8 Likelihood Examples

This chapter combines all the methods we've considered so far to carry out more complete analyses of some of the example data sets, specifically the data of Vonesh and Bolker (2005) on tadpole predation, Wilson (2004) on goby survival, and Duncan and Duncan (2000) on seed removal.

8.1 Tadpole Predation

8.1.1 Introduction

The goal of Vonesh and Bolker's (2005) tadpole predation study was to quantify the effects of prey size and density on predation rate, and to use the results along with data on growth rates to understand the trade-offs between growth and survival. The response variable in all of the data we will consider here is the number of tadpoles killed by a given number or density of predators in a specified amount of time; the covariates are changing (initial) number of tadpoles (which gives rise to a *functional response* curve) and the size of tadpoles (estimating the presence of a "size refuge").

The binomial distribution is an obvious choice as a stochastic model for predation data, because the data are a discrete sample with a fixed upper limit. The challenge for the frog predation data is to decide on deterministic models that adequately describe the changes in predation probability with tadpole size and density.

8.1.2 Fitting the Size-Predation Curve

Vonesh and Bolker (2005) used the function

$$\gamma(S) = \frac{e^{\epsilon(\phi - S)}}{1 + e^{\beta\epsilon(\phi - S)}} \tag{8.1.1}$$

to represent the dependence of predation probability $\gamma(S)$ on prey size S (Figure 8.1).

The location parameter ϕ represents a baseline prey size at which 50% of tadpoles are eaten; ϵ is the rate of change of mortality with size, controlling the steepness of the curve; and β determines the asymmetry of the curve—the extent to which prey

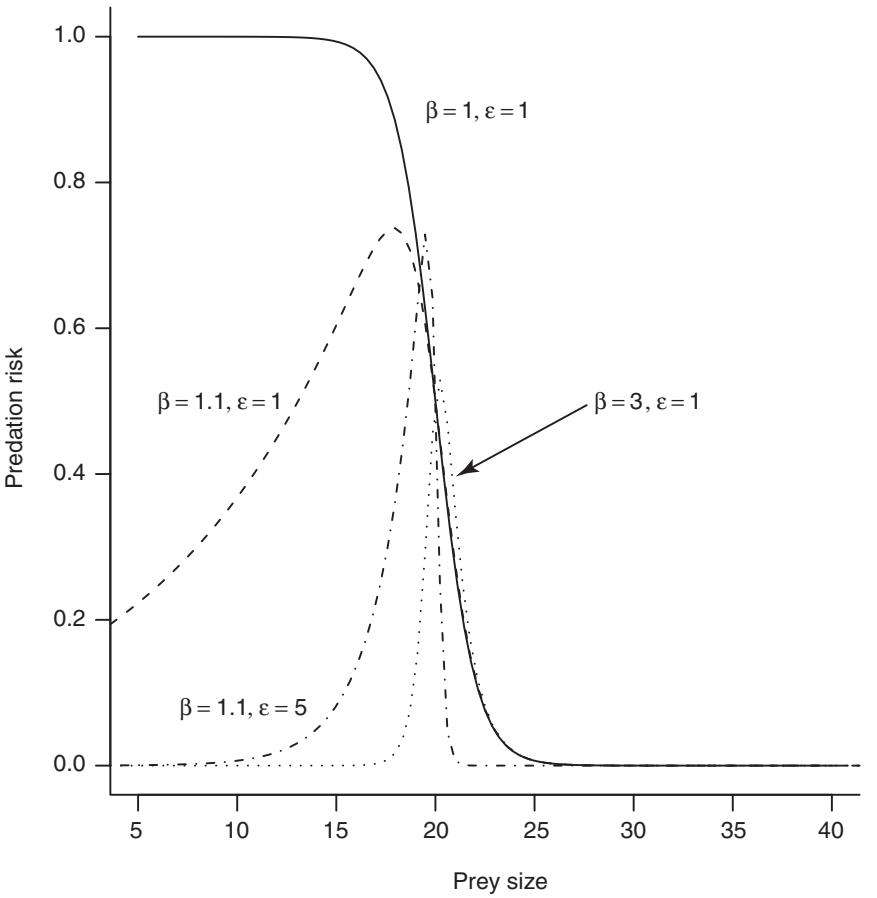

Figure 8.1 Modified logistic function from Vonesh and Bolker (2005) (eq. 8.1.1). Location parameter $\phi = 20$ for all curves.

escape predation at both small and large sizes. If $\beta = 1$, then (8.1.1) describes a logistic predation function that decreases (if $\epsilon > 0$) or increases (if $\epsilon < 0$) with size.

Some slightly tedious calculus establishes that the most vulnerable size is $\hat{S} = \phi + \log(\beta - 1)/(\epsilon\beta)$, which gives a predation probability

$$(\beta - 1)^{(-1/\beta)}/(1 + 1/(\beta - 1)).$$

The peak predation probability depends only on β. If $\beta < 1$, then the function is monotonically decreasing, with no peak. (To find $\hat{S}$, solve $d\gamma/dS = 0$ for S, using the quotient and chain rules to calculate the derivative, and remembering that in this case you only need to find where the numerator is zero. Then plug $\hat{S}$ back into $\gamma(S)$ to find the predation probability.)

A more traditional function to describe a humped (unimodal) dependence of predation on size is the *generalized Ricker function* (Persson et al., 1998),

$$y = b\left(\frac{S}{a}\exp\left(1 - \frac{S}{a}\right)\right)^{\alpha}. \tag{8.1.2}$$

This function is basically a reparameterization of the Ricker function ($y = axe^{-bx}$) with an added power parameter α that can broaden or narrow the peak. When $\alpha = 1$, the generalized Ricker reduces to the standard Ricker function.

A third possibility is another modification of the Ricker, which I will call the truncated Ricker: this function shifts the Ricker's origin away from zero by a distance t, and sets the function to zero below t so that it doesn't become negative:

$$y = \begin{cases} 0 & \text{if } S < t \\ b\left(\frac{S-t}{a} e^{1-\left(\frac{S-t}{a}\right)}\right) & \text{if } S \geq t. \end{cases} \tag{8.1.3}$$

All of these functions are phenomenological rather than mechanistic: while ecologists have ideas about the mechanisms leading to low predation at small size (poor detectability and being of little value to the predator) and large size (escape speed and predator gape limitation), they don't know enough about these mechanisms to guess at an appropriate functional form.

Load the data and `attach` it (don't forget to detach it later):

```
> data(ReedfrogSizepred)
> attach(ReedfrogSizepred)
```

Define the functions (`modlogist` for the modified logistic, `powricker` and `tricker` for the generalized (power) and truncated Ricker):

```
> modlogist = function(x, eps, beta, phi) {
+     exp(eps * (phi - x))/(1 + exp(beta * eps * (phi -
+         x)))
+ }
> powricker = function(x, a, b, alpha) {
+     b * (x/a * exp(1 - x/a))^alpha
+ }
> tricker = function(x, a, b, t, min = 1e-04) {
+     ifelse(x < t, min, b * ((x - t)/a * exp(1 - (x -
+         t)/a)))
+ }
```

Set up negative log-likelihood functions for each model, including one for the modified logistic that uses a beta-binomial distribution (p. 126) of numbers killed (`NLL.modlogist.bb`, with overdispersion parameter θ) instead of a binomial in order to account for possible overdispersion.*

```
> NLL.modlogist = function(eps, beta, phi) {
+     p.pred = modlogist(TBL, eps, beta, phi)
```

* A quick and dirty way to check for overdispersion is to compute the *residual deviance*, which is $-2\times$ the log-likelihood for the most complex model you fit. For sufficiently large data sets the scaled residual deviance should be χ^2 distributed with degrees of freedom equal to the residual degrees of freedom. However, Venables and Ripley (2002, p. 208) warn that this estimate can be misleading for moderate-size data sets (e.g., expected Poisson means less than 5 or expected number of successes in a binomial trial (Np) less than 10). For this data set, the quick and dirty approach suggests that there is overdispersion, but the likelihood fit below shows more accurately that there isn't.

```
+     -sum(dbinom(Kill, size = 10, prob = p.pred,
+         log = TRUE))
+ }
> NLL.modlogist.bb = function(eps, beta, phi, theta) {
+     p.pred = modlogist(TBL, eps, beta, phi)
+     -sum(dbetabinom(Kill, size = 10, prob = p.pred,
+         theta = theta, log = TRUE))
+ }
> NLL.powricker = function(a, b, alpha) {
+     p.pred = powricker(TBL, a, b, alpha)
+     -sum(dbinom(Kill, size = 10, prob = p.pred,
+         log = TRUE))
+ }
> NLL.tricker = function(a, b, t) {
+     p.pred = tricker(TBL, a, b, t)
+     -sum(dbinom(Kill, size = 10, prob = p.pred,
+         log = TRUE))
+ }
```

Eyeballing the data (Figure 8.2) gives approximate starting parameters for the modified logistic of $\{\phi = 15, \beta = 1.1, \epsilon = 5\}$ (compare Figure 8.1, and use ϕ to shift the peak to approximately $S = 15$). I'll start the beta-binomial version at the best-fit parameters for the binomial model and add $\theta = 1000$ (representing very little overdispersion—the beta-binomial becomes binomial as $\theta \to \infty$), setting the `parscale` control option to let R know that we expect this parameter to be larger than the others. (In an initial exploration with worse starting parameter guesses, I also played around with options like `method="Nelder-Mead"` and setting the `maxit` control parameter larger in order to get the optimization to work.)

```
> FSP.modlogist = mle2(NLL.modlogist, start = list(eps = 5,
+     beta = 1.1, phi = 15))
> FSP.modlogist.bb = mle2(NLL.modlogist.bb, start = as.list
+     (c(coef(FSP.modlogist), list(theta = 1000))),
+     control = list(parscale = c(1, 1, 1, 1000)))
```

The beta-binomial fit estimates $\theta = 6865$, evidence that the beta-binomial model is not really necessary; the decrease in negative log-likelihood is only 0.003.

We hardly need to run the likelihood ratio test (`anova(FSP.modlogist,FSP.modlogist.bb)`) or the AIC calculation (`AICtab(FSP.modlogist,FSP.modlogist.bb)`). Even dividing the p-value for the Likelihood Ratio test by 2 to account for the fact that the null hypothesis is on the boundary (i.e., the beta-binomial model reduces to the binomial model when $\theta \to \infty$) makes no difference to the conclusions.

If we try to get confidence limits on θ, however, we run into trouble:

```
> confint(FSP.modlogist.bb, which = "theta")

Profiling has found a better solution, so original
fit had not converged:
New minimum= 12.13806
```

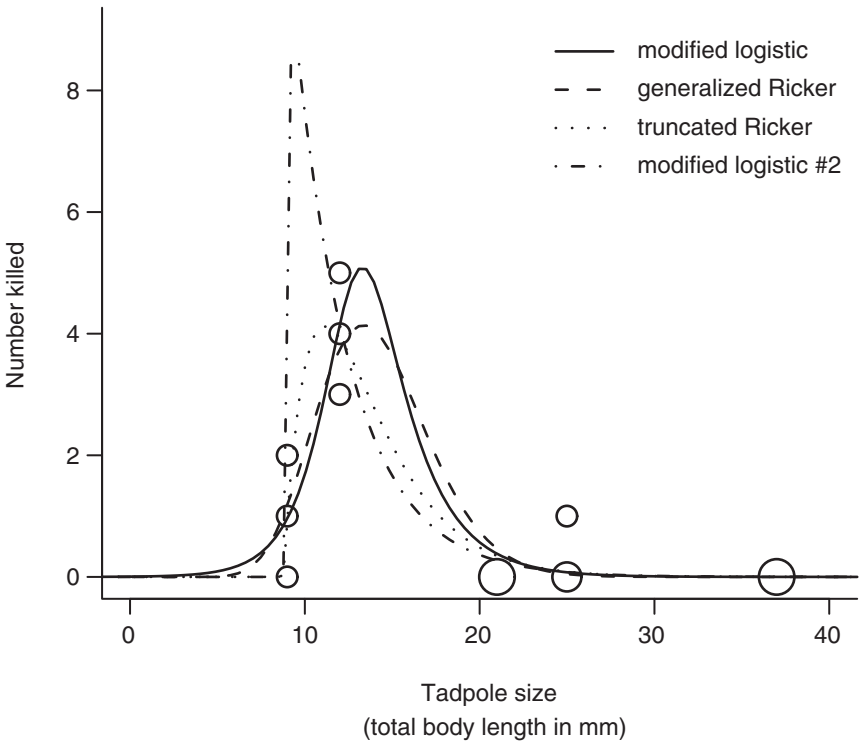

Figure 8.2 Size-predation relationship for *Hyperolius spinigularis* tadpoles: modified logistic, generalized and truncated Ricker fits.

```
Parameter values:
      eps        beta.beta         phi           theta
   0.3577257     8.9873023     9.7457033     3405.1429647
Error in onestep(step) : try restarting fit from values above
```

Refitting the parameters from this new starting point (using `modlogist` instead of `modlogist.bb`, and extending the maximum number of iterations):

```
> FSP.modlogist2 = mle2(NLL.modlogist, start = list
+     (eps = 0.357, beta = 8.99, phi = 9.75),
+     control = list(maxit = 1000))
```

The parameters of this fit are quite different:

```
> rbind(coef(FSP.modlogist), coef(FSP.modlogist2))
```

```
              eps             beta          phi
[1,] 0.4042309      2.470003     12.908932
[2,] 0.3045399     67.080841      9.109064
```

and the negative log-likelihood is slightly lower (11.77 vs. 12.15). You can use `plot(calcslice(FSP.modlogist,FSP.modlogist2))` to calculate and plot the

TABLE 8.1
Comparison of fits for tadpole size-dependent predation models

	AIC	df	Weight
Modified logistic (fit 2)	29.5	3	0.350
Truncated Ricker	29.9	3	0.297
Modified logistic (fit 1)	30.3	3	0.238
Generalized Ricker	31.8	3	0.115

negative log-likelihoods along a "slice" through parameter space, showing that the two different fits probably do represent distinct local minima (Figure 7.10).

However, despite fitting the data a little better the fit seems unrealistic, spiking up abruptly to a high predation rate and then dropping exponentially (Figure 8.2).

Fitting the generalized and truncated Ricker models:

```
> FSP.powricker = mle2(NLL.powricker, start = list(a = 0.4,
+     b = 0.3, alpha = 1))
> FSP.tricker = mle2(NLL.tricker, start = list(a = 0.4,
+     b = 0.3, t = 8))
```

The confidence limits on α for the generalized Ricker (`confint(FSP.powricker, parm="alpha")`) are $\{7.18, 31.69\}$—the standard Ricker ($\alpha = 1$) is clearly not competitive.

Calculating AIC values with `AICtab`, we get the results presented in Table 8.1. None of the models is nested (indeed, all have the same number of parameters), and all the fits are (almost) within 2 log-likelihood units of each other. Burnham and Anderson would recommend using the weighted predictions of all the models in subsequent analyses, but in this case (where we are just trying to gain qualitative insights into life-history trade-offs) this extra complication feels unnecessary. In this case, I would be willing to override the narrow definition of "best fit" and discard the first two models because I don't believe that predation risk is going to increase sharply as tadpoles grow bigger than 9 mm, as suggested by the truncated Ricker or by the second fit to the modified logistic. I might even choose the generalized Ricker, the worst-fitting model, over the first fit of the modified logistic, because the generalized Ricker is better established in the literature. The lesson here is that the sparser the data, the more you have to use your judgment in selecting a model—whether or not you are explicitly Bayesian.

8.1.3 Fitting the Functional Response Curve

The other data set we will examine from Vonesh and Bolker (2005) is the functional response experiment, which varied the density of tadpoles (with total body length ≈ 12.8 mm). As many as 67% (10/15) of the tadpoles in an experiment were eaten, suggesting that we should allow for the effect of depletion over the course of the experiment. The standard model for saturating functional responses is the Holling type II response, $N = aPTN_0/(1 + ahN_0)$, where N is the number eaten, N_0 is the

starting number/density, a and h are baseline attack rate and handling time, P is the predator number or density, and T is the total exposure time.* The Rogers random-predator equation, which allows for depletion, is

$$N = N_0(1 - e^{a(Nh-PT)}). \tag{8.1.4}$$

The Rogers random-predator equation (8.1.4) contains N on both the left- and right-hand sides of the equation; traditionally, one has had to use iterative numerical methods to compute the function (Vonesh and Bolker, 2005). However, the *Lambert W* function (Corless et al., 1996), which gives the solution to the equation $W(x)e^{W(x)} = x$, can be used to compute the Rogers equation efficiently: in terms of the Lambert W the Rogers equation is

$$N = N_0 - \frac{W(ahN_0e^{-a(PT-hN_0)})}{ah}. \tag{8.1.5}$$

Implement this equation (using the `lambertW` function in the `emdbook` package) in R, as well as the Holling type II function for comparison:

```
> rogers.pred = function(N0, a, h, P, T) {
+     N0 - lambertW(a * h * N0 * exp(-a * (P * T -
+         h * N0)))/(a * h)
+ }
> holling2.pred = function(N0, a, h, P, T) {
+     a * N0 * P * T/(1 + a * h * N0)
+ }
```

Load and attach the data (detaching is up to you):

```
> data(ReedfrogFuncresp)
> attach(ReedfrogFuncresp)
```

Write the likelihood functions:

```
> NLL.rogers = function(a, h, T, P) {
+     if (a < 0 || h < 0)
+         return(NA)
+     prop.exp = rogers.pred(Initial, a, h, P, T)/Initial
+     -sum(dbinom(Killed, prob = prop.exp, size = Initial,
+         log = TRUE))
+ }
> NLL.holling2 = function(a, h, P = 1, T = 1) {
+     -sum(dbinom(Killed, prob = a * T * P/(1 + a *
+         h * Initial), size = Initial, log = TRUE))
+ }
```

* P and T are usually ignored in the Holling equation, giving the function units of "number eaten per predator per unit time," but we include them here for consistency with the Rogers equation.

In the Rogers likelihood function I constrained the range of the function by simply returning `NA` if $a < 0$ or $h < 0$, rather than using constrained optimization; if you are not using `L-BFGS-B`, this shortcut sometimes works.

What about initial values? Eyeballing the data (Figure 8.3), we see the initial slope of the functional response curve is about 0.5 (50% of tadpoles are killed at low densities) and the asymptote is about 50. These values correspond to $aPT = 0.5$ or $a = 0.5/(PT) \approx 0.012$ and $PT/h = 50$ or $h \approx 0.84$. These values will be overestimates, but still usable, as starting points for the Rogers estimation as well:

```
> FFR.rogers = mle2(NLL.rogers, start = list(a = 0.012,
+     h = 0.84), data = list(T = 14, P = 3))
> FFR.holling2 = mle2(NLL.holling2, start = list(a = 0.012,
+     h = 0.84), data = list(T = 14, P = 3))
```

Running `AICtab(FFR.rogers,FFR.holling2,weights=TRUE)` shows that the Holling type II is a marginally better fit (0.3 log-likelihood unit difference):

	AIC	df	Weight
Holling type II	97.4	2	0.536
Rogers	97.7	2	0.464

The best-fit Holling and Rogers curves are practically indistinguishable in the plot (Figure 8.3) as well: However, we strongly prefer the Rogers curve on biological grounds, because we know that predators are depleting tadpole prey significantly over the course of the experiment. The "Rogers (no depletion)" curve in Figure 8.3 shows that depletion decreases the effect of predation by about two tadpoles across the board—as much as a 40% effect at low numbers. It will be important to take depletion into account when we compare experiments with different exposure times and predator densities below.

	a	h
Rogers	0.0171	0.814
Holling type II	0.0126	0.704

Taking depletion into account leads to a 36% increase in the estimated attack rate and a 16% increase in the estimated handling time.

8.1.4 Combined Effects of Size and Density

Vonesh and Bolker (2005) combined the effects of size and density by algebraically combining the parameters of the separate size and density fits. Here, we will instead combine all the data in a single likelihood function, estimating the functional response parameter (h) and the size-dependent attack rate parameters (α, β, and ϵ) at the same time.* The only thing we need to sort out is that the experiments were run in different volumes, as well as with different numbers of predators and for different lengths of time. The functional response experiments were run in 300-L tanks ($1.2 \times 0.8 \times 0.4$ m

* It would be realistic to make the handling time vary as a function of size as well (Persson et al., 1998), but unfortunately we don't have enough data.

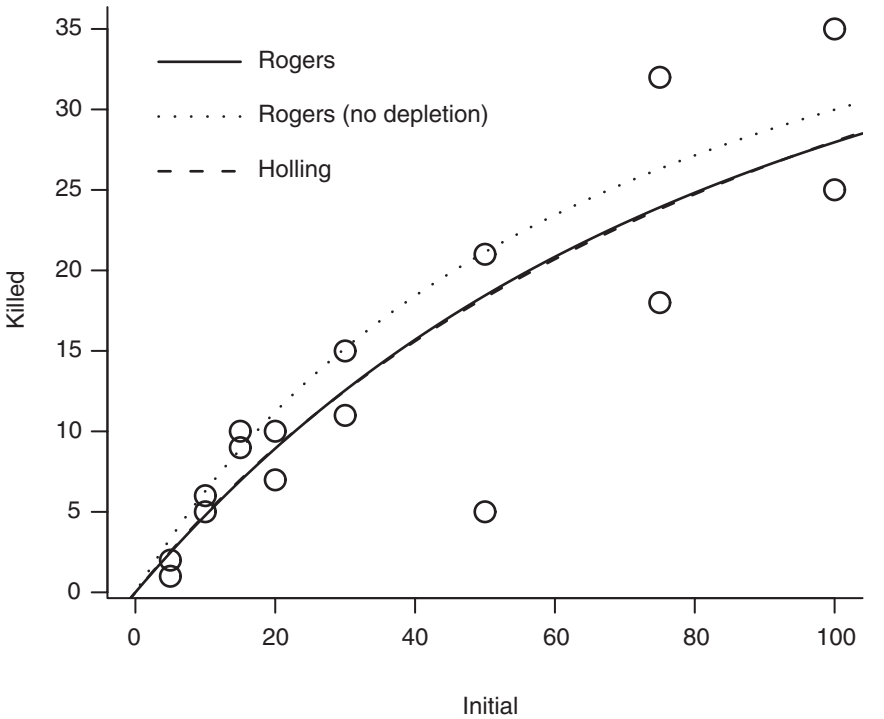

Figure 8.3 Functional response fit to frog predation data. Both Holling type II and Rogers random-predator fits are shown, but are barely distinguishable. "Rogers (no depletion)" curve plots the expected functional response from the estimated Rogers parameters in the absence of depletion.

high) filled to 220 L; the size experiments were run in 35-L plastic tubs (0.32 m in diameter) filled to 25 L. Based on the way that predators foraged, Vonesh and Bolker (2005) assumed that predation success depended on the area of the foraging arena ($1.2 \cdot 0.8 = 0.96$ m^2 vs. $\pi((0.32)/2)^2 = 0.080$ m^2) rather than its volume. To make the predation probabilities match, we have to divide the predator numbers by area.* It is convenient to collect the auxiliary parameters for each experiment (number of predators, area, exposure time, etc.) in a couple of lists:

```
> xpars.Funcresp = list(T = 14, P = 3, vol = 220,
+     area = 1.2 * 0.8, size = 12.8)
> xpars.Sizepred = list(T = 3, P = 2, vol = 25, area = pi *
+     0.16^2, initprey = 10)
```

Put together a combined data set representing the initial numbers, size, number killed, predator density, and exposure time for both experiments, using `rep` to repeat values where necessary:

```
> n.Funcresp = nrow(ReedfrogFuncresp)
> n.Sizepred = nrow(ReedfrogSizepred)
```

* But *not* the prey numbers—figuring this out reminded me of an old riddle, "If a hen and a half lays an egg and a half in a day a half, how many eggs can one hen lay in a day?"

```
> combInit = c(ReedfrogFuncresp$Initial,
+     rep(xpars.Sizepred$initprey, n.Sizepred))
> combSize = c(rep(xpars.Funcresp$size, n.Funcresp),
+     ReedfrogSizepred$TBL)
> combKilled = c(ReedfrogFuncresp$Killed,
+     ReedfrogSizepred$Kill)
> combP = rep(c(xpars.Funcresp$P/xpars.Funcresp$area,
+     xpars.Sizepred$P/xpars.Sizepred$area), c(n.Funcresp,
+     n.Sizepred))
> combT = rep(c(xpars.Funcresp$T, xpars.Sizepred$T),
+     c(n.Funcresp, n.Sizepred))
```

Write a combined function for the expected proportion eaten, computing the attack rate a from the parameters ϵ, β, and ϕ and combining it with the handling time h:

```
> prop.eaten = function(N0, S, h, P, T, eps, beta,
+     phi, minprop = .Machine$double.eps) {
+     a = modlogist(S, eps = eps, beta = beta, phi = phi)
+     N.eaten = rogers.pred(N0, a = a, h = h, P = P,
+         T = T)
+     prop = N.eaten/N0
+     prop[prop <= 0] = minprop
+     prop[prop >= 1] = 1 - minprop
+     prop
+ }
```

The value `.Machine$double.eps` is a built-in constant corresponding to the smallest difference between numeric values your computer can keep track of without rounding (it is 2.22×10^{-16} on the machine I am using). Using `minprop` to adjust values that are ≤ 0 or ≥ 1 takes care of the cases where the `rogers.pred` function returns an expected proportion eaten slightly less than zero, or exactly equal to 1 (which causes an infinite negative log-likelihood if no tadpoles are eaten); these minor errors happen because of round-off error.

A negative log-likelihood function incorporating the proportion eaten:

```
> NLL.rogerscomb = function(a, h, eps, beta, phi, T = combT,
+     P = combP) {
+     if (h < 0)
+         return(NA)
+     prob = prop.eaten(combInit, combSize, h, P, T,
+         eps, beta, phi)
+     dprob = dbinom(combKilled, prob = prob, size =
+         combInit, log = TRUE)
+     -sum(dprob)
+ }
```

Set the starting values by combining h from the Rogers fit (which has to be put inside its own `list`) with the attack rates from the size-dependence fit (which will be a slight underestimate since they don't incorporate the effects of handling time):

```
> startvals = c(list(h = coef(FFR.rogers)["h"]),
+     as.list(coef(FSP.modlogist)))
```

Finding the optimum, avoiding the alternate fit (fit 2 above) when profiling, and avoiding overflow errors is quite finicky in this case. The easiest way to avoid the alternate fit is to restrict β, but using the `L-BFGS-B` optimizer leads to lots of headaches with `NA`s being produced in the Lambert W function. I used a two-stage method—first, optimizing with `method="Nelder-Mead"` and using `confint(FPcomb,method="quad")` to get approximate confidence limits:

```
> FPcomb = mle2(NLL.rogerscomb, start = startvals,
+     method = "Nelder-Mead")
> confint(FPcomb, method = "quad")
```

Then I used slightly larger values for the upper and lower bounds to refit the model and get more precise confidence limits (`confint` must use the same optimization rules that were used in the original fit). Getting this to work took some frustrating trial and error, including incorporating debugging statements like

```
> cat(h, eps, beta, phi, "\n")
```

or

```
> if (any(!is.finite(prob))) cat("NAs:", h, eps, beta,
+     phi, "\n")
```

or

```
> if (any(!is.finite(dprob))) {
+     browser()
+ }
```

into the `NLL.rogerscomb` function to track down where the problems were occurring in order to set bounds that would prevent `NA`s. `cat` prints a list of variables to the screen in the middle of a function evaluation ("`\n`" specifies a new line), while `browser` stops the function and lets you examine the values of different variables. In the course of this exploration I also went back and incorporated the minimum and maximum bounds in `prop.eaten`, which I had initially left out.

```
> FPcomb = mle2(NLL.rogerscomb, start = startvals,
+     method = "L-BFGS-B", lower = c(0.7, 0.5, 1, 14),
+     upper = c(1.8, 2.25, 2, 20), control = list(parscale =
+         c(1, 1, 1, 10)))
> FPcomb.ci = confint(FPcomb)
```

What is the combined estimate of the proportion eaten under the conditions of the size-predation experiment (12.8 mm body length, 2 predators in an area of 0.08 m^2 for 3 days)? How well does it match the estimate based only on the size-predation experiment? (That is, does combining the data change the baseline estimate from the size-predation experiment?)

Figure 8.4 is mildly alarming at first sight, showing that the estimate of the size refuge changes markedly when we incorporate the data from the functional response experiment. This suggests a major difference between the two experiments. A closer

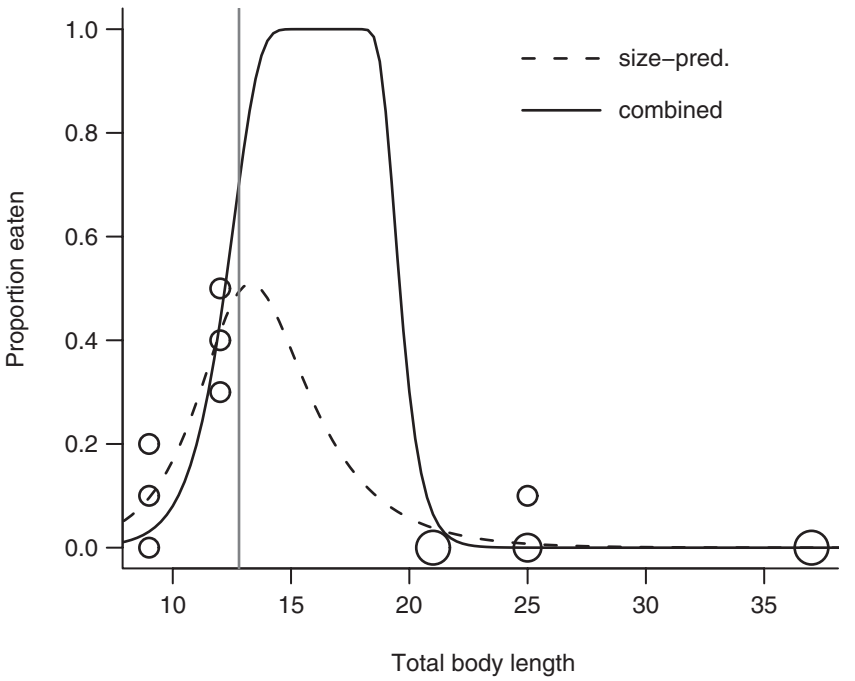

Figure 8.4 Observed number eaten as a function of size; predicted values from size-predation experiment only and from all data combined.

look, however, shows that the major difference between the results falls in a region where we have no data, between 12.8 and 21 mm body length. The slightly higher predation rate in the functional response experiment (even corrected for predator exposure) pulls the curve up.

How would we go about quantifying the uncertainty in the two curves and convincing ourselves that they're not (statistically) significantly different?

Calculating the estimates of the proportion eaten at size 12.8 mm from the size-predation fit alone:

```
> c1 = coef(FSP.modlogist)
> FSP.expprop.mean = modlogist(12.8, c1["eps"], c1["beta"],
+     c1["phi"])
```

and from the combined fit:

```
> c2 = coef(FPcomb)
> FP.expprop.mean = prop.eaten(N0 = 10, S = 12.8, c2["h"],
+     P = 2/0.08, T = 3, eps = c2["eps"], beta = c2["beta"],
+     phi = c2["phi"])
```

The estimated predation proportions are 0.49 for the size-predation experiment alone and 0.7 for the combined data—a difference that certainly might be biologically significant, if it were statistically significant.

As discussed in Chapter 7, population prediction intervals are a simple way to calculate the confidence intervals of a quantity of interest that is not a parameter in

the model. Using `mvrnorm` to generate 5000 values from the sampling distribution of the parameters and calculating the 95% population prediction intervals of the size-predation data:

```
> set.seed(1001)
> FSP.expprop.pars = mvrnorm(5000, mu = c1,
+     Sigma = vcov(FSP.modlogist))
> FSP.expprop.val = numeric(5000)
> for (i in 1:5000) {
+     FSP.expprop.val[i] = modlogist(12.8, FSP.expprop.pars
+         [i, 1], FSP.expprop.pars[i, 2], FSP.expprop.pars
+         [i, 3])
+ }
> FSP.expprop.ppi = quantile(FSP.expprop.val, c(0.025,
+     0.975))
```

Doing the same thing for the combined fit:

```
> FP.expprop.pars = mvrnorm(5000, mu = c2,
+     Sigma = vcov(FPcomb))
> FP.expprop.val = numeric(5000)
> for (i in 1:5000) {
+     FP.expprop.val[i] = prop.eaten(N0 = 10, S = 12.8,
+         P = 2/0.08, T = 3, h = FP.expprop.pars[i,
+             "h"], eps = FP.expprop.pars[i, "eps"],
+         beta = FP.expprop.pars[i, "beta"],
+             phi = FP.expprop.pars[i, "phi"])
+ }
> FP.expprop.ppi = quantile(FP.expprop.val, c(0.025,
+     0.975))
```

	Mean	Low	High
Size-predation	0.494	0.397	0.852
Combined	0.702	0.641	0.992

The results show that the uncertainty in the estimates is large enough that at least the confidence limits of the size-predation estimates (0.4, 0.85) overlap with the estimate from the combined data (0.7), if not vice versa.

Vonesh and Bolker (2005) took results like these (although they did not try fitting the combined data as we have done here) and used them together with size-dependent growth rate estimates from a growth experiment to simulate the survival of tadpoles hatching at different sizes. They found that because smaller-starting tadpoles grew faster through the window of vulnerability between 10 and 20 mm, their overall survival was comparable to tadpoles that hatched at a larger size.

This analysis suggests several more questions:

- Because it must compromise between two sets of data with slightly different survival rates, the fit of the combined curve to the size-predation data is slightly worse than the fit of the size-predation curve itself (Figure 8.4). We initially rejected the need for a beta-binomial model to account for overdispersion, but the larger deviations suggest that it might be worth testing again.

- Following Vonesh and Bolker (2005), we assumed that predator efficiency scaled with area, not volume; this approach may have understated the predator threat in the functional response experiment, leading to an inflation of the expected proportion eaten per unit of exposure. The total predator exposure ($P \times T$) in the functional response experiment was $14 \times 3 = 42$ predator-days, in contrast to $3 \times 2 = 6$ predator-days for the size-predation experiment. If we calculate PT/area for each experiment and take the ratio, we get a relative risk of $43.8/74.6 = 0.6$; overall predator pressure per unit area was lower in the functional response experiment. On the other hand, repeating the same calculation but scaling by volume instead gives a risk ratio of $0.19/0.24 = 0.8$—less difference, leading to less inflation of the per-predator risk in the functional response. We could adjust the model by adding a scaling factor to account for the differences between the experiments, and tentatively interpret it in terms of the geometry of the foraging arena (Petersen et al., 1999). While we clearly don't have enough data to make a decision just from these two experiments, the discrepancy between the results of the two experiments does open up some interesting questions.

8.2 Goby Survival

Next, we will take a look at the effects of density and "quality" (spatial variation in habitat quality correlated with natural rates of immigration) on the survival of the small marine gobies *Elacatinus evelynae* and *E. prochilos* in field experiments (Wilson, 2004).

The questions here are straightforward: How fast do fish die (or disappear) at different levels of density and quality? Do quality, density, or their interaction (i.e., an effect of quality on the density-dependent mortality rate) have significant effects on mortality?

As a reminder, the data contain information on the survival of marine gobies in experiments where ambient density was manipulated on coral heads with different background settlement rates. Settlement rates were suspected to be correlated with some unknown aspect of environmental quality, such as flow patterns or availability of refuges (Wilson and Osenberg, 2002), which revealed itself through lower mortality rates (Figure 8.6).

8.2.1 Preliminaries

Load and attach the data, remembering to detach them later:

```
> library(emdbookx)
> data(GobySurvival)
> attach(GobySurvival)
```

In the data, time starts from day 1 (the day the fish were put on the reef) and runs until day 12; any fish that survived past the end of the experiment (i.e., that were still present on day 12) were given a "last day seen" (`d2`) value of 70 in the original

data set. For the following analysis, time should start from zero and run to ∞ (the cumulative distribution functions we will be using can handle infinite values), so we will subtract 1 from `d1` and `d2` and set the ending value of `d2` to `Inf`:

```
> day1 = d1 - 1
> day2 = ifelse(d2 == 70, Inf, d2 - 1)
```

As discussed in Chapter 7, we will use the Weibull distribution to fit the data, allowing for the observed decrease in mortality rate over time. We are interested in whether mortality is density-dependent, and whether quality affects either the density-independent or the density-dependent mortality rate. We may need to allow for the possibility that different experiments show different results (this data set combines the results from five experiments run over the course of three years).

The most complete model of the survival time of an individual fish in experiment x with density (number of neighboring fish) d and quality (background settlement rate) q would be

$$\begin{aligned} T &\sim \text{Weibull}(a_x(d,q), s_x(d,q)) \\ a_x(d,q) &= \exp(\alpha_{a,x} + \beta_{a,x} \cdot q + (\gamma_{a,x} + \delta_{a,x} \cdot q)d) \\ s_x(d,q) &= \exp(\alpha_{s,x} + \beta_{s,x} \cdot q + (\gamma_{s,x} + \delta_{s,x} \cdot q)d). \end{aligned} \tag{8.2.1}$$

In other words, we are fitting the shape and scale parameters on the log scale. For both the (log) shape and scale parameter we are allowing for a baseline or intercept value (α), a linear effect of quality (β), a linear effect of density (γ), and an interaction between density and quality (δ)—i.e., a linear effect of quality on the density-dependent mortality coefficient. As indicated by the x in the subscripts, we are also allowing each parameter to be different in each experiment, for a total of 40 (!) parameters. Given that we have only 369 observations, unevenly divided among experiments (with as few as 11 observations in an experiment), and that each observation tells us fairly imprecisely when a fish disappeared, this model is certainly more complex than we can hope to fit.

We might try anyway, fitting all possible submodels and using model-selection rules to decide which pieces really belong in the model,* but even so there would be far too many submodels to consider. There are two possibilities for the intercept parameter α (the same for all experiments or different among experiments), and three for each of the other parameters β, γ, and δ (zero for all experiments, meaning no effect of density or quality or their interaction; nonzero but the same for all experiments; or different for different experiments). There are 34 possible models for shape and 34 for scale,† or $34^2 = 1156$ models in total, even for this moderate-sized problem!

We must make some a priori decisions about which parameters to drop—decisions made harder by the difficulty of graphically representing the dependence of survival on continuous covariates. Figure 8.5 shows the effects of the shape

* The statistical equivalent of the advice of a crusading abbot who when asked how to tell the innocents and the heretics apart said, "Kill them all, God will recognize his own."

† You might expect $2 \times 3 \times 3 \times 3 = 54$ models for each parameter of the Weibull, but there are a few combinations that don't make sense—specifically, fitting the δ (density-quality interaction parameter) if either the density or quality effect is set to zero.

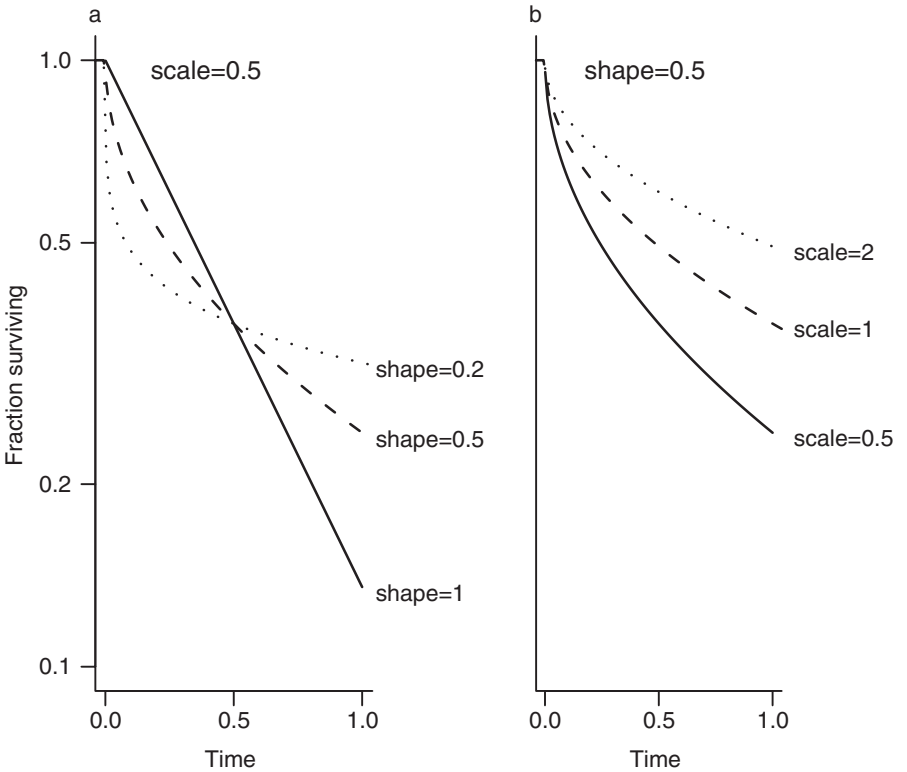

Figure 8.5 Comparisons of Weibull distributions with differing scale and shape parameters. The R commands to plot the curves are variations on `curve(pweibull(x,shape=...,scale=...,lower.tail=FALSE))`.

and scale parameter on the Weibull distribution. Comparing these differences to the survival curves in Figure 8.6 suggests that the scale, but not the shape, of the Weibull distribution varies between density and quality categories. Figure 8.6 also suggests an interaction between quality and density categories, because survival in the low-quality/high-density category is considerably below that in any other category. Figure 8.6 does not separate the results of different experiments. Drawing this figure to check might be worth while, but for now we will assume that the only possible difference among experiments is in the baseline scale parameter, not in the effects of density and quality. Wilson (2004) used a standard survival analysis to demonstrate nonsignificant interactions between experiment and density/quality, supporting this assumption.

These simplifications reduce our most complex model to

$$\begin{aligned} T &\sim \text{Weibull}(a, s_x(d,q)) \\ s_x(d,q) &= \exp\left(\alpha_{s,x} + \beta_s \cdot q + (\gamma_s + \delta_s \cdot q)d\right), \end{aligned} \tag{8.2.2}$$

with nine parameters (five for differences in scale among experiments, three for the effects of density and quality and their interaction on scale, and one for the shape parameter). Our suite of models reduces to 10. If we denote the simplest model (a single shape and scale parameter) by 0; the presence of experiment effects ($\alpha_i \neq \alpha_j$ for

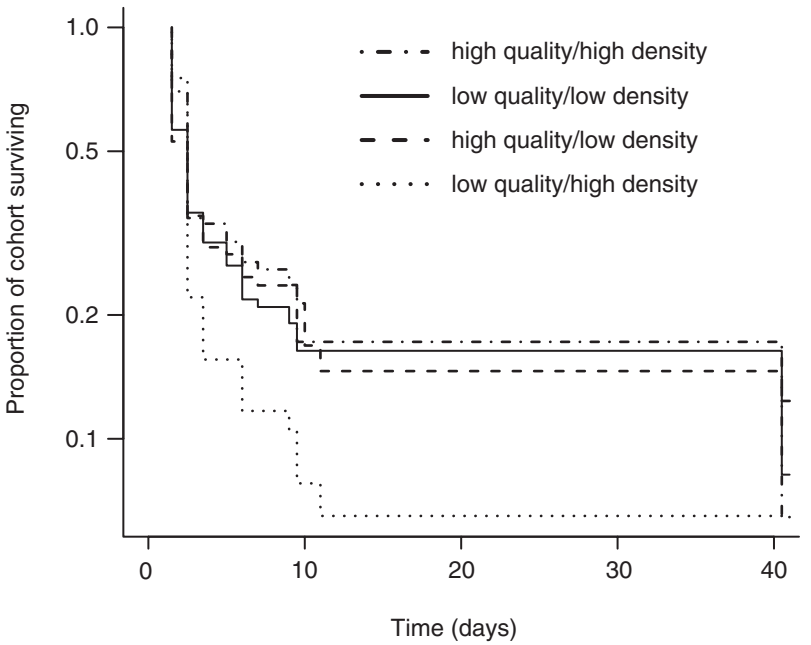

Figure 8.6 Goby survival curves by quality and density categories (above/below median values), based on mean survival time `(d1+d2)/2`.

at least one pair of experiments) by `x`; a quality effect ($\beta_s \neq 0$) by `q`; a density effect ($\gamma_s > 0$) by `d`; and a quality-density interaction ($\delta_s > 0$) by `i`, then our remaining models with their numbers of parameters are

`0` (2)	`x` (6)	`xq` (7)	`xqd` (8)	`xqdi` (9)
	`q` (3)	`xd` (7)	`qdi` (5)	
	`d` (3)	`qd` (4)		

(These are all combinations of `x`, `q`, `d`, and `i`, with the restriction that `i` cannot be included without both `q` and `d`.) If we wanted to allow the shape parameter to vary with quality and density but not experiment, we would have a most-complex model with 12 parameters and a total of 40 (4×10) model possibilities.

Here is the most complex model, which fits scale and shape parameters that differ with quality, density, and their interaction:

```
> NLL.GS.xqdi = function(lscale0, lscale.q, lscale.d,
+     lscale.i, lscale.x2, lscale.x3, lscale.x4, lscale.x5,
+     lshape) {
+     lscalediff = c(0, lscale.x2, lscale.x3, lscale.x4,
+         lscale.x5)
+     scale = exp(lscale0 + lscalediff[exper] + lscale.q *
+         qual + (lscale.d + lscale.i * qual) * density)
+     shape = exp(lshape)
+     -sum(log(pweibull(day2, shape, scale) - pweibull(day1,
+         shape, scale)))
+ }
```

The only unusual thing here is that we've parameterized the difference among experiments so that the baseline parameter (`lscale0`) represents the log of the scale parameter (at density = 0 and quality = 0) in experiment 1, while the experiment parameters (`lscale.x2`, etc.) represent the differences between experiment 1 and the other experiments. This parameterization, which is consistent with the way that other functions in R define parameters, makes it possible to test the hypothesis that all experiments are the same by setting `lscale.x2` and the other experiment parameters to zero. The differences among parameters are indexed by `exper` and added to the baseline value along with the effects of density and quality.

Since we don't know exactly when (between `day1` and `day2`) a given fish disappeared, we calculate the probability that it disappeared somewhere between `day1` and `day2` taking the difference between the probability that it disappeared before `day2` (`pweibull(day2,...)`) and the probability that it disappeared before `day1` (`pweibull(day1,...)`); we take the log only *after* calculating the difference.

What about starting values for this model? The mean of the Weibull distribution with shape a and scale s is $s\Gamma(1 + 1/a)$, which for an exponential ($a = 1$) is equal to s. We'll start $\log(s)$ from the log of the overall mean survival time (calculated from `d1` and `d2` rather than `day1` and `day2` because `day2` contains infinite values that will mess up the mean calculation), and $\log(a)$ from 0, which represents an exponential distribution. Since the rest of the parameters represent differences from the baseline case, we'll try starting them all from zero.

```
> totmeansurv = mean((d1 + d2)/2)
> startvals.GS = list(lscale0 = log(totmeansurv),
+     lscale.x2 = 0, lscale.x3 = 0, lscale.x4 = 0,
+     lscale.x5 = 0,
+     lscale.q = 0, lscale.d = 0, lscale.i = 0, lshape = 0)
> GS.xqdi = mle2(NLL.GS.xqdi, startvals.GS)
```

Looking at the estimates of the parameters and their approximate p-values:

```
> summary(GS.xqdi)

Maximum likelihood estimation

Call:
mle2(minuslogl = NLL.GS.xqdi, start = startvals.GS)

Coefficients:
            Estimate   Std. Error  z value       Pr(z)
lscale0     1.9506010  0.7450665     2.6180    0.008844 **
lscale.q   -0.0137277  0.0993038    -0.1382    0.890051
lscale.d   -0.2198680  0.0973726    -2.2580    0.023945 *
lscale.i    0.0126382  0.0130451     0.9688    0.332644
lscale.x2  -1.0707399  0.5000217    -2.1414    0.032243 *
lscale.x3  -0.7677602  0.3830876    -2.0041    0.045055 *
lscale.x4  -0.1315136  1.0460335    -0.1257    0.899949
lscale.x5   0.0048526  0.9516556     0.0051    0.995932
lshape     -1.0016188  0.0944042   -10.6099  < 2.2e-16 ***
---
```

```
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

-2 log L: 886.122
```

From this summary, it appears that there may be an effect of experiment (experiments 2 and 3 both show significantly shorter survival times than experiment 1), an effect of density, and a shape parameter that is significantly less than 1 ($\log(a) < 0$)—that is, per capita mortality declines significantly with time.

In a stepwise analysis, we would continue by dropping the interaction term from the model (dropping the parameters for experiments 4 and 5 doesn't really make sense, since they are part of the overall difference among experiments). One shortcut for dropping terms from an `mle2` fit, rather than writing another likelihood function that is missing one term, is to use the `fixed` argument to set a subset of the parameters to zero. For example, to drop the interaction term from the model:

```
> GS.xqd = mle2(NLL.GS.xqdi, startvals.GS,
+     fixed = list(lscale.i = 0))
```

We can use the Likelihood Ratio Test on particular series of nested hypotheses to test specific conclusions. For example, we might be most interested in testing whether quality and density have an effect. We attempt to drop the interaction term first, then quality, then density. Because the differences among experiments are potentially important, and an unavoidable part of the experimental design, we leave them in the model. Therefore we want to test the sequence of models `xqdi` → `xqd` → `xd` → `x`.

Fitting the remaining two models in the sequence:

```
> GS.xd = mle2(NLL.GS.xqdi, startvals.GS,
+   fixed = list (lscale.i = 0, lscale.q = 0))
> GS.x = mle2(NLL.GS.xqdi, startvals.GS,
+   fixed = list (lscale.i = 0, lscale.q = 0, lscale.d = 0))
```

Applying `anova` to run the Likelihood Ratio test:

```
> anova(GS.xqdi, GS.xqd, GS.xd, GS.x)

    Tot Df   Deviance   Chisq    Df   Pr(>Chisq)
1      9      886.12
2      8      887.04    0.9139   1     0.33907
3      7      890.77    3.7384   1     0.05318   .
4      6      895.30    4.5210   1     0.03348   *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

This analysis confirms the results of `summary` on the most complex model to some extent. It finds that the effect of the interaction (model 1 vs. 2) is insignificant and the effect of density is significant at $p = 0.03$ (model 3 vs. 4). The effect of quality (when added to a model that already accounts for density) is weakly significant. The parameter values (`coef(GS.xqd)`) show the positive effect of quality (0.076) to be about half the negative effect of density (-0.149) on the log scale; adding one competitor to a reef decreases the scale parameter (and hence survival) by a factor of $e^{-0.149} = 0.86$, while an additional background settler indicates some element

TABLE 8.2
Comparison of goby survival models

Model	Parameters	ΔAIC	AIC Weights	ΔAIC_c	ΔBIC
qd	4	0.00	0.23	0.00	2.54
xqd	8	0.25	0.20	0.54	18.44
qdi	5	0.92	0.15	0.98	7.38
xqdi	9	1.34	0.12	1.73	23.44
d	3	1.37	0.12	1.32	0.00
xd	7	1.99	0.09	2.19	16.27
xq	7	4.02	0.03	4.22	18.30
x	6	4.51	0.02	4.63	14.88
q	3	4.85	0.02	4.80	3.48
0	2	5.50	0.02	5.42	0.22

of quality that increases survival on average by a factor of $e^{0.076} = 1.08$. (You can interpret small coefficients on the log scale as approximate percentage differences: $0.076 \approx 8\%$ increase and $-0.149 \approx 15\%$ decrease.)

Alternatively, we can simply fit the remaining six models (`qdi`, `qd`, `xq`, `d`, `q`, `0`—not shown) and use information criteria (`AICtab`, `AICctab`, or `BICtab`) to compare the results. Table 8.2 shows perhaps too much information—because of the different weighting used by the different information criteria, they give qualitatively different answers. AIC and AIC_c prefer the model that incorporates the effects of quality and density, with all the models considered plausible (ΔAIC, $\Delta AIC_c < 6$ for all candidate models) but with the simplest models weighted very little: In contrast, BIC prefers the simplest models (`0`, `d`), ruling out the most complex ones ($\Delta BIC > 10$ for `xqd`, `xqdi`, `xd`, `xq`, `x`).

What should one conclude in this situation, with too many possible answers? There isn't really a good answer, except that one should decide *in advance* which model selection approach (if any) comes closest to answering the kind of question you have, rather than trying several and then having to choose among the answers. Here there is fairly strong evidence that density decreases survival, and that the effect of quality is about half as strong (per fish present) as that of density. In terms of the range of values used in the experiment, density and quality have approximately equivalent effects (density has a range of 9, from 2 to 11, while quality ranges from 1 to 18).

This particular analysis leaves few loose ends, but there are a number of possible directions for further exploration:

- We have followed standard survival analysis in making the mortality rate an exponential function of covariates such as density. Fisheries biologists commonly model mortality as a linear (additive) function of density instead (i.e., Prob(survival to t) proportional to $e^{-a+b \cdot d}$ rather than Prob(survival to t) proportional to $e^{-e^{a+b \cdot d}}$). The exponential analysis is more convenient because it

guarantees that the mortality rate will always be positive regardless of the parameters, thus avoiding the need for constrained optimization. For small mortality rates the analysis will give approximately the same answers, since by Taylor expansion the exponential is approximately linear near zero. It would be interesting to redo the analysis with a linear model and see how similar the answers were. More challengingly, one could explore the dependence of survival on density and quality in greater detail—perhaps graphically—and see if a more flexible function could give a better answer, although with this small a data set greater flexibility might not be warranted.

- We ignored differences in shape parameter. Returning to explore the possibilities of differences in shape (representing the differences in change in mortality over time) some more, would be interesting as would exploring with a wider variety of data; does the shape parameter vary with the mode of mortality?

8.3 Seed Removal

For the Duncan seed predation/seed removal data, some of the ecological questions are: How does the probability of seed removal vary as a function of distance from the forest edge (10 or 25 m)? With species, possibly as a function of seed mass? By time?

Since most of the predictor variables are categorical in this case (species; distance from forest), the deterministic models are relatively simple—simply different probabilities for different levels of the factors. On the other hand, the distribution of the number of seeds taken is unusual, so most of the initial modeling effort will go into finding an appropriate stochastic model.

8.3.1 Preliminaries

Load the data:

```
> data(SeedPred)
```

Drop `NA`s and records where there are zero seeds available; attach the results, remembering to detach them later.

```
> SeedPred = na.omit(subset(SeedPred, available > 0))
> attach(SeedPred)
```

About 90% of the data consist of "zero taken" entries. We don't want to ignore these data, but sometimes we can see more if we look only at the nonzero cases; we'll use `nz` for that case.

```
> nz = subset(SeedPred, taken > 0)
```

8.3.2 Stochastic Model: Which Distribution?

I used `barchart` from the `lattice` package to look at the data in a variety of different ways—rearranging the order of the factors in the table to get different arrangements

of panels and bars, plotting data with zero-taken data included and excluded, and dropping factors from the `table` command to see coarser views of the data:

```
> barchart(table(nz$taken, nz$available, nz$dist,
+     nz$species), stack = FALSE)
> barchart(table(nz$taken, nz$species, nz$dist,
+     nz$available), stack = FALSE)
> barchart(table(nz$species, nz$available, nz$dist,
+     nz$taken), stack = FALSE)
> barchart(table(nz$available, nz$dist, nz$taken),
+     stack = FALSE)
> barchart(table(nz$available, nz$species, nz$taken),
+     stack = FALSE)
```

I could also have included the argument `subset=taken>0`, instead of defining `nz` beforehand, to restrict the plots to nonzero data.

Plot all data (not just cases where some seeds are taken):

```
> barchart(table(available, dist, taken), stack = FALSE)
```

Plot by date:

```
> tcumfac = cut(nz$tcum, breaks = c(0, 20, 40, 60,
+     180))
> barchart(table(nz$available, tcumfac, nz$taken),
+     stack = FALSE)
> barchart(table(available, tcumfac, taken), stack = FALSE)
```

Two additional useful arguments are `auto.key=TRUE`, to draw a legend for the bar colors, and `scales=list(relation="free")`, to allow different scales in each panel.

As with the reed frog predation experiment, the data are discrete and the results have an upper limit (i.e., the number of seeds available for removal at the beginning of the interval). The zero-inflated binomial introduced in Chapter 4 might make sense, if there were more zeros in the data set than expected from the binomial sampling process (e.g., if the probability distribution had modes both at zero and away from zero). This distribution would be appropriate if predators sometimes missed the site entirely. However, Figure 8.7 shows that the seed removal data set doesn't look like a zero-inflated binomial either, because the distribution is lowest in the middle and increases gradually for higher or lower values. Compare that with Figure 4.1 (p. 107), which shows that the probability distribution function of the zero-inflated binomial distribution usually drops toward zero, then has a spike at zero ($p(0) > p(1)$, $p(1) < p(2)$).

Next I tried the beta-binomial distribution, which allows for variability in the underlying probabilities per trial and can be bimodal at 0 and N for extreme values of the overdispersion parameter, and a zero-inflated beta-binomial distribution.

One should really test the fits of distributions on a small piece of the data set or allowing for different parameters for each combination of factors; variation among groups can mask the shape of the underlying distribution. However, trying to fit parameters for an unknown distribution for all combinations of factors simultaneously can be tedious, and the exploratory graphical analysis described

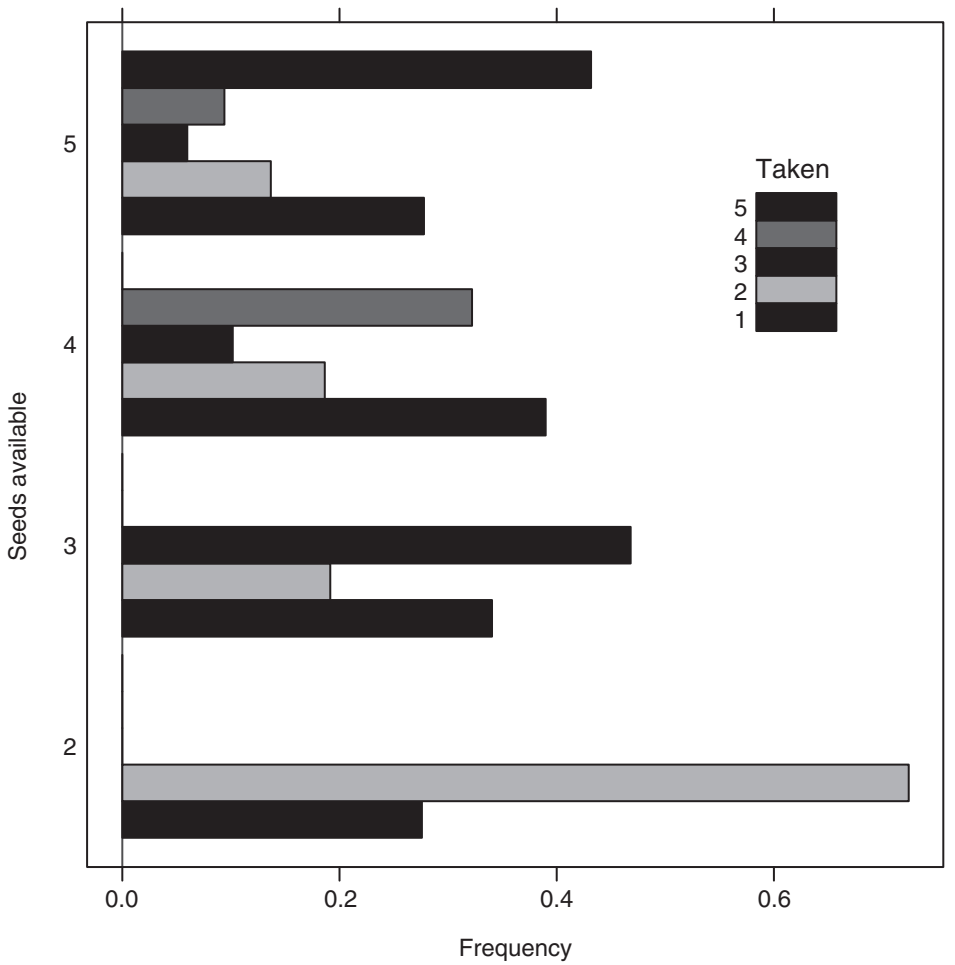

Figure 8.7 Distribution of overall number of seeds taken as a function of the number available, when number available > 1 and number taken > 0.

above convinced me that the pattern shown in Figure 8.7 holds up even when the data are disaggregated by species, distance, or date.

Using the `dzinbinom` function in the `emdbook` package as a model, I constructed probability density functions for the zero-inflated binomial (`dzibinom`) and zero-inflated beta-binomial (`dzibb`):

```
> dzibinom = function(x, prob, size, zprob, log = FALSE) {
+     logv = log(1 - zprob) + dbinom(x, prob = prob,
+         size = size, log = TRUE)
+     logv = ifelse(x == 0, log(zprob + exp(logv)),
+         logv)
+     if (log)
+         logv
+     else exp(logv)
+ }
```

```
> dzibb = function(x, size, prob, theta, zprob, log = FALSE)
+     { logv = ifelse(x > size, NA, log(1 - zprob) +
+         dbetabinom(x, prob = prob, size = size,
+             theta = theta, log = TRUE))
+     logv = ifelse(x == 0, log(zprob + exp(logv)),
+         logv)
+     if (log)
+         logv
+     else exp(logv)
+ }
```

Next I took a shortcut and used the formula interface to `mle2` rather than writing an explicit negative log-likelihood function. Since the zero-inflation probability must be between 0 and 1, I fitted it on a logit scale, using `plogis` to transform it on the fly:

```
> SP.zibb = mle2(taken ~ dzibb(size = available, prob,
+     theta, plogis(logitzprob)), start = list(prob = 0.5,
+     theta = 1, logitzprob = 0))
```

There were warnings about `NaN`s in `lbeta`, but the final answers look reasonable. I was surprised to see that the zero-inflation probability was so small: `plogis(-2.33)` = 0.089. I suspected that the zero-inflation parameter and the overdispersion parameter (θ) might both be affecting the number of zeros, so I checked the correlations among the parameters:

```
> cov2cor(vcov(SP.zibb))
```

	prob	theta	logitzprob
prob	1.0000000	0.2885011	0.9867901
theta	0.2885011	1.0000000	0.3436282
logitzprob	0.9867901	0.3436282	1.0000000

Indeed, `logitzprob` and `prob` are 99% correlated—suggesting that we could drop the zero-inflation parameter from the model.

```
> SP.bb = mle2(taken ~ dbetabinom(size = available,
+     prob, theta), start = list(prob = 0.5, theta = 1))

> logLik(SP.bb) - logLik(SP.zibb)

'log Lik.' 0.07956568 (df=2)
```

The log-likelihood difference is only about 0.08. Even allowing for the fact that the null value of the zero-inflation parameter is on the boundary, so that the appropriate $\bar{\chi}^2_1$ p-value is half the usual χ^2_1 p-value, this difference is certainly not significant.

Just for completeness, I fitted the zero-inflated binomial too (although I didn't think it would fit well):

```
> SP.zib = mle2(taken ~ dzibinom(size = available,
+     prob = p, zprob = plogis(logitzprob)), start = list
+     (p = 0.2, logitzprob = 0))
```

Using AIC to compare all three distributions confirmed my suspicions:

```
> AICtab(SP.zib, SP.zibb, SP.bb, sort = TRUE, weights =
+     TRUE)
```

	AIC	df	weight
SP.bb	3626.1	2	0.746
SP.zibb	3628.3	3	0.254
SP.zib	4045.6	2	<0.001

Figure 8.8 compares the predictions of the different distributions, with stacked barplots showing the breakdown of different numbers of seeds taken for each number of seeds available.

The R code to calculate this distribution for the data first computes the table of number-taken-by-number-available, then uses `sweep` to divide each column (margin 2) by its sum:

```
> comb = table(taken, available)
> pcomb = sweep(comb, 2, colSums(comb), "/")
```

The equivalent computation for the zero-inflated beta-binomial sets up an empty matrix with six rows (for 0 to 5 seeds taken) and five columns (for 1 to 5 seeds available). For each number available N, it then sets the first $N+1$ rows in column N of the matrix to the predicted probability of taking 0 to N seeds.

```
> mtab = matrix(0, nrow = 6, ncol = 5)
> for (N in 1:5) {
+     cvals = coef(SP.zibb)
+     mtab[1:(N + 1), N] = dzibb(0:N, size = N,
+     prob = cvals["prob"],
+         theta = cvals["theta"], zprob = plogis(cvals
+         ["logitzprob"]))
+ }
```

Similar calculations work for the other two distributions.

As we would expect from the statistical results so far, the zero-inflated beta-binomial and beta-binomial predictions look nearly identical, and much closer than the zero-inflated binomial results. However, there are still visible discrepancies for the cases of 4 and 5 seeds available—the predicted distributions are more regular, and have more even distributions, than the observed. None of the three models capture the increased probability that one seed would be taken (dark blocks) when four or five seeds were available.

We can calculate standard Pearson χ^2 p-values for the probability of the observed numbers taken for each number of seeds available:

```
> pval = numeric(5)
> for (N in 1:5) {
```

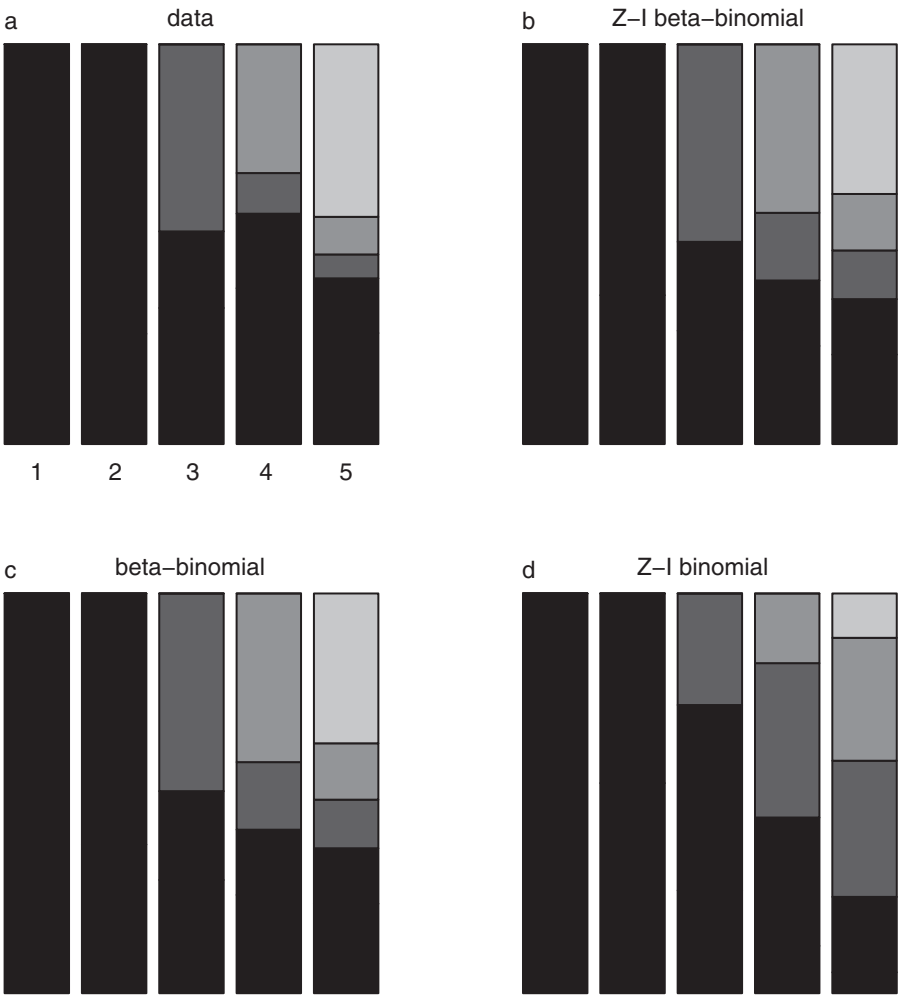

Figure 8.8 Observed and predicted distribution of number of seeds taken as a function of number available. (Zero-taken results are omitted, and columns are rescaled to add to 1.)

```
+      obs = comb[1:(N + 1), N]
+      prob = mtab[1:(N + 1), N]
+      pval[N] = chisq.test(obs, p = prob)$p.value
+ }
```

The *p*-values are:

1	2	3	4	5
0.53	0.29	0.81	0.01	<0.001

There are still statistically significant discrepancies between the expected and observed distributions when 4 or 5 seeds are available. We could try to find a way to make the stochastic model more complex and accurate, but we have reached the limit of what we can do with simple models, and we may also have reached the limit

of what we can do with the data. The mechanism for the pattern remains obscure. While I can imagine mechanisms that would lead to all seeds or none being taken, it's hard to see why it's least likely that 3 out of 5 available seeds would be taken. I suspect that there is some disaggregation of the data by species, date, etc., that would divide stations into those where few or many seeds were taken, with an extreme pattern in each case that combines to create the observed bimodal pattern, but I haven't been able to find it.

8.3.3 Deterministic Model: Differences among Species, Distance, Space, and Time

Now we can check for differences among distances from the forest, species, and possibly differences in space and time: How does the distribution of number of seeds removed vary? Does p, the overall probability that a seed will be removed, vary? Does θ (the overdispersion parameter, which in this case is more related to the probability that any seeds will be removed) vary? Do they both vary?

8.3.3.1 DIFFERENCES AMONG TRANSECTS (DISTANCE FROM EDGE)

`mle2`'s formula interface allows us to specify that some parameters vary among groups, by giving a `parameters` argument which is a list of the formulas for each group (p. 206). Here I wanted to parameterize the model so that `mle2` would estimate the probability and overdispersion parameter for each distance, rather than estimating the parameters for the first transect and the difference between the first and second transect, so I used the formulas `prob~dist-1` and `theta~dist-1` to fit the model without an intercept.

```
> SP.bb.dist = mle2(taken ~ dbetabinom(prob, theta,
+       size = available),
+ parameters = list(prob ~ dist - 1, theta ~
+     dist - 1), start = as.list(coef(SP.bb)))
```

A Likelihood Ratio test on the two models suggests a significant difference between transects:

```
> anova(SP.bb, SP.bb.dist)

Likelihood Ratio Tests
Model 1: SP.bb, taken~dbetabinom(prob,theta,size=available)
Model 2: SP.bb.dist,
          taken~dbetabinom(prob,theta,size=available):
          prob~dist-1, theta~dist-1
Tot    Df    Deviance    Chisq     Df    Pr(>Chisq)
1      2      3622.1
2      4      3615.6     6.4823    2      0.03912   *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Reparameterizing the model in terms of differences between the 10-m and 25-m transect rather than the p and θ values for each transect (i.e., dropping the `-1` in the parameter formulas) allows us to calculate confidence limits on the differences between transects. At the same time, I decided to switch to fitting p on a logit scale and θ on a log scale. With the formula interface, I can do the inverse transformations on the fly with `plogis` and `exp`.

Set up starting values, using `qlogis` (the logit transform) and `log` to transform the estimated values of the p and θ parameters from above.

```
> startvals = list(lprob =
+     qlogis(coef(SP.bb.dist)["prob.dist10"]),
+     ltheta = log(coef(SP.bb.dist)["theta.dist10"]))

> SP.bb.dist2 = mle2(taken ~ dbetabinom(plogis(lprob),
+     exp(ltheta),size = available), parameters =
+     list(lprob ~ dist, ltheta ~ dist), start = startvals)
```

The summary of the model now gives us approximate p-values on the parameters, showing that the difference between transects is caused by a change in p and not a change in θ.

```
> summary(SP.bb.dist2)

Maximum likelihood estimation

Call:
mle2(minuslogl = taken ~ dbetabinom(plogis(lprob),
    exp(ltheta), size = available), start = startvals,
    parameters = list(lprob ~ dist, ltheta ~ dist))

Coefficients:
                    Estimate    Std. Error  z value    Pr(z)
lprob.(Intercept)   -2.7968262   0.0813997   -34.3592  < 2e-16  ***
lprob.dist25         0.2663037   0.1110270     2.3985  0.01646  *
ltheta.(Intercept)  -1.1255457   0.1261399    -8.9230  < 2e-16  ***
ltheta.dist25       -0.0035835   0.1719498    -0.0208  0.98337
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

-2 log L: 3615.627
```

(The highly significant p-values for `lprob.10` and `ltheta.10` are not biologically significant: they merely show that $\text{logit}(p_{10}) \neq 0$ (i.e., $p_{10} \neq 0.5$) and $\log \theta_{10} \neq 0$ ($\theta_{10} \neq 1$), neither of which is ecologically interesting.)

Now reduce the model, allowing only p to vary between transects:

```
> SP.bb.probdist = mle2(taken ~ dbetabinom(plogis(lprob),
+     exp(ltheta), size = available), parameters =
+     list(lprob ~ dist), start = startvals)
```

Both the LRT and the AIC approaches suggest that the best model is one in which p varies between transects but θ does not (although the AIC table suggests that the more complex model with differing θ should be kept in consideration):

```
> anova(SP.bb, SP.bb.probdist, SP.bb.dist)

Likelihood Ratio Tests
Model 1: SP.bb,taken~dbetabinom(prob,theta,size=available)
Model 2: SP.bb.probdist,
          taken~dbetabinom(plogis(lprob),exp(ltheta),size=available):
          lprob~dist
Model 3: SP.bb.dist,
          taken~dbetabinom(prob,theta,size=available):
          prob~dist-1, theta~dist-1
Tot    Df    Deviance    Chisq     Df    Pr(>Chisq)
1      2      3622.1
2      3      3615.6     6.4819    1      0.01090  *
3      4      3615.6     0.0004    1      0.98341
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

> AICtab(SP.bb, SP.bb.probdist, SP.bb.dist, sort = TRUE,
+      weights = TRUE)

                    AIC      df    weight
SP.bb.probdist    3621.6    3     0.678
SP.bb.dist        3623.6    4     0.250
SP.bb             3626.1    2     0.072
```

How big is the difference between transects?

```
> c1 = coef(SP.bb.probdist)
> plogis(c(c1[1], c1[1] + c1[2]))

lprob.(Intercept)    lprob.(Intercept)
   0.05751881           0.07372130
```

The difference is small—6% vs. 7% probability of removal per observation. This difference is unlikely to be ecologically significant, and it reminds us that when we have a big data set (4406 observations) even small differences can be statistically significant. On the other hand, Duncan and Duncan (2000) failed to find a significant difference between the transects—so the likelihood framework is more powerful, and has given us answers in terms (average percent difference in probability of removal) that we can understand.

8.3.3.2 DIFFERENCES AMONG SPECIES

Now I proceeded to test differences among species. First I tried a model with both θ and p varying. (Both parameters are again fitted on transformed scales, logit and log respectively.)

```
> SP.bb.sp = mle2(taken ~ dbetabinom(plogis(lprob),
+     exp(ltheta), size = available), parameters =
+     list(lprob ~ species, ltheta ~species),
+     start = startvals)
```

The parameter estimates (shown in full by `summary(SP.bb.sp)`; here I dropped one column of the table) suggest that, as in the case of differences among transects, differences in p and not in θ are driving the differences among species:

```
                      Estimate    Std. Error       Pr(z)
lprob.(Intercept)     -1.925509      0.1428      < 2.2e-16    ***
lprob.speciescd        0.329247      0.2186        0.1321056
lprob.speciescor      -1.332956      0.2144        5.090e-10  ***
lprob.speciesdio      -0.991505      0.2111        2.645e-06  ***
lprob.speciesmmu      -0.432409      0.2130        0.0423696  *
lprob.speciespol       0.413143      0.2098        0.0489483  *
lprob.speciespsd      -1.274415      0.2207        7.704e-09  ***
lprob.speciesuva      -1.302890      0.2146        1.266e-09  ***
ltheta.(Intercept)    -0.824310      0.2240        0.0002327  ***
ltheta.speciescd      -0.560802      0.3473        0.1063536
ltheta.speciescor      0.016070      0.3292        0.9610611
ltheta.speciesdio     -0.377969      0.3276        0.2485773
ltheta.speciesmmu     -0.618604      0.3354        0.0651542  .
ltheta.speciespol      0.152877      0.3331        0.6462837
ltheta.speciespsd     -0.173435      0.3405        0.6105292
ltheta.speciesuva     -0.058962      0.3341        0.8599198
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

So I fitted a model with only probability p, and not overdispersion θ, varying by species:

```
> SP.bb.probsp = mle2(taken ~ dbetabinom(plogis(lprob),
+     exp(ltheta), size = available), parameters = list
+     (lprob ~ species), start = startvals)
```

Once again, both LRT and AIC suggest that only the p parameters differ among species:

```
> anova(SP.bb.sp, SP.bb.probsp, SP.bb)

Likelihood Ratio Tests
Model 1: SP.bb.sp,
          taken~dbetabinom(plogis(lprob), exp(ltheta),
          size=available): lprob~species, ltheta~species
Model 2: SP.bb.probsp,
          taken~dbetabinom(plogis(lprob), exp(ltheta),
          size=available): lprob~species
Model 3: SP.bb, taken~dbetabinom(prob,theta, size=available)
```

```
Tot   Df   Deviance    Chisq    Df  Pr(>Chisq)
1     16    3460.4
2     9     3469.8      9.3894  7     0.2259
3     2     3622.1    152.2873  7     <2e-16   ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
> AICtab(SP.bb.sp, SP.bb.probsp, SP.bb, sort = TRUE,
+     weights = TRUE)
```

```
               AIC     df    weight
SP.bb.probsp   3487.8    9      0.909
SP.bb.sp       3492.4   16      0.091
SP.bb          3626.1    2     <0.001
```

Now I want to know whether seed mass and p are related. If they were, I could fit a likelihood model where p was treated as a function of seed mass, reducing the number of parameters to estimate and perhaps allowing me to predict removal probabilities for other species on the basis of their seed masses.

```
> SP.bb.probsp0 = mle2(taken ~ dbetabinom(plogis(lprob),
+     exp(ltheta), size = available), parameters = list
+     (lprob ~ species - 1), start = startvals,
+     method = "L-BFGS-B",
+     lower = rep(-10, 9), upper = rep(10, 9))
```

Fitting this model was numerically problematic. In my first attempt, using default methods and parameters, `mle2` found a ridiculous answer (all the logit probabilities were strongly negative, giving removal probabilities near zero) and crashed while evaluating the Hessian. I used `skip.hessian=TRUE` to temporarily stop `mle2` from crashing and `trace=TRUE` to see what was happening. Switching to `method="Nelder-Mead"` helped stabilize the calculation, but it failed to converge until I increased the number of iterations to 3000 (`control=list(maxit=3000)`), and even then it got stuck on a solution that was worse than the previous model. (In this case, since all I am doing is reparameterizing the previous model, `mle2` ought to be able to achieve an equally good fit.) I then went back to BFGS and tried changing the size of the finite difference interval both down (`control=list(ndeps=rep(1e-4,9))`) and up (`control=list(ndeps=rep(1e-2,9))`), neither of which helped. I finally got the model to fit as well as the previous parameterization by switching to L-BFGS-B and setting the parameter boundaries to disallow ridiculous fits.

```
> predprob = plogis(coef(SP.bb.probsp0))[1:8]
> SP.bb.ci = plogis(confint(SP.bb.probsp0,
+     method = "quad"))[1:8, ]
```

Figure 8.9 shows the results: rather than the possible trend toward higher seed removal for larger seeds that I expected, the figure shows elevated removal rates for the three smallest-seeded species (explained by Duncan and Duncan as a possible artifact of small seeds being washed out of the trays by rainfall), and a somewhat elevated rate for species mmu; in this case, I would want to go back and see if there

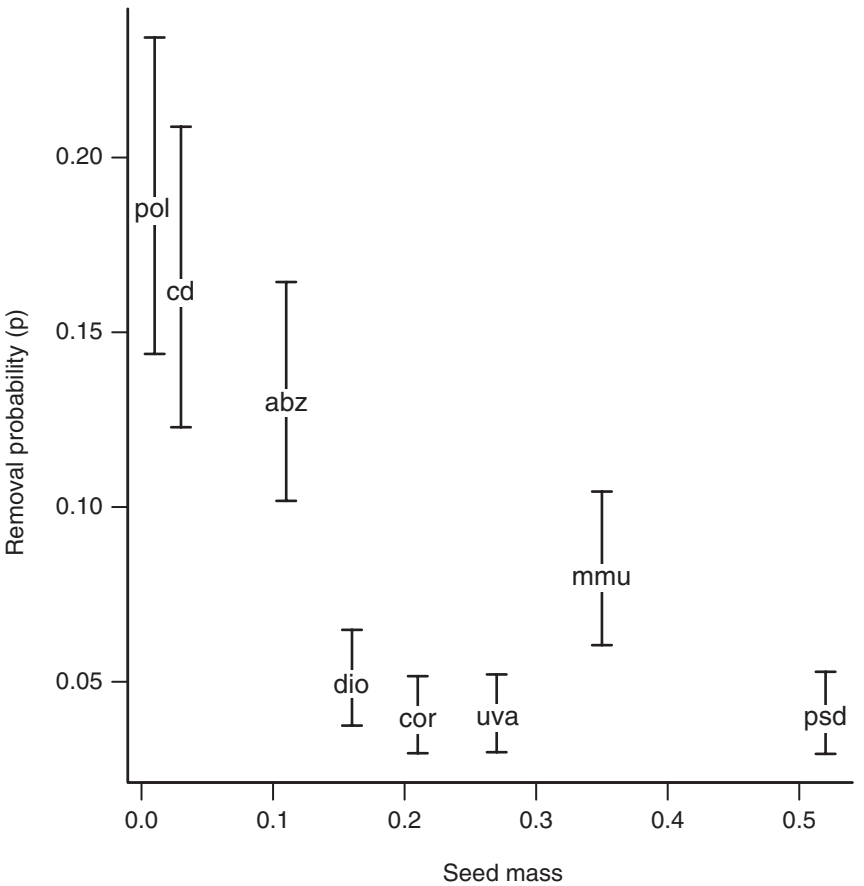

Figure 8.9 Removal probability parameter (p) as a function of seed mass; error bars show quadratic confidence intervals.

was something special about this species' characteristics or the way it was handled in the experiment.

8.3.3.3 IS THERE A SPECIES-DISTANCE INTERACTION?

The initial scan of the data suggested that some species might be more sensitive to the distance from the edge: This possibility is certainly biologically sensible (some species might be taken by specialized seed predators that have more restricted movement), and it is the kind of information that could easily be masked by looking at aggregated data.

Using the formula interface, we can simply say `lprob~species*dist` to allow for such an interaction: if you need to code such a model by hand, `interaction(f1,f2)` will create a factor that represents the interaction of factors `f1` and `f2`.

```
> SP.bb.probspdist = mle2(taken ~ dbetabinom(plogis(lprob),
+     exp(ltheta), size = available), parameters = list
```

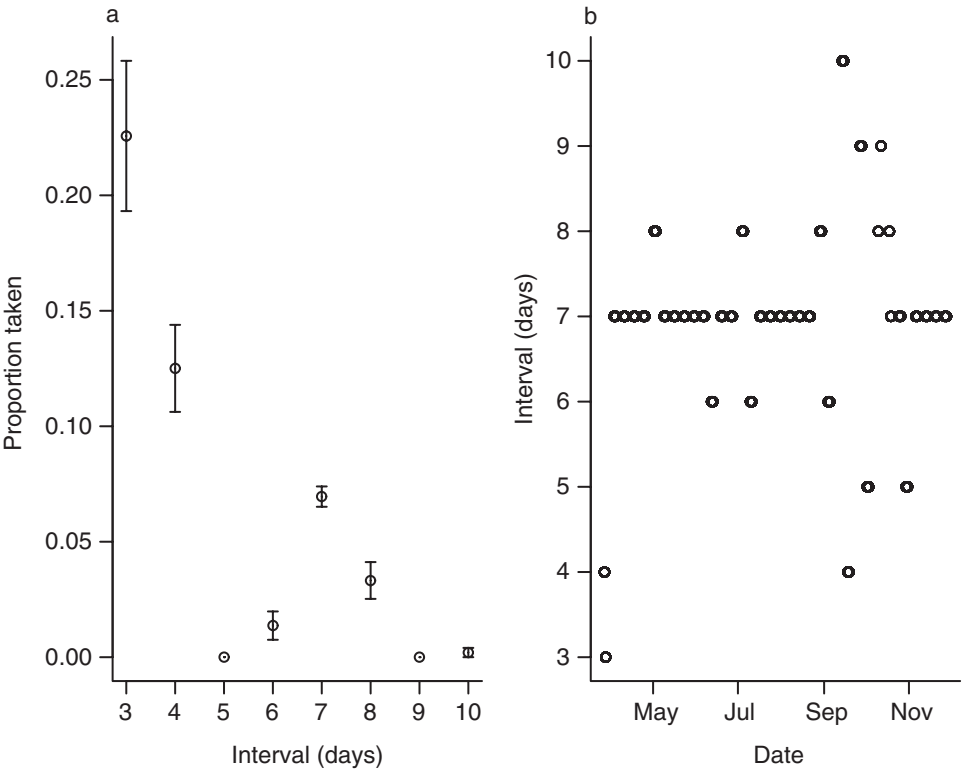

Figure 8.10 Relationships between proportion removed and time interval (Δt), and between Δt and date.

```
+      (lprob ~ species * dist), start = startvals, method =
+      "L-BFGS-B", lower = rep(-10, 9), upper = rep(5, 9))
```

I had to restrict the upper bounds still further, to 5, to make L-BFGS-B happy, since values of 10 gave `NaN` results for some parameter combinations.

A Likelihood Ratio test (`anova(SP.bb.probsp,SP.bb.probspdist)`) gives a p-value of 0.054; AIC says that the model without distance × species interaction is best, but only by a little bit:

```
> AICtab(SP.bb, SP.bb.probsp, SP.bb.probspdist, SP.bb.sp,
+      SP.bb.probdist, SP.bb.dist, weights = TRUE,
+      sort = TRUE)
```

	AIC	df	weight
SP.bb.probsp	3487.8	9	0.559
SP.bb.probspdist	3488.6	17	0.386
SP.bb.sp	3492.4	16	0.056
SP.bb.probdist	3621.6	3	<0.001
SP.bb.dist	3623.6	4	<0.001
SP.bb	3626.1	2	<0.001

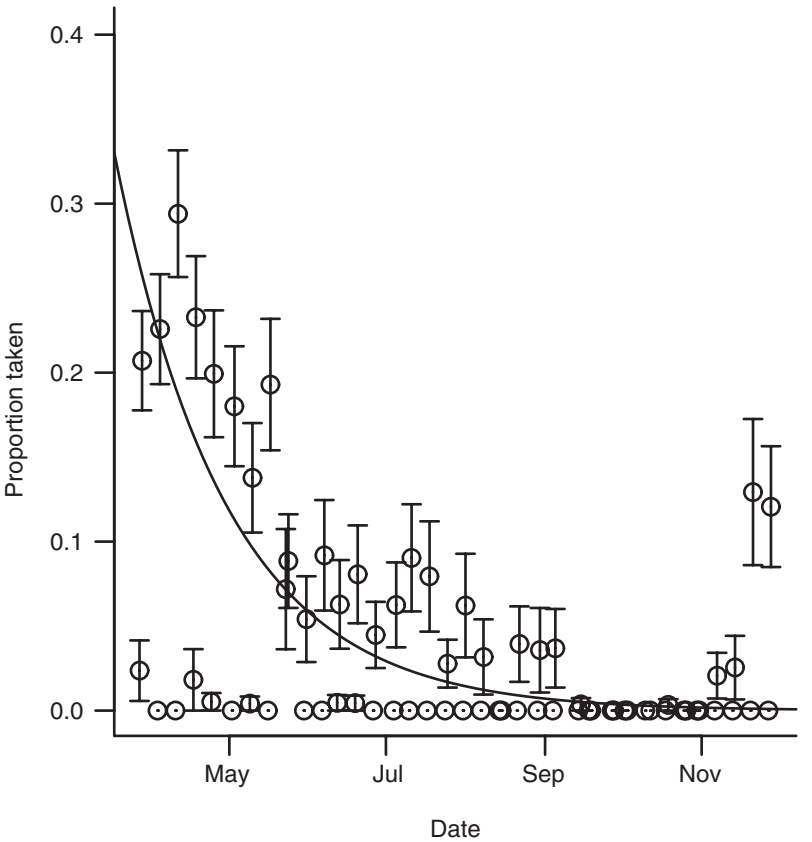

Figure 8.11 Proportion taken as a function of date. The line shows fitted exponential dependence ($p = 0.26 \times e^{-0.023t}$), based on a fitted model that lumps all the species together.

8.3.3.4 OTHER ISSUES: TIME

One issue that I have neglected so far is that the intervals between observations varied between 3 and 14 days. To account for these differences in exposure time, I could use a model like $p = 1 - e^{-r(\Delta t)}$, which assumes that seeds are taken at a constant rate r. Do the predictions improve, or the conclusions change, if I account for the time interval allowed for removal?

Before going to the trouble of building a model, let's look at the data again. Calculate the mean and standard error of the proportion taken, using `tapply` to calculate means and standard deviations of proportions divided up by the time interval (`tint`); then use `table` to calculate the number of observations for each time interval and divide by $\sqrt{n}$ to convert standard deviations to standard errors.

```
> mean.prop.taken = tapply(taken/available, tint, mean,
+     na.rm = TRUE)
> sd.prop.taken = tapply(taken/available, tint, sd,
+     na.rm = TRUE)
```

```
> n.tint = table(tint)
> se.prop.taken = sd.prop.taken/sqrt(n.tint)
```

Figure 8.10a is a surprise: the model $p = 1 - e^{-r(\Delta t)}$ suggests the proportion taken should increase rather than decrease with Δt. What's going on? Figure 8.10b, which plots the time interval between observations against date, gives the answer: the short-interval (3–4 day) observations were mostly made before May, when the removal rate was high, while the longest intervals between observations (10 days) are in September.

This brings us to the issue of temporal variation: we already know from Figure 2.1 in Chapter 2 that the removal rate decreases over time. Figure 8.11 shows the relationship between proportion removed and date, calculated in the same way as the removal vs. Δt relationship. Removal appears to decrease exponentially with time. Replotting the data with a logarithmic y scale suggests that the removal rate might level off above zero, but it's hard to tell. Similarly, it's hard to know what causes the anomalously low proportions for some sampling dates throughout the study and the anomalously high proportions at the very end of the study. Nevertheless, we can add a parameter to the model allowing for exponential decrease in removal rate over time:

```
> SP.bb.probspdate = mle2(taken ~ dbetabinom(plogis(lprob)*
+     exp(-tcum * date), exp(ltheta), size = available),
+     parameters = list(lprob ~ species), start =
+         c(startvals, date = 0), method = "L-BFGS-B",
+         lower = c(rep(-10, 9), 0), upper = c(rep(5, 9),
+         2))
```

The model incorporating date is 237.6 log-likelihood units better—the model should definitely include the effect of date.

We have gotten a lot of mileage from these data, but as always there are more questions we could ask: Do the removal rates of different species drop off at different rates? Can we figure out what causes the anomalous samples in Figure 8.11? Once we have split the data according to these criteria, can we simplify the underlying distribution?