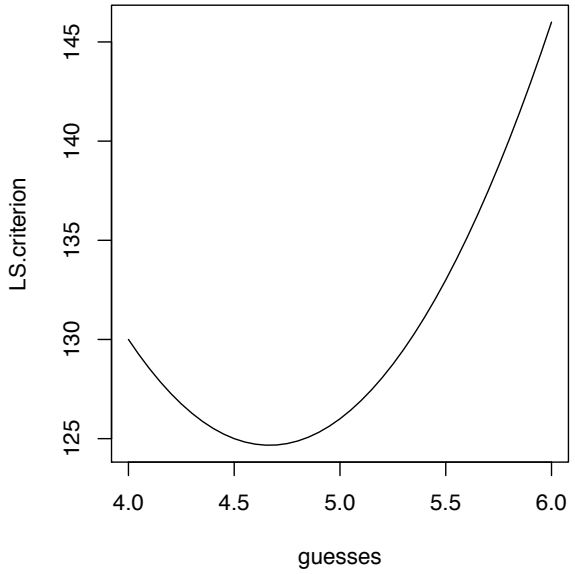


Fig. B.3: Illustration of the least squares criterion. Our objective function returns (i.e. generates) the squared deviations between the fitted model and the data. Optimization minimizes the criterion ("LS.criterion") and thereby finds the right guess (x axis).

```
> (results <- optimize(f, c(0, 10), y = y))

$minimum
[1] 4.667

$objective
[1] 124.7
```

We see that `optimize` returns two components in a list. The first is called `minimum`, which is the parameter value that causes our function `f` to be at a minimum. The second component, `objective` is the value of `f` when mu = 4.667.

Next we demonstrate `mle2`, a function for maximum likelihood estimation. Maximum likelihood relies on probability distributions to find the probability of observing a particular data set, assuming the model is correct. This class of optimization routines finds the parameters that maximize that probability.

Let us solve the same problem as above. For the same data, `y`, we create a maximum likelihood function to calculate the mean. In maximum likelihood, we actually minimize the negative logarithm of the likelihood because it is more computationally stable — the same parameters that minimize the negative log-likelihood also maximize the likelihood. We assume that the data are normally distributed, so it makes sense to assume that the probabilities derive from the normal probability density function.

```
> LL <- function(mu, SD) {
+     -sum(dnorm(y, mean = mu, sd = SD, log = TRUE))
+ }
```

This objective function calculates the negative logarithm of the probability density of each datum, given a particular mean and standard deviation, `mu, SD`. The optimization routine, `mle2`, then finds `mu` and `SD` that minimize the negative log-likelihood of those data.

```
> library(bbmle)
> (fit <- mle2(LL, start = list(mu = 5, SD = 1), control = list(maxit = 10^5)))

Call:
mle2(minuslogl = LL, start = list(mu = 5, SD = 1), control = list(maxit = 10^5))

Coefficients:
   mu    SD
4.667 3.223

Log-likelihood: -31.07
```

Another way to put this objective function into `mle2` is with a formula interface.

```
> mle2(y ~ dnorm(mu, sd = SD), start = list(mu = 1,
+     SD = 2))

Call:
mle2(minuslogl = y ~ dnorm(mu, sd = SD), start = list(mu = 1,
    SD = 2))

Coefficients:
   mu    SD
4.667 3.223

Log-likelihood: -31.07
```

We can examine this more closely, examing the probablities associated with the profile confidence intervals.

```
> summary(fit)

Maximum likelihood estimation

Call:
mle2(minuslogl = LL, start = list(mu = 5, SD = 1), control = list(maxit = 10^5))

Coefficients:
   Estimate Std. Error z value   Pr(z)
mu    4.667      0.930    5.02 5.3e-07
SD    3.223      0.658    4.90 9.6e-07

-2 log L: 62.14

> pr <- profile(fit)
```

```
> par(mar = c(5, 4, 3, 2))
> plot(pr)
```

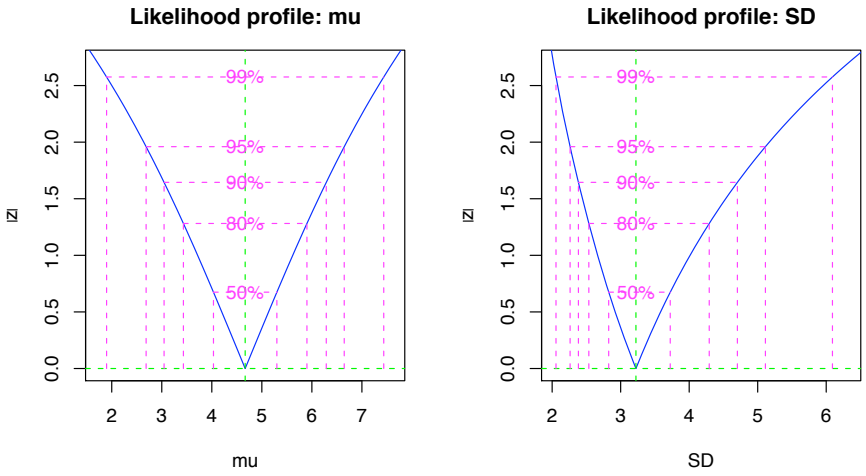


Fig. B.4: Profile confidence intervals for various limits, based on `mle2`.

Often we have reason to limit parameters to particular bounds. Most often, we may need to ensure that a parameter is greater than zero, or less than zero, or less often between zero and one. Sometimes we have a rationale based on physical or biological constraints that will limit a parameter within particular values.

To constrain parameters, we could use a routine that applies constraints directly (see particular optimization methods under `mle2`,`nlminb`, and `optim`). We could also transform the parameters, so that the optimizer uses the transformed version, while the ODE model uses the parameters in their original units. For instance, we could let the optimizer find the best value of a logarithm of our parameter that allows the original parameter to make model predictions that fit the data. Another consequence of using logarithms, rather than the original scale is that it facilitates computational procedures in estimating vary large and very small numbers. An example helps make this clear — see an extended example in Chap. 6, on disease models.

B.12 Derivatives

We can use `deriv` and `D` to have R provides derivatives. First we supply an expression, and then we get gradients.

```
> host1 <- expression(R * H * (1 + a * P)^-k)
> D(host1, "H")
```

```
R * (1 + a * P)^-k
```

B.13 Graphics

R is well known for its graphics capabilities, and entire books have been written of the subject(s). For beginners, however, R can be frustrating when compared to the point-and-click systems of most graphics "packages." This frustration derives from two issues. First, R's graphics have of a learning curve, and second, R requires us to type in, or code, our specifications. The upsides of these are that R has infinite flexibility, and total replicability, so that we get exactly the right figure, and the same figure, every time we run the same code.

B.13.1 `plot`

The most used graphics function is `plot`. Here I demonstrate several uses.

First let's just create the simplest scatterplot (Fig. B.5a).

```
> data(trees)
> attach(trees)
> plot(Girth, Height)
```

To this we can add a huge variety of variation, using arguments to `plot`.

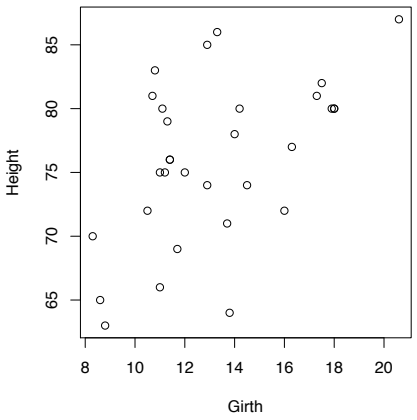
B.13.2 Adding points, lines and text to a plot

After we have started a plot, we may want to add more data or information. Here set up a new graph without plotting points, add text at each point, then more points, a line and some text.

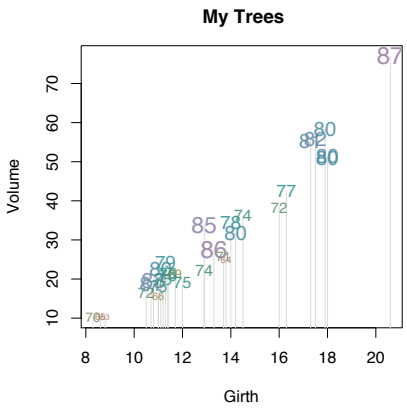
```
> par(mar = c(5, 4, 3, 2))
> plot(Girth, Volume, type = "n", main = "My Trees")
> points(Girth, Volume, type = "h", col = "lightgrey",
+     pch = 19)
```

Now we want to add points for these data, using the tree heights as the plotting symbol. We are going to use an alternate coloring system, designed with human perception in mind (`hcl`). We scale the colors so that the hue varies between 30 and 300, depending on the height of the tree; I allow the symbols to be transparent (90% opaque) overlapping. I also allow the size of the numbers to vary with height (`cex = 0.5 + hts`) Last, we add a legend (Fig. B.5b).

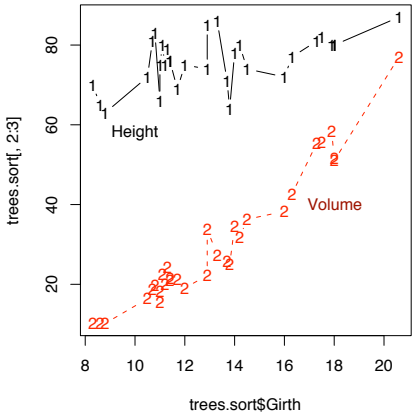
```
> hts <- (Height - min(Height))/max(Height - min(Height))
> my.colors <- hcl(h = 30 + 270 * hts, alpha = 0.9)
> text(Girth, Volume, Height, col = my.colors, cex = 0.5 +
+     hts)
```



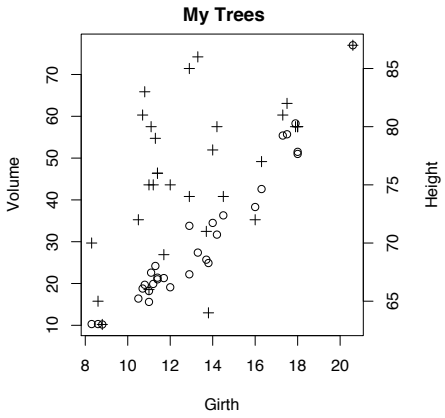
(a) Simple



(b) With Colors



(c) Two Y vars.



(d) Two Y axes

Fig. B.5: See code for graphics parameters used to generate these plots. Fig. (b) uses an alternate color scheme that provides human perception-adjusted hsv (hue, saturation, and value) specification.

B.13.3 More than one response variable

We often plot more than one response variable on a single axis. We could use `lines` or `points` to add each additional variable. We could also use `matplot` to plot a matrix of variables *vs.* one predictor (Fig. B.5c).

```
> trees.sort <- trees[order(trees$Girth, trees$Height),
+     ]
> matplot(trees.sort$Girth, trees.sort[, 2:3], type = "b")
> text(18, 40, "Volume", col = "darkred")
> text(10, 58, "Height")
```

Table B.1: Commonly used arguments to `plot`. See help pages at `?plot` and `?plot.default` for more information.

Argument	Meaning
`type`	Determines the type of X-Y plot, for example **p**, **l**, **s**, for points, lines, stair-step, and none, respectively. "None" is useful for setting up a plotting region upon which to elaborate (see example below). Defaults to **p**; see ?plot.default for other types.
`axes`	Indicates whether to plot the axes; defaults to TRUE. Useful if you want more control over each axis by using the `axis` function separately (see below).
`pch`	Point character (numeric value, 1–21). This can be a single value for an entire plot, or take on a unique value for each point, or anything in between. Defaults to 1. *To add text more than one character in length, for instance a species name, we can easily add text to a plot at each point (see the next section).*
`lty`	Line type, such as solid (1), dashed (2), etc. Defaults to 1.
`lwd`	Line width (numeric value usually 0.5–3; default is 1).
`col`	Color; can be specified by number (e.g., 2), or character (e.g. "red"). Defaults to 1 ("black"). R has tremendous options for color; see `?hcl`.
`main, ylab, xlab`	Text for main title, or axis labels.
`xlim, ylim`	Limits for x and y axes, e.g. `ylim=c(0, 1.5)` sets the limits for the y-axis at zero and 1.5. Defaults are calcualted from the data.
`log`	Indicates which axes should use a (natural) logarithm scale, e.g. `log = 'xy'` causes both axes to use logarithmic scales.

We frequently want to add a second y-axis to a graph that has a different scale (Fig. B.5d). The trick we use here is that we plot a graph, but then tell R we want to do the next command "*. . . as if it was on a new device*"[5] while it really is not. We overlay what we just did with new stuff, without clearing the previous stuff.

For our example, let's start with just X and our first Y. Note we also specify extra margin space room on the right hand side, preparing for the second Y axis.

```
> quartz(, 4, 4)
> par(mar = c(5, 4, 2, 4))
> plot(Girth, Volume, main = "My Trees")
```

Now we try our trick. We draw a new plot "as if" it were a new graph. We use the same X values, and the new Y data, and we also specify no labels. We also use a different line type, for clarity.

```
> par(new = TRUE)
> plot(Girth, Height, axes = FALSE, bty = "n", xlab = "",
+     ylab = "", pch = 3)
```

[5] From the `par` help page.

Now we put the new Y values on the fourth side, the right hand Y axis. We add a Y axis label using a function for *marginal text* (Fig. B.5d).

```
> axis(4)
> mtext("Height", side = 4, line = 3)
```

```
> par(mar = c(5, 4, 2, 4))
> plot(Girth, Volume, main = "My Trees")
> par(new = TRUE)
> plot(Girth, Height, axes = FALSE, bty = "n", xlab = "",
+     ylab = "", pch = 3)
> axis(4)
> mtext("Height", side = 4, line = 3)
```

B.13.4 Controlling Graphics Devices

When we make a graph with the `plot` function, or other function, it will typically open a graphics window on the computer screen automatically; if we desire more control, we can use several functions to be more deliberate. We create new graphics "devices" or graphs in several ways, including the functions `windows()` (Microsoft Windows OS), `quartz()` (Mac OS), `x11()` (X11 Window system). For instance, to open a "graphics device" on a Mac computer that is 5 inches wide and 3 inches tall, we write

```
> quartz(width = 5, height = 3)
```

To do the same thing on a computer running Windows, we type

```
> windows(width = 5, height = 3)
```

To control the *par*ameters of the graph, that is, what it looks like, aside from data, we use arguments to the `par` function. Many of these arguments refer to *sides* of the graph. These a numbered 1–4 for the bottom X axis, the left side Y axis, the top, and the right side Y axis. Arguments to `par` are many (see `?par`), and include the following.

`mar` controls the width of margins on each side; units are number of lines of text; defaults to c(5, 4, 4, 2) + 0.1, so the bottom has the most room, and the right hand side has the least room.

`mgp` controls the spacing of the axis title, labels and the actual line itself; units of number of lines of text, and default to c(3, 1, 0), so the axis title sits three lines away from the edge of the plotting region, the axis labels, one line away and the axis line sits at the edge of the plotting region.

`tcl` tick length, as a fraction of the height of a line of text; negative values put the tick marks outside, positive values put the tick marks inside. Defaults to -0.5.

We can build each side of the graph separately by initiating a graph but not plotting axes `plot(..., axes = FALSE)`, and then adding the axes separately. For instance, `axis(1)` adds the bottom axis.

Last, we can use `layout` to make graph with several smaller subgraphs (see also (`mfrow` and `mfcol` arguments to `par` and the function `split.screen`). The

function `layout` takes a matrix as its argument, the matrix contains a sequence of numbers that tells R how to fill the regions Graphs can fit in more than one of these regions if indicated by the same number.

Here we create a compound graphic organized on top of a 4×4 grid; it will have two rows, will be be filled in by rows. The first graph will be the upper left, the second the upper right, and the third will fill the third and fourth spots in the second. We will fill each with a slightly different plot of the same data (Fig. B.6).

```
> quartz(, 5, 5)
> layout(matrix(c(1, 2, 3, 3), nrow = 2, byrow = TRUE))
> plot(Girth, Height)
```

Now we add the second and third ones but with different settings.

```
> par(mar = c(3, 3, 1, 1), mgp = c(1.6, 0.2, 0), tcl = 0.2)
> plot(Girth, Height)
> par(mar = c(3, 3, 2, 1), mgp = c(1.6, 0.2, 0), tcl = 0.2)
> plot(Girth, Height, axes = FALSE, xlim = c(8, 22))
> axis(1, tcl = -0.3)
> axis(2, tick = F)
> rug(Height, side = 2, col = 2)
> title("A Third, Very Wide, Plot")
```

B.13.5 Creating a Graphics File

Now that you have made this beautiful thing, I suppose you would like to stick it into a manuscript. One way to get graphics out of R and into something else (presentation software, a manuscript), is to create a graphics device, and then save it with `dev.print` in a format that you like, such as PDF, postscript, PNG, or JPEG.

For instance, we might do this to save a graphics file in our working directory.

```
> getwd()
> quartz(, 4, 4)
> plot(Height, Volume, main = "Tree Data")
> dev.print(pdf, "MyTree.pdf")
```

This should have saved a small PDF figure in your current working directory, returned by `getwd`.

You will have to find your own way to make graphics files that suits your operating system, your preferred applications, and your personality.

B.14 Graphical displays that show distributions

Here we take a quick look at ways to reveal distributions of data. First, two views to see in the Console, a six number summary of quantiles and the mean, and the good ol' stem and leaf plot, a favorite of computational botanists everywhere.

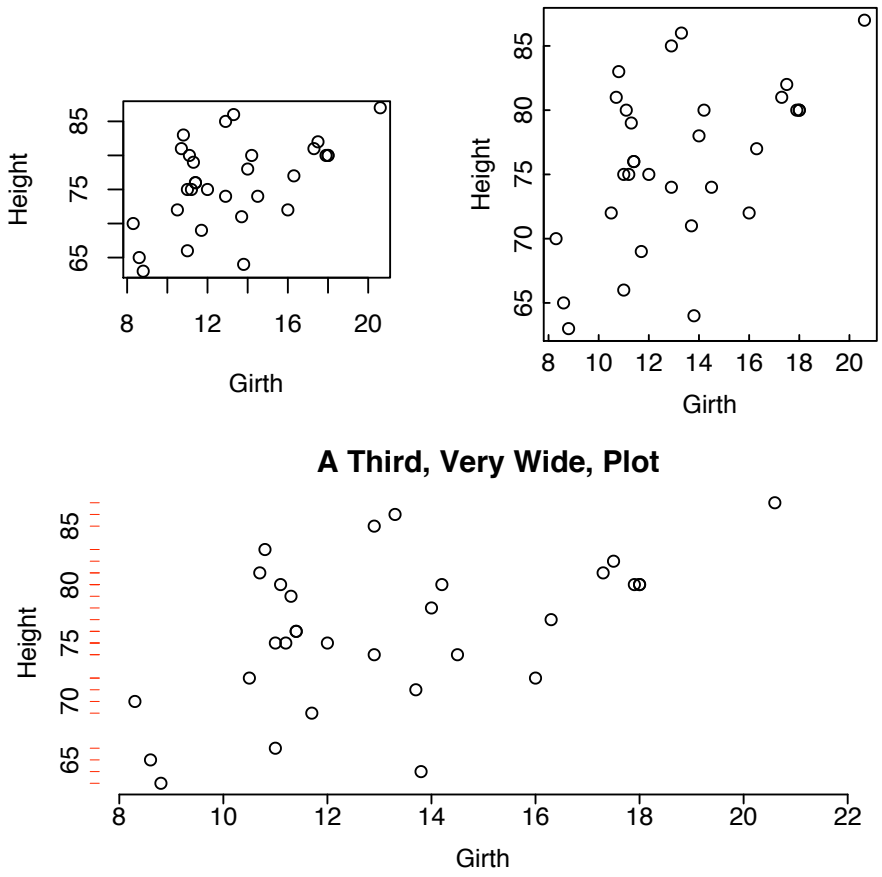


Fig. B.6: A variety of examples with different graphics *par*ameters.

```
> summary(Girth)

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
    8.3    11.0    12.9    13.2    15.2    20.6

> stem(Girth)

  The decimal point is at the |

   8 | 368
  10 | 57800123447
  12 | 099378
  14 | 025
  16 | 03359
  18 | 00
  20 | 6
```

Here we will create 4 various plots revealing different ways to look at your data, each with a couple bells and whistles. For kicks, we put them into a single compound figure, in a "layout" composed of a matrix of graphs.

```
> layout(matrix(c(1, 2, 2, 3, 4, 4), nrow = 2, byrow = TRUE))
> plot(1:length(Girth), Girth, xlab = "Order of Sample Collection?")
> hist(Girth, prob = TRUE)
> rug(Girth)
> lines(density(Girth))
> boxplot(Girth, main = "Boxplot of Girth")
> points(jitter(rep(1, length(Girth))), Girth)
> qqnorm(log(Girth))
> qqline(log(Girth))
> title(sub = "Log transformed data")
```

B.15 Eigenanalysis

Performing eigenanalysis in Ris easy. We use the `eigen` function which returns a list with two components. The first named component is a vector of eigenvalues and the second named component is a matrix of corresponding eigenvectors. These will be numeric if possible, or complex, if any of the elements are complex numbers.

Here we have a typical demographic stage matrix.

```
> A <- matrix(c(0, 0.1, 10, 0.5), nrow = 2)
> eig.A <- eigen(A)
> str(eig.A)
```

```
List of 2
 $ values : num [1:2] 1.28 -0.78
 $ vectors: num [1:2, 1:2] -0.9919 -0.127 -0.997 0.0778
```

Singular value decomposition (SVD) is a generalization of eigenanalysis and is used in Rfor some applications where eigenanalysis was used historically, but where SVD is more numerically accurate (`prcomp` for principle components analysis).

B.16 Eigenanalysis of demographic versus Jacobian matrices

Eigenanalyses of demographic and Jacobian matrices are worth comparing. In one sense, they have similar meanings — they both describe the asymptotic (long-term) properties of a system, either population size (demographic matrix) or a perturbation at an equilibrium. The quantitative interpretation of the eigenvalues will therefore differ.

In the case of the stage (or age) structured demographic model, the elements of the demographic matrix are discrete per capita increments of change over a

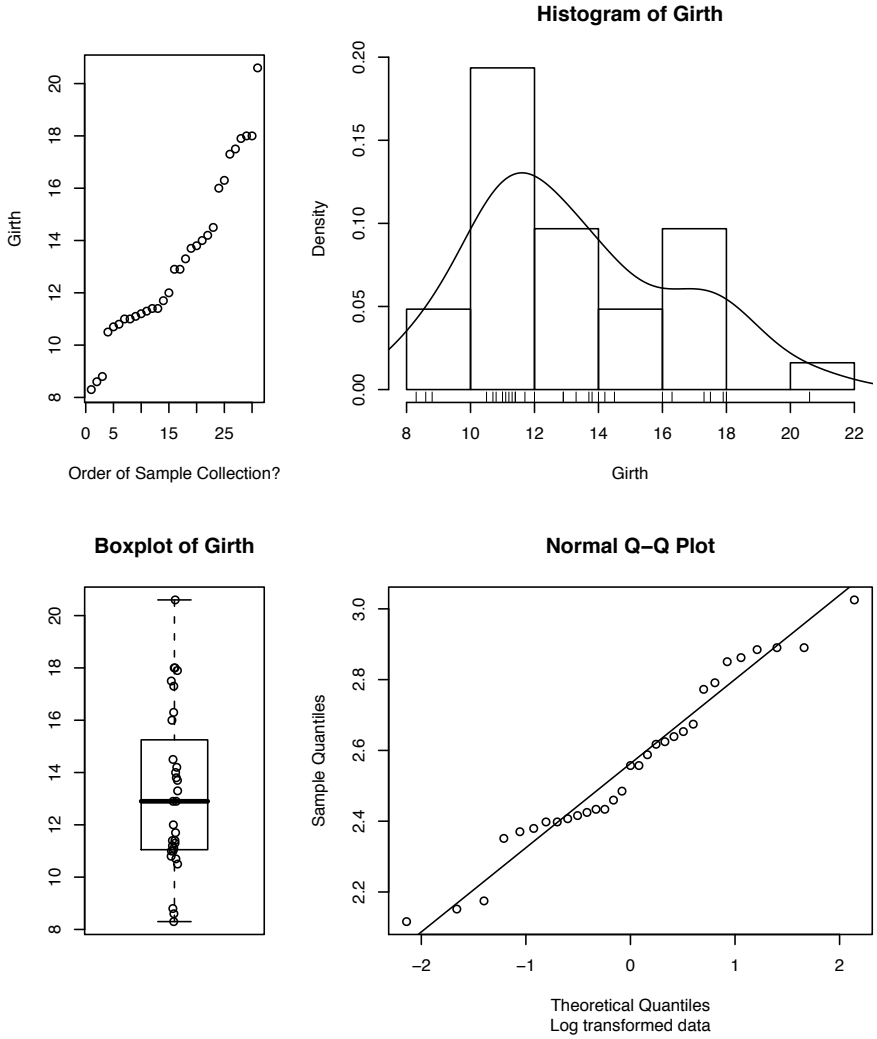


Fig. B.7: Examples of ways to look at the distribution of your data. See `?hist`, for example, for more information.

specified time interval. This is directly analogous to the finite rate of increase, λ, in discrete unstructured models. (Indeed, an unstructured discrete growth model is a stage-structured model with one stage). Therefore, the eigenvalues of a demographic matrix will have the same units — a per capita increment of change. That is why the dominant eigenvalue has to be greater than 1.0 for the population to increase, and less than 1 (not merely less than zero) for the population to decline.

In the case of the Jacobian matrix, comprised of continuous partial differential equations, the elements are per capita *instantaneous* rates of change. As differential equations, they describe the *instantanteous* rates of change, analogous to r. Therefore, values greater than zero indicate increases, and values less than zero indicate decreases. Because these rates are evaluated at an equilibrium, the equilibrium acts like a new zero — positive values indicate growth away from the equilibrium, and negative values indicate shrinkage back toward the equilibrium. When we evaluate these elements at the equilibrium, the numbers we get are in the same units as r, where values greater than zero indicate increase, and values less than zero indicate decrease. The change they describe is the instantaneous per capita rate of change of each population with respect to the others. The eigenvalues summarizing all of the elements Jacobian matrix thus must be less than zero for the disturbance to decline.

So, in summary, the elements of a demographic matrix are discrete increments over a real time interval. Therefore its eigenvalues represent relative per capita growth rates a discrete time interval, and we interpret the eigenvalues with respect to 1.0. On the other hand, the elements of the Jacobian matrix are instantaneous per captia rates of change evaluated at an equilibrium. Therefore its eigenvalues represent the per capita *instantaneous* rates of change of a tiny perturbation at the equilibrium. We interpret the eigenvalues with respect to 0 indicating whether the perturbation grows or shrinks.

B.17 Symbols used in this book

I am convinced that one of the biggest hurdles to learning theoretical ecology — and the one that is easiest to overcome — is to be able to "read" and hear them in your head. This requires being able to pronounce Greek symbols. Few of us learned how to pronounce "α" in primary school. Therefore, I provide here an incomplete simplistic American English pronunciation guide for (some of) the rest of us, for symbols in this book. Only a few are tricky, and different people will pronounce them differently.

Table B.2: Symbols and their pronunciation; occasional usage applies to lowercase, unless otherwise specified. A few symbols have common variants. Any symbol might be part of any equation; ecologists frequently ascribe other meanings which have to be defined each time they are used. See also http://en.wikipedia.org/wiki/Greek_letters

Symbol	Spelling	Pronunciation; occasional or conventional usage
A, α	alpha	al'-fa; point or local diversity (or a parameter in the logseries abundance distribution)
B, β	beta	bay'-ta; turnover diversity
Γ, γ	gamma	gam'-ma; regional diversity
Δ, δ, ∂	delta	del'-ta; change or difference
E, ϵ, ε	epsilon	ep'-si-lon; error
Θ, θ	theta	thay'-ta ("th" as in "thanks"); in neutral theory, biodiversity.
Λ, λ	lambda	lam'-da; eigenvalues, and finite rate of increase
M, μ	mu	meeoo, myou; mean
N, ν	nu	noo, nou
Π, π	pi	pie; uppercase for product (of the elements of a vector)
P, ρ	rho	row (as in "a boat"); correlation
Σ, σ, ς	sigma	sig'-ma; standard deviation (uppercase is used for summation)
T, τ	tau	(sounds like what you say when you stub your toe - "Ow!" but with a "t").
Φ, ϕ	phi	fie, figh
X, χ	chi	kie, kigh
Ψ, ψ	psi	sie, sigh
Ω, ω	omega	oh-may'-ga; degree of omnivory

References

1. D. Alonso, R. S. Etienne, and A. J. McKane. The merits of neutral theory. *Trends in Ecology & Evolution*, 21:451–457, 2006.
2. D. Alonso and A. J. McKane. Sampling hubbell's neutral theory of biodiversity. *Ecology Letters*, 7:901–910, 2004.
3. J. Antonovics and H.M. Alexander. Epidemiology of anther-smut infection of *Silene alba* (= *S. latifolia*) caused by *Ustilago violacea* – patterns of spore deposition in experimental populations. *Proceedings of the Royal Society B-Biological Sciences*, 250:157–163, 1992.
4. R. A. Armstrong. Fugitive species: Experiments with fungi and some theoretical considerations. *Ecology*, 57:953–963, 1976.
5. O. Arrhenius. Species and area. *Journal of Ecology*, 9:95–99, 1921.
6. J. P. Bakker and F. Berendse. Constraints in the restoration of ecological diversity in grassland and heathland communities. *Trends in Ecology & Evolution*, 14:63–68, 1999.
7. F. A. Bazzaz. *Plants in Changing Environments*. Cambridge University Press, Boston, 1996.
8. L. Becks, F. M. Hilker, H. Malchow, K. Jurgens, and H. Arndt. Experimental demonstration of chaos in a microbial food web. *Nature*, 435:1226–1229, 2005.
9. G. Bell. The distribution of abundance in neutral communities. *The American Naturalist*, 155:606–617, 2000.
10. G. Bell. Neutral macroecology. *Science*, 293(5539):2413–2418, 2001.
11. E. L. Berlow, A. M. Neutel, J. E. Cohen, P. C. de Ruiter, B. Ebenman, M. Emmerson, J. W. Fox, V. A. A. Jansen, J. I. Jones, G. D. Kokkoris, D. O. Logofet, A. J. McKane, J. M. Montoya, and O. Petchey. Interaction strengths in food webs: Issues and opportunities. *Journal of Animal Ecology*, 73:585–598, 2004.
12. E. J. Berry, D. L. Gorchov, B. A. Endress, and M. H. H. Stevens. Source-sink dynamics within a plant population: the impact of substrate and herbivory on palm demography. *Population Ecology*, 50:63–77, 2008.
13. B. M. Bolker. *Ecological Models and Data in R*. Princeton University Press, Princeton, NJ, 2008.
14. B. M. Bolker, S. W. Pacala, and C. Neuhauser. Spatial dynamics in model plant communities: What do we really know? *American Naturalist*, 162:135–148, 2003.
15. J.H. Brown. *Macroecology*. The University of Chicago Press, Chicago, 1995.
16. J.H. Brown and D.W. Davidson. Competition between seed-eating rodents and ants in desert ecosystems. *Science*, 196:880–882, 1977.

17. J.H. Brown and A. Kodric-Brown. Turnover rate in insular biogeography: effect of immigration on extinction. *Ecology*, 58:445–449, 1977.
18. K P. Burnham and D. R. Anderson. *Model Selection and Multimodel Inference: A Practical Information-Theoretic Approach*. Springer, New York, 2002.
19. J. E. Byers, K. Cuddington, C. G. Jones, T. S. Talley, A. Hastings, J. G. Lambrinos, J. A. Crooks, and W. G. Wilson. Using ecosystem engineers to restore ecological systems. *Trends in Ecology & Evolution*, 21:493–500, 2006.
20. C.E. Caceres. Temporal variation, dormancy, and coexistence: A field test of the storage effect. *Proceedings of the National Academy of Sciences, U.S.A.*, 94:9171–9175, 1997.
21. T. J. Case. *An Illustrated Guide to Theoretical Ecology*. Oxford University Press, Inc., New York, 2000.
22. H. Caswell. Community structure: A neutral model analysis. *Ecological Monographs*, 46:327–354, 1976.
23. H. Caswell. *Matrix Population Models: Construction, Analysis, and Interpretation*. Sinauer Associates, Inc., Sunderland, MA, USA, 2nd edition, 2001.
24. J. M. Chase and M. Leibold. *Ecological Niches: Linking Classical and Contemporary Approaches*. University of Chicago Press, Chicago, 2003.
25. X. Chen and J. E. Cohen. Global stability, local stability and permanence in model food webs. *Journal of Theoretical Biology*, 212:223–235, 2001.
26. P. L. Chesson. Multispecies competition in variable environments. *Theoretical Population Biology*, 45:227–276, 1994.
27. P. L. Chesson. General theory of competitive coexistence in spatially-varying environments. *Theoretical Population Biology*, 58:211–237, 2000.
28. P. L. Chesson. Mechanisms of maintenance of species diversity. *Annual Review of Ecology and Systematics*, 31:343–366, 2000.
29. P. L. Chesson. Quantifying and testing coexistence mechanisms arising from recruitment fluctuations. *Theoretical Population Biology*, 64:345–357, 2003.
30. P. L. Chesson and R. R. Warner. Environmental variability promotes coexistence in lottery competitive systems. *The American Naturalist*, 117:923–943, 1981.
31. J. S. Clark, S. LaDeau, and I. Ibanez. Fecundity of trees and the colonization-competition hypothesis. *Ecological Monographs*, 74:415–442, 2004.
32. J. S. Clark and J. S. McLachlan. Stability of forest biodiversity. *Nature*, 423:635–638, 2003.
33. J. E. Cohen. Human population: The next half century. *Science*, 302:1172–1175, 2003.
34. J.E. Cohen, F. Briand, and C.M. Newman. *Community Food Webs: Data and Theory*, volume 20 of *Biomathematics*. Springer-Verlag, Berlin, 1990.
35. S. L. Collins and S. M. Glenn. Importance of spatial and temporal dynamics in species regional abundance and distribution. *Ecology*, 72:654–664, 1991.
36. R. Condit, N. Pitman, E. G. Leigh, J. Chave, J. Terborgh, R. B. Foster, P. Nunez, S. Aguilar, R. Valencia, G. Villa, H. C. Muller-Landau, E. Losos, and S. P. Hubbell. Beta-diversity in tropical forest trees. *Science*, 295:666–669, 2002.
37. J.H. Connell. Diversity in tropical rain forests and coral reefs. *Science*, 199:1302–1310, 1978.
38. J.H. Connell and W.P. Sousa. On the evidence needed to judge ecological stability or persistence. *The American Naturalist*, 121:789–824, 1983.
39. R.F. Constantino, J.M. Cushing, B. Dennis, and R.A. Desharnais. Experimentally induced transitions in the dynamic behaviour of insect populations. *Nature*, 375:227–230, 1995.
40. J. M. Craine. Reconciling plant strategy theories of grime and tilman. *Journal of Ecology*, 93:1041–1052, 2005.

41. M. J. Crawley and J. E. Harral. Scale dependence in plant biodiversity. *Science*, 291:864–868, 2001.
42. T. O. Crist and J. A. Veech. Additive partitioning of rarefaction curves and species-area relationships: Unifying alpha-, beta- and gamma-diversity with sample size and habitat area. *Ecology Letters*, 9:923–932, 2006.
43. T. O. Crist, J. A. Veech, J. C. Gering, and K. S. Summerville. Partitioning species diversity across landscapes and regions: A hierarchical analysis of alpha, beta, and gamma diversity. *American Naturalist*, 162:734–743, 2003.
44. D. T. Crouse, L. B. Crowder, and H. Caswell. A stage-based population model for loggerhead sea turtles and implications for conservation. *Ecology*, 68:1412–1423, 1987.
45. J. F. Crow and M. Kimura. *An Introduction to Population Genetics Theory*. Harper & Row, New York, 1970.
46. A. C. Davison and D. V. Hinkley. *Bootstrap Methods and Their Applicaiton*. Cambridge Series on Statistical and Probabilistic Mathematics. Cambridge University Press, New York, 1997.
47. J. A. Dunne, R. J. Williams, and N. D. Martinez. Food-web structure and network theory: The role of connectance and size. *Proceedings of the National Academy of Sciences of the United States of America*, 99:12917–12922, 2002.
48. S. P. Ellner and J. Guckenheimer. *Dynamic Models in Biology*. Princeton University Press, Princeton, NJ, 2006.
49. S. P. Ellner and P. Turchin. When can noise induce chaos and why does it matter: A critique. *Oikos*, 111:620–631, 2005.
50. B. A. Endress, D. L. Gorchov, and R. B. Noble. Non-timber forest product extraction: Effects of harvest and browsing on an understory palm. *Ecological Applications*, 14:1139–1153, 2004.
51. R. S. Etienne. A new sampling formula for neutral biodiversity. *Ecology Letters*, 8:253–260, 2005.
52. R. S. Etienne. A neutral sampling formula for multiple samples and an 'exact' test of neutrality. *Ecology Letters*, 10:608–618, 2007.
53. W.J. Ewens. *Mathematical Population Genetics, I. Theoretical Introduction*, volume 27 of *Interdisciplinary Applied Mathematics*. Springer, New York, 2nd ed. edition, 2004.
54. J.M. Facelli, P. Chesson, and N. Barnes. Differences in seed biology of annual plants in arid lands: A key ingredient of the storage effect. *Ecology*, 86:2998–3006, 2005.
55. R. A. Fisher, A. S. Corbet, and C. B. Williams. The relation between the number of species and the number of individuals in a random sample of an animal population. *Jounral of Animal Ecology*, 12:42–58, 1943.
56. G. F. Fussmann, S. P. Ellner, N. G. Hairston, L. E. Jones, K. W. Shertzer, and T. Yoshida. Ecological and evolutionary dynamics of experimental plankton communities. *Advances in Ecological Research*, 37:221–243, 2005.
57. D. L. Gorchov and D. Christensen. Projecting the effect of harvest on the population biology of goldenseal, *Hydrasts canadensis* l., a medicianl herb in Ohio, USA forests. In New York Botanical Garden Society for Economic Botany, editor, *Symposium on Assessing Local Management of Wild Plant Resources: Approaches in Population Biology*, 2002.
58. N. J. Gotelli. *A Primer of Ecology*. Sinauer Associates, Inc., Sunderland, MA, 3rd ed. edition, 2001.
59. N. J. Gotelli and R. K. Colwell. Quantifying biodiversity: procedures and pitfalls in the measurement and comparison of species richness. *Ecology Letters*, 4:379–391, 2001.

60. N. J. Gotelli and G. R. Graves. *Null Models in Ecology*. Smithsonian Institution Press, Washington, DC, 1996.
61. N. J. Gotelli and B. J. McGill. Null versus neutral models: what's the difference? *Ecography*, 29:793–800, 2006.
62. N.G. Gotelli and W.G. Kelley. A general model of metapopulation dynamics. *Oikos*, 68:36–44, 1993.
63. N.J. Gotelli. Metapopulation models: the rescue effect, the propagule rain, and the core-satellite hypothesis. *The American Naturalist*, 138:768–776, 1991.
64. J. L. Green and J. B. Plotkin. A statistical theory for sampling species abundances. *Ecology Letters*, 10:1037–1045, 2007.
65. K. L. Gross and P. A. Werner. Colonizing abilities of "biennial" plant species in relation to ground cover: implications for their distriubtions in a successional sere. *Ecology*, 63:921–931, 1982.
66. N. G. Hairston, Sr. *Ecological Experiments: Purpose, Design, and Execution*. Cambridge Studies in Ecology. Cambridge University Press, Cambridge, UK, 1991.
67. J.M. Halley. Ecology, evolution and 1/f-noise. *Trends in Ecology & Evolution*, 11:33–37, 1996.
68. P. A. Hamback, K. S. Summerville, I. Steffan-Dewenter, J. Krauss, G. Englund, and T. O. Crist. Habitat specialization, body size, and family identity explain lepidopteran density-area relationships in a cross-continental comparison. *Proceeding of the National Academy of Sciences, USA*, 104:8368–8373, 2007.
69. Hankin, R.K.S. *untb: ecological drift under the UNTB*, 2007. R package version 1.3-3.
70. I. Hanski. Dynamics of regional distribution: The core and satellite species hypothesis. *Oikos*, 38:210–221, 1982.
71. J. Harte, T. Zillio, E. Conlisk, and A. B. Smith. Maximum entropy and the state-variable approach to macroecology. *Ecology*, 89:2700–2711, 2008.
72. M.P. Hassell. *The Dynamics of Arthropod Predator-Prey Systems*, volume 13 of *Monographs in Population Biology*. Princeton University Press, Princeton, 1978.
73. A. Hastings. Disturbance, coexistence, history and competition for space. *Theoretical Population Biology*, 18:363–373, 1980.
74. A. Hastings. Transients: the key to long-term ecological understanding? *Trends in Ecology and Evolution*, 19:39–45, 2004.
75. A. Helm, I. Hanski, and M. Partel. Slow response of plant species richness to habitat loss and fragmentation. *Ecology Letters*, 9:72–77, 2006.
76. M. O. Hill. Diversity and evenness: a unifying notion and its consequences. *Ecology*, 54:427–432, 1973.
77. C.S. Holling. Some characteristics of simple types of predation and parasitism. *Canadian Entomologist*, 91:385, 1959.
78. C.S. Holling. Resilience and stability of ecological systems. *Annual Review of Ecology and Systematics*, 4:1–23, 1973.
79. R.D. Holt and G.A Polis. A theoretical framework for intraguild predation. *The American Naturalist*, 149:745–764, 1997.
80. H. S. Horn and R. H. MacArthur. Competition among fugitive species in a harlequin environment. *Ecology*, 53:749–752, 1972.
81. S. P. Hubbell. *A Unified Theory of Biodiversity and Biogeography*. Monographs in Population Biology. Princeton University Press, Princeton, N.J., 2001.
82. S. H. Hurlbert. The nonconcept of species diversity: A critique and alternative parameters. *Ecology*, 52:577–586, 1971.

83. G. C. Hurtt and S. W. Pacala. The consequences of recruitment limitation: Reconciling chance, history, and competitive differences between plants. *Journal of Theoretical Biology*, 176:1–12, 1995.
84. G. E. Hutchinson. *An Introduction to Population Ecology*. Yale University Press, New Haven, CT, 1978.
85. G. R. Huxel and K. McCann. Food web stability: The influence of trophic flows across habitats. *American Naturalist*, 152:460–469, 1998.
86. F. Jabot and J. Chave. Inferring the parameters of the neutral theory of biodiversity using phylogenetic information, and implications for tropical forests. *Ecology Letters*, 12:239–248, 2009.
87. S. A. Juliano. Nonlinear curve fitting: predation and functional response curves. In S. M. Scheiner and J. Gurevitch, editors, *Design and Analysis of Ecological Experiments*. Oxford University Press, 2001.
88. P. Kareiva and U. Wennergren. Connecting landscape patterns to ecosystem and population processes. *Nature*, 373:299–302, 1995.
89. C. K. Kelly and M. G. Bowler. Coexistence and relative abundance in forest trees. *Nature*, 417:437–440, 2002.
90. B. E. Kendall, J. Prendergast, and O. N. Bjornstad. The macroecology of population dynamics: Taxonomic and biogeographic patterns in population cycles. *Ecology Letters*, 1:160–164, 1998.
91. W. O. Kermack and W. G. McCormick. A contribution to the mathematical theory of epidemics. *Proceedings of the Royal Society, Series A*, 115:700–721, 1927.
92. C. J. Keylock. Simpson diversity and the shannon-wiener index as special cases of a generalized entropy. *Oikos*, 109:203–207, 2005.
93. S. E. Kingsland. *Modeling Nature*. University of Chicago Press, Chicago, 1985.
94. C.J. Krebs, S. Boutin, R. Boonstra, A.R.E. Sinclair, J.N.M. Smith, M.R.T. Dale, K. Martin, and R. Turkington. Impact of food and predation on the snowshoe hare cycle. *Science*, 269:1112–1115, 1995.
95. R. Lande. Extinction thresholds in demographic models of territorial populations. *American Naturalist*, 130:624–635, 1987.
96. R. Lande. Demographic models of the northern spotted owl (*Strix occidentalis caurina*). *Oecologia*, 75:601–607, 1988.
97. R. Lande. Statistics and partitioning of species diversity, and similarity among multiple communities. *Oikos*, 76:5–13, 1996.
98. R. Lande, P. J. DeVries, and T. R. Walla. When species accumulation curves intersect: Implications for ranking diversity using small samples. *Oikos*, 89:601–605, 2000.
99. A. M. Latimer, J. A. Silander, and R. M. Cowling. Neutral ecological theory reveals isolation and rapid speciation in a biodiversity hot spot. *Science*, 309:1722–1725, 2005.
100. L. P. Lefkovitch. The study of population growth in organisms grouped by stages. *Biometrics*, 21:1–18, 1965.
101. C.L. Lehman and D. Tilman. Competition in spatial habitats. In D. Tilman and P. Kareiva, editors, *Spatial Ecology: The Role of Space in Population Dynamics and Interspecific Interactions*, pages 185–203. Princeton University Press, Princeton, NJ, 1997.
102. M. A. Leibold, M. Holyoak, N. Mouquet, P. Amarasekare, J. M. Chase, M. F. Hoopes, R. D. Holt, J. B. Shurin, R. Law, D. Tilman, M. Loreau, and A. Gonzalez. The metacommunity concept: A framework for multi-scale community ecology. *Ecology Letters*, 7:601–613, 2004.

103. M. A. Leibold and M. A. McPeek. Coexistence of the niche and neutral perspectives in community ecology. *Ecology*, 87:1399–1410, 2006.
104. M.A. Leibold. A graphical model of keystone predators in food webs: trophic regulation of abundance, incidence, and diversity patterns in communities. *The American Naturalist*, 147:784–812, 1996.
105. Jr. Leigh, E. G. *Tropical Forest Ecology: A View from Barro Colorado Island*. Oxford University Press, Oxford, UK, 1999.
106. F. Leisch. Sweave: Dynamic generation of statistical reports using literate data analysis. In Wolfgang Härdle and Bernd Rönz, editors, *Compstat 2002 — Proceedings in Computational Statistics*, pages 575–580. Physica Verlag, Heidelberg, 2002.
107. P. H. Leslie. On the use of matrices in certain population mathematics. *Biometrika*, 35:183–212, 1945.
108. S. A. Levin, J. E. Cohen, and A. Hastings. Dispersal strategies in a patchy environment. *Theoretical Population Biology*, 26:165–191, 1984.
109. R. Levins. The strategy of model building in population biology. *American Scientist*, 54:421–431, 1966.
110. R. Levins. Some demographic and genetic consequences of environmental heterogeneity for biological control. *Bulletin of the Entomological Society of America*, 15:237–240, 1969.
111. R. Levins and D. Culver. Regional coexistence of species and competition between rare species. *Proceeding of the National Academy of Sciences, U.S.A.*, 68:1246–1248, 1971.
112. R.C. Lewontin. The meaning of stability. In *Diversity and Stability in Ecological Systems*, volume 22 of *Brookhaven Symposia in Biology*, pages 13–24, Upton, NY, 1969. Brookhaven National Laboratory.
113. R. Lincoln, G. Boxshall, and P. Clark. *A Dictionary of Ecology, Evolution and Systematics*. Cambridge University Press, Cambridge UK, 2nd edition, 1998.
114. R. Lindborg and O. Eriksson. Historical landscape connectivity affects prresent plant species diversity. *Ecology*, 85:1840–1845, 2004.
115. Z. T. Long and I. Karel. Resource specialization determines whether history influences community structure. *Oikos*, 96:62–69, 2002.
116. M. Loreau, N. Mouquet, and R. D. Holt. Meta-ecosystems: A theoretical framework for a spatial ecosystem ecology. *Ecology Letters*, 6:673–679, 2003.
117. A. J. Lotka. *Elements of Mathematical Biology*. Dover Publications, Inc., Mineola, NY, 1956.
118. R. H. MacArthur. On the relative abundance of bird species. *Proceedings of the National Academy of Sciences of the United States of America*, 43:293–295, 1957.
119. R. H. MacArthur. Some generalized theorems of natural selection. *Proceedings of the National Academy of Sciences, USA*, 48:1893–1897, 1962.
120. R. H. MacArthur. *Geographical Ecology: Patterns in the Distribution of Species*. Harper & Row, New York, 1972.
121. R. H. MacArthur and E. O. Wilson. An equilibrium theory of insular zoogeography. *Evolution*, 17:373–387, 1963.
122. R. H. MacArthur and E. O. Wilson. *The Theory of Island Biogeography*. Monographs in Population Biology. Princeton University Press, Princeton, 1967.
123. R.H. MacArthur and E.R. Pianka. On optimal use of a patchy environment. *The American Naturalist*, 100:603–609, 1966.
124. A. E. Magurran. *Measuring Biological Diversity*. Princeton University Press, 2004.

125. T. R. Malthus. *An Essay on the Principle of Population*. J. Johnson, London, 1798.
126. B.F.J. Manly. *Randomization, Bootstrap and Monte Carlo Methods in Biology*. Chapman and Hall, London, 1997.
127. R. M. May. *Stability and Complexity in Model Ecosystems*, volume 6 of *Monographs in Population Biology*. Princeton University Press, Princeton, NJ, 1973.
128. R. M. May. Biological populations with nonoverlapping generation: stable points, stable cycles, and chaos. *Science*, 186:645–647, 1974.
129. R. M. May. *Ecology and Evolution of Communities*, chapter Patterns of species abundance and diversity, pages 81–120. Harvard University Press, Cambridge, MA, 1975.
130. R. M. May. Thresholds and multiple breakpoints in ecosystems with a multiplicity of states. *Nature*, 269:471–477, 1977.
131. R. M. May. Host-parasitoid systems in patchy environments: A phenomenological model. *Journal of Animal Ecology*, 47:833–844, 1978.
132. R. M. May. *Stability and Complexity in Model Ecosystems*. Princeton Landmarks in Biology. Princeton University Press, 2001.
133. R. M. May. Network structure and the biology of populations. *Trends in Ecology & Evolution*, 21:394–399, 2006.
134. R.M. May. Will a large complex system be stable? *Nature*, 238:413–414, 1972.
135. R.M. May. *Theoretical Ecology: Principles and Applications*. Blackwell Scientific Publications, Oxford, 1976.
136. H. McCallum, N. Barlow, and J. Hone. How should pathogen transission be modeled? *Trends in Ecology and Evolution*, 16:295–300, 2001.
137. K. McCann. Density-dependent coexistence in fish communities. *Ecology*, 79:2957–2967, 1998.
138. K. McCann and A. Hastings. Re-evaluating the omnivory - stability relationship in food webs. *Proceedings of the Royal Society of London Series B-Biological Sciences*, 264:1249–1254, 1997.
139. A. McKane, D. Alonso, and R. V. Sole. Mean-field stochastic theory for species-rich assembled communities. *Physical Review E*, 62:8466–8484, 2000.
140. M. A. McPeek and S. Kalisz. Population sampling and bootstrapping in complex designs: Demographic analysis. In S.M. Scheiner and J. Gurevitch, editors, *Design and Analysis of Ecological Experiments*, pages 232–252. Chapman & Hall, New York, 1993.
141. F. Messier. Ungulate population models with predation - a case study with the north american moose. *Ecology*, 75:478–488, 1994.
142. P. J. Morin. *Community Ecology*. Blackwell Science, Inc., Malden, MA, 1999.
143. P. J. Morin. Productivity, intraguild predation, and population dynamics in experimental food webs. *Ecology*, 80:752–760, 1999.
144. H. Morlon, G. Chuyong, R. Condit, S.P. Hubbell, D. Kenfack, D. Thomas, R. Valencia, and J. L. Green. A general framework for the distance-decay of similarity in ecological communities. *Ecology Letters*, 11:904–917, 2008.
145. I. Motomura. A statistical treatment of associations [in Japanese]. *Japanese Journal of Zoology*, 44:379–383, 1932.
146. S. Nee and R.M. May. Dynamics of metapopulations: habitat destruction and competitive coexistence. *Journal of Animal Ecology*, 61:37–40, 1992.
147. M. Nei. *Molecular Evolutionary Genetics*. Columbia University Press, New York, 1987.
148. M.G. Neubert and H. Caswell. Alternatives to resilience for measuring the responses of ecological systems to perturbations. *Ecology*, 78:653–665, 1997.

149. A.J. Nicholson and V.A. Bailey. The balance of animal populations - Part I. *Proceedings of the Zoological Society of London*, pages 551–598, 1935.
150. I. Noy-Meir. Stability of grazing systems: An application of predator-prey graphs. *The Journal of Ecology*, 63:459–481, 1975.
151. J. Oksanen, R. Kindt, P. Legendre, R. O'Hara, G. L. Simpson, P. Solymos, M. H. H. Stevens, and H. Wagner. *vegan: Community Ecology Package*, 2008. R package version 1.15-1.
152. S. W. Pacala and M. Rees. Models suggesting field experiments to test two hypotheses explaining successional diversity. *The American Naturalist*, 152:729–737, 1998.
153. S.W. Pacala. Dynamics of plant communities. In M.J. Crawley, editor, *Plant Ecology*, pages 532–555. Blackwel Science, Ltd., Oxford, UK, 1997.
154. S.W. Pacala, M.P. Hassell, and R.M. May. Host-parasitoid associations in patchy environments. *Nature*, 344:150–153, 1990.
155. O.L. Petchey. Environmental colour affects the dynamics of single species populations. *Proceedings of the Royal Society of London B*, 267:747–754, 2001.
156. R. H. Peters. Some general problems in ecology illustrated by food web theory. *Ecology*, 69:1673–1676, 1988.
157. T. Petzoldt. R as a simulation platform in ecological modelling. *R News*, 3:8–16, 2003.
158. T. Petzoldt and K. Rinke. simecol: An object-oriented framework for ecological modeling in r. *Journal of Statistical Software*, 22:1–31, 2007.
159. S.L. Pimm and J.H. Lawton. Number of trophic levels in ecological communities. *Nature*, 268:329 331, 1977.
160. S.L. Pimm and J.H. Lawton. On feeding on more than one trophic level. *Nature*, 275:542–544, 1978.
161. J. Pinheiro and D. Bates. *Mixed-effects Models in S and S-PLUS*. Springer, New York, 2000.
162. W. Platt and I. Weis. Resource partitioning and competition within a guild of fugitive prairie plants. *The American Naturalist*, 111:479–513, 1977.
163. J. B. Plotkin, M. D. Potts, D. W. Yu, S. Bunyavejchewin, R. Condit, R. Foster, S. Hubbell, J. LaFrankie, N. Manokaran, L.H. Seng, R. Sukumar, M. A. Nowak, and P. S. Ashton. Predicting species diversity in tropical forests. *Proceedings of the National Academy of Sciences, U.S.A.*, 97(20):10850–10854, 2000.
164. J.B. Plotkin and H.C. Muller-Landau. Sampling the species composition of a landscape. *Ecology*, 83:3344–3356, 2002.
165. G. A. Polis, C. A. Myers, and R. D. Holt. The ecology and evolution of intraguild predation: potential competitors that eat each other. *Annual Review of Ecology and Systematics*, 20:297–330, 1989.
166. G.A. Polis. Complex trophic interactions in deserts: an empirical critique of food web ecology. *The American Naturalist*, 138:123–155, 1991.
167. D. M. Post. The long and short of food-chain length. *Trends in Ecology & Evolution*, 17:269–277, 2002.
168. F. W. Preston. The commonness, and rarity, of species. *Ecology*, 29:254–283, 1948.
169. F. W. Preston. Time and space and variation of variation. *Ecology*, 41:611–627, 1960.
170. F. W. Preston. The canonical distribution of commonness and rarity: part I. *Ecology*, 43:185–215., 1962.
171. S. Pueyo, F. He, and T. Zillio. The maximum entropy formalism and the idiosyncratic theory of biodiversity. *Ecology Letters*, 10:1017–1028, 2007.

172. H. R. Pulliam. Sources, sinks, and population regulation. *The American Naturalist*, 132:652–661, 1988.
173. R Development Core Team. *R: A language and environment for statistical computing*. R Foundation for Statistical Computing, Vienna, Austria, version 2.8.1 edition, 2008.
174. D. Rabinowitz and J. K. Rapp. Dispersal abilities of seven sparse and common grasses from a Missouri prairie. *American Journal of Botany*, 68:616–624, 1981.
175. H. L. Reynolds and S. W. Pacala. An analytical treatment of root-to-shoot ratio and plant competition for soil nutrient and light. *American Naturalist*, 141:51–70, 1993.
176. J. F. Riebesell. Paradox of enrichment in competitive systems. *Ecology*, 55:183–187, 1974.
177. M. L. Rosenzweig. Why the prey curve has a hump. *American Naturalist*, 103:81–87, 1969.
178. M. L. Rosenzweig. *Species Diversity in Space and Time*. Cambridge University Press, Cambridge, 1995.
179. M.L. Rosenzweig. Paradox of enrichment: destabilization of exploitation ecosystems in ecological time. *Science*, 171:385–387, 1971.
180. M.L. Rosenzweig and R.H. MacArthur. Graphical representation and stability conditions of predator-prey interactions. *American Naturalist*, 97:209–223, 1963.
181. J. Roughgarden. *Primer of Ecological Theory*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1998.
182. J. Roughgarden and F. Smith. Why fisheries collapse and what to do about it. *Proceedings of the National Academy of Sciences, U.S.A.*, 93:5078–5083, May 1996.
183. P. F. Sale. The maintenance of high diversity in coral reef fish communities. *The American Naturalist*, 111:337–359, 1977.
184. M. Scheffer, S. Szabo, A. Gragnani, E. H. van Nes, S. Rinaldi, N. Kautsky, J. Norberg, R. M. M. Roijackers, and R. J. M. Franken. Floating plant dominance as a stable state. *Proceeding of the National Academy of Sciences, U.S.A.*, 100:4040–4045, 2003.
185. O. J. Schmitz. Perturbation and abrupt shift in trophic control of biodiversity and productivity. *Ecology Letters*, 7:403–409, 2004.
186. A. Schröder, L. Persson, and A. M. De Roos. Direct experimental evidence for alternative stable states: A review. *Oikos*, 110:3–19, 2005.
187. A. Shmida and S. P. Ellner. Coexistance of plant species with similar niches. *Vegetatio*, 58:29–55, 1984.
188. R. M. Sibly, D. Barker, M. C. Denham, J. Hone, and M. Pagel. On the regulation of populations of mammals, birds, fish, and insects. *Science*, 309:607–610, 2005.
189. J. G. Skellam. Random dispersal in theoretical populations. *Biometrika*, 38:196–218, 1951.
190. K. Soetaert, T. Petzoldt, and R. W. Setzer. *deSolve: General solvers for ordinary differential equations (ODE) and for differential algebraic equations (DAE)*. R package version 1.2-3.
191. N. C. Stenseth, W. Falck, O. N. Bjornstad, and C. J. Krebs. Population regulation in snowshoe hare and Canadian lynx: Asymmetric food web configurations between hare and lynx. *Proceedings of the National Academy of Sciences of the United States of America*, 94:5147–5152, 1997.
192. P. A. Stephens and W. J. Sutherland. Consequences of the allee effect for behaviour, ecology and conservation. *Trends in Ecology & Evolution*, 14:401–405, 1999.

193. R. W. Sterner and J. J. Elser. *Ecological Stoichiometry: The Biology of Elements From Molecules to the Biosphere*. Princeton University Press, Princeton, N.J., 2002.
194. M. H. H. Stevens, O. L. Petchey, and P. E. Smouse. Stochastic relations between species richness and the variability of species composition. *Oikos*, 103:479–488, 2003.
195. M. H. H. Stevens and C. E. Steiner. Effects of predation and nutrient enrichment on a food web with edible and inedible prey. *Freshwater Biology*, 51:666–671, 2006.
196. G. Sugihara. Minimal community structure: an explanation of species abundance patterns. *The American Naturalist*, 116:770–787, 1980.
197. K. S. Summerville and T. O. Crist. Determinants of lepidopteran community composition and species diversity in eastern deciduous forests: roles of season, eco- region and patch size. *Oikos*, 100:134–148, 2003.
198. K. S. Summerville and T. O. Crist. Contrasting effects of habitat quantity and quality on moth communities in fragmented landscapes. *Ecography*, 27:3–12, 2004.
199. R. M. Thompson, M. Hemberg, B. M. Starzomski, and J. B. Shurin. Trophic levels and trophic tangles: The prevalence of omnivory in real food webs. *Ecology*, 88:612–617, 2007.
200. D. Tilman. *Resource Competition and Community Structure*. Monographs in Population Biology. Princeton University Press, Princeton, NJ, 1982.
201. D. Tilman. *Plant Strategies and the dynamics and structure of plant communities*, volume 26 of *Monographs in Population Biology*. Princeton University Press, Princeton, NJ, 1988.
202. D. Tilman. Competition and biodiversity in spatially structured habitats. *Ecology*, 75:2–16, 1994.
203. D. Tilman, R. M. May, C. L. Lehman, and M. A. Nowak. Habitat destruction and the extinction debt. *Nature*, 371:65–66, 1994.
204. David Tilman. Resource competition and plant traits: A response to Craine et al. 2005. *Journal of Ecology*, 95:231–234, 2007.
205. M. Tokeshi. Niche apportionment or random assortment: species abundance patterns revisited. *Journal of Animal Ecology*, 59:1129–1146, 1990.
206. M. Tokeshi. *Species Coexistence: Ecological and Evolutionary Perspectives*. Blackwell Science, Ltd, Oxford, UK, 1999.
207. J. Vandermeer. Omnivory and the stability of food webs. *Journal of Theoretical Biology*, 238:497–504, 2006.
208. J. Vandermeer. Oscillating populations and biodiversity maintenance. *Bioscience*, 56:967–975, 2006.
209. J. Vandermeer, I. de la Cerda, I. G.and Perfecto, D. Boucher, J. Ruiz, and A. Kaufmann. Multiple basins of attraction in a tropical forest: Evidence for nonequilibrium community structure. *Ecology*, 85:575–579, 2004.
210. J. Vandermeer and P. Yodzis. Basin boundary collision as a model of discontinuous change in ecosystems. *Ecology*, 80:1817–1827, 1999.
211. M. J. Vanni, W. H. Renwick, J. L. Headworth, J. D. Auch, and M. H. Schaus. Dissolved and particulate nutrient flux from three adjacent agricultural watersheds: A five-year study. *Biogeochemistry*, 54:85–114, 2001.
212. J.A. Veech and T.O. Crist. Partition: software for hierarchical additive partitioning of species diversity, version 2.0. http:// www.users.muohio.edu/ cristto/ partition.htm, 2007.
213. M. Vellend and M. A. Geber. Connections between species diversity and genetic diversity. *Ecology Letters*, 8:767–781, 2005.

214. I. Volkov, J. R. Banavar, F. L. He, S. P. Hubbell, and A. Maritan. Density dependence explains tree species abundance and diversity in tropical forests. *Nature*, 438:658–661, 2005.
215. I. Volkov, J. R. Banavar, S. P. Hubbell, and A. Maritan. Neutral theory and relative species abundance in ecology. *Nature*, 424:1035–1037, 2003.
216. I. Volkov, J. R. Banavar, S. P. Hubbell, and A. Maritan. Patterns of relative species abundance in rainforests and coral reefs. *Nature*, 450:45–49, 2007.
217. R. R. Warner and P. L. Chesson. Coexistence mediated by recruitment fluctuations: a field guide to the storage effect. *The American Naturalist*, 125:769–787, 1985.
218. D. Wedin and D. Tilman. Competition among grasses along a nitrogen gradient: initial conditions and mechanisms of competition. *Ecological Monographs*, 63:199–229, 1993.
219. R. H. Whittaker. Vegetation of the siskiyou mountains, oregon and california. *Ecological Monographs*, 30:279–338, 1960.
220. J. A. Wiens. Spatial scaling in ecology. *Functional Ecology*, 3:385–397, 1989.
221. R. J. Williams and N. D. Martinez. Simple rules yield complex food webs. *Nature*, 404:180–182, 2000.
222. S. D. Wilson and D. Tilman. Competitive responses of 8 old-field plant-species in 4 environments. *Ecology*, 76:1169–1180, 1995.
223. J. T. Wootton. Field parameterization and experimental test of the neutral theory of biodiversity. *Nature*, 433:309–312, 2005.

Index