CHAPTER 24

Model checking and comparison

There are generally many options available when modeling a data structure, and once we have successfully fit a model, it is important to check its fit to data. It is also often necessary to compare the fits of different models.

Our basic approach for checking model fit is—as we have described in Sections 8.3–8.4 for simple regression models—to simulate replicated datasets from the fitted model and compare these to the observed data. We discuss the general approach in Section 24.1 and illustrate in Section 24.2 with an extended example of a set of models fit to an experiment in animal learning. The methods we demonstrate are not specific to multilevel models but become particularly important as models become more complicated.

Although the methods described here are quite simple, we believe that they are not used as often as they could be, possibly because standard statistical techniques were developed before the use of computer simulation. In addition, fitting multilevel models is a challenge, and users are often so relieved to have successfully fit a model with convergence that there is a temptation to stop and rest rather than check the model fit. Section 24.3 discusses some tools for comparing different models fit to the same data.

Posterior predictive checking is a useful direct way of assessing the fit of the model to various aspects of the data. Our goal here is not to compare or choose among models but rather to explore the ways in which any of the models being considered might be lacking.

24.1 Principles of predictive checking

Monitoring the quality of a statistical model implies the detection of systematic differences between the model and observed data. Posterior predictive checks set this up by generating replicated datasets from the predictive distribution of the fitted model; these replicated datasets are then compared to the observed dataset with respect to any features of interest. The functions of data and model parameters that we use to compare to the model are called *test summaries* or *discrepancy variables*; we also consider the special case of *test statistics*, which depend on the (replicated) data only. This is the formal treatment of the simulation-based model-checking introduced in Section 8.3.

We use the notation $y = (y_1, \dots, y_n)$ for discrete observed data, X for the matrix of predictor variables, and θ for the vector of all parameters. We assume that a model has been fit and that we have a set of simulations, $\theta^{(s)}$, $s = 1, \dots, n_{\text{sims}}$.

We further assume that, for each of these draws, a replicated dataset, $y^{\text{rep}\,(s)}$, has been simulated from the predictive distribution of the data $p(y^{\text{rep}}|X, \theta = \theta^{(s)})$; the ensemble of simulated datasets $(y^{\text{rep}\,(\text{s})}, \dots, y^{\text{rep}\,(\text{n}_{\text{sims}})})$ thus represents the posterior predictive distribution, $p(y^{\text{rep}}|X, y)$. For simplicity, we suppress the conditioning on X in the notation that follows, but in some examples we shall allow X to vary and simulate X^{rep} as well (see Section 24.2). Predictive simulation of the replicated datasets y^{rep}, conditional on θ, is usually extremely easy—typically requiring nothing more than simulation from known independent distributions—even though obtaining posterior simulations of θ usually requires complicated Markov chain simulation methods.

We check the model by means of discrepancy variables $T(y, \theta)$. If θ were known, one could perform a goodness-of-fit test by comparing the observed $T(y, \theta)$ to the distribution of the discrepancy variables in the replications, $T(y^{\mathrm{rep}}, \theta)$, with the statistical significance of the test summarized by a p-value, $p = \Pr(T(y^{\mathrm{rep}}, \theta) > T(y, \theta)|y, \theta)$. (Here, we consider only one-sided tests, with the understanding that the corresponding two-sided p-value is $2 \cdot \min(p, 1-p)$.) In the more usual case of unknown θ, the test comparison is averaged over the uncertainty in θ (that is, the posterior distribution), with a posterior predictive p-value, $\Pr(T(y^{\mathrm{rep}}, \theta) > T(y, \theta)|y) = \int \Pr(T(y^{\mathrm{rep}}, \theta) > T(y, \theta)|y, \theta)p(\theta|y)d\theta$, which can be estimated from the simulations by $\sum_{s=1}^{n_{\mathrm{sims}}} 1_{T(y^{\mathrm{rep}\,(s)}, \theta(s)) > T(y, \theta^{(s)})}/n_{\mathrm{sims}}$, where 1_A is the indicator function that is 1 if the condition A is true and 0 otherwise.

As to the choice of discrepancy variables, we focus here on methods for detecting systematic discrepancies between model and data, not on the related problem of discovering outliers in otherwise-reasonable mdels. Some of the discrepancy variables we develop have been used in Bayesian methods for outlier detection but there with a focus on individual observations rather than on larger patterns. By comparison, the discrepancy variables we consider often average over sections of the data. In addition, we seek discrepancy variables that are easy to interpret and are also generally applicable to a wide range of problems. In many cases, this means that we would like to check qualitative features of the model (for example, independence, monotonicity, and unimodality) to give a better understanding of directions of model improvement.

Classical residual plots and binned residual plots can be viewed as particular examples of graphical discrepancy variables. If $T(y, \theta)$ is a graph, rather than a number, it would not make sense to compute a p-value, but we still could compare the plot, explicitly or implicitly, with what would be obtained under replications from the model. This is the approach illustrated in Sections 8.3–8.4.

The application of the predictive check method goes as follows. Several discrepancy variables are chosen to reveal interesting features of the data or discrepancies between the model and the data. For each discrepancy variable, each simulated *realized value* $T(y, \theta^{(s)})$ is compared to the corresponding simulated *replicated value* $T(y^{\mathrm{rep}\,(s)}, \theta^{(s)})$. Large and systematic differences between realized and replicated values indicate a misfit of the model to the data, in the sense that the observed data do not look typical, in this respect, of the data predicted under the model. In some cases, differences between the realized data and replications are apparent visually; other times, it can be useful to compute the p-value of a realized discrepancy to see whether it could plausibly have arisen by chance under the model.

In any applied problem, it is appropriate to check aspects of the data and model that are of particular substantive interest. By their nature, such diagnostics can never be routine or automatic, but we can give some general suggestions. First, it is often useful to display the entire dataset (or, if that is not possible for a highly multivariate problem, various data-rich summaries) and compare to some predictive replications of the data to get an idea of what would be expected under the model. Patterns seen in this sort of exploratory check can be used as the basis for more systematic model checks. As in exploratory data analysis in general, the reference distribution of replicated datasets provides a standard of comparison by which the observed discrepancies can be measured—the goal is not to find statistical significance but rather to reveal areas where the data look different from what would have been expected under the model. Second, one can directly compute the predictive distribution of any function of data and parameters using the posterior simulations of $(\theta, y^{\mathrm{rep}})$—and thus directly check the fit with respect to any easily computable discrepancy variable of interest. Third, it often makes sense to set up discrepancy variables with an eye toward how the model might be improved—for example, summarizing between-group variability if one is considering fitting a random effects model.

24.2 Example: a behavioral learning experiment

Experimental data and historical context

We investigate the effectiveness of various model checks for an analysis of a logistic regression model applied to data from a well-known experiment on behavioral learning conducted around 1950. In this experiment, each of 30 dogs was given a sequence of 25 trials; in each trial, a light was switched on for ten seconds and then an electric shock was applied to the metal cage in which the dog was sitting. In each trial, the dog had an opportunity, once the light went on, to jump into an adjoining cage and thus avoid the shock. In the initial trial, all the dogs received the shock (since they did not know the meaning of the signal), and in the succeeding 24 trials they learned to avoid it. The left side of Figure 24.1 displays the experimental data for the 30 dogs, ordered by the time of the last trial in which they were shocked. (This ordering has nothing to do with the order in which the experiment was performed on the dogs; we choose it simply to make the between-dog variation in the data more visible.) Interest lies in the factors that affected the dogs' learning; in particular, did they learn more from successful avoidances than from shocks? Another question is: can the variation in responses among the 30 dogs be explained by a single stochastic learning model, or is there evidence in the data for underlying between-dog variation?

We choose this example to study model-checking methods because the data and the associated stochastic learning model have an interesting structure, with replications over dogs and the probability of avoidance of an electric shock dependent on previous outcomes. As we shall see, the sequential nature of the model has important implications for some of the predictive distributions used to calibrate the model checks. Specifically, the logistic regression model fits these data reasonably well but has interesting systematic patterns of misfit. These data are also of historical interest because the analysis by Bush and Mosteller in 1955 includes an early example of simulation-based model checking: they compared the observed data to simulations of 30 dogs from their model (with parameters fixed at their maximum likelihood estimates, which was reasonable in this case since they are accurately estimated from this dataset).

Setting up the logistic regression model and estimating its parameters

We use the notation $y_{jt} = 1$ or 0 to indicate a shock or avoidance for trial t on dog j, for $t = 0, \ldots, 24$ and $j = 1, \ldots, 30$. We fit a logistic regression model,

$$\Pr(y_{jt} = 1) = \mathrm{logit}^{-1}(\beta_0 + \beta_1 X_{1jt} + \beta_2 X_{2jt}), \tag{24.1}$$

```
        REAL DOGS                FAKE DOGS (logit model)          FAKE DOGS (log model)

trial number                     trial number                     trial number

0    5   10   15   20  24        0    5   10   15   20  24        0    5   10   15   20  24
SS.S.S...................        ..SS.SSS.................        S.SS.....................
SSSS.S...................        SSSSS.SS.................        SSS.S....................
SSSSSSS..................        S.SSS...S................        SSSSS....................
SSSSSSS..................        SSSS.SS.S................        SS.SS.S..................
SSSSSSSS.................        .SSSS.SSS.S..............        SSS.S.S..................
SSS.S..S.................        S...SS..S.S..............        SSSSSSS..................
SSSS.S.S.................        SSSSSS.SSSS..............        SSSS.S.S.................
SSS..S.SS................        SSS..SSS.S.S.............        SSS.SSS.S................
SSSS.S..S................        SSSSSS..SSSS.............        SSSS..S.S................
SSS.....S................        SSS.S.SS..SS.............        SSSSSSSSS................
S.SSSS.SSS...............        SSSSSSS...SS.............        SSSSSSSS.S...............
SSSSSSS..SS..............        SSSSSSS.SS.SS............        SS...S...S.S.............
SS.S...S..S..............        SSSS.S.S....S............        SSSSSSSS..SS.............
SSSSS.SSSSSS.............        .SSSSSSSS.S.S............        SSSSS.SSS..S.............
SSSS.SS..S.S.............        SSSSSSSS.SSSSS...........        SSS..S.....S.............
SSSSS.....SS.............        SS.SSSSSSSSS.SS..........        SS.S.SSS.S.S.............
SSS.S.S...S.S............        SSS.S.SS..SS....S........        SSSSSS.SSS..S............
SSSS......S.S............        SSSSS...SS.SS.SSS........        SSSS..S.S...S............
SSSSSSS.SSSSSS...........        SSSSSS..S..SS...S........        SSSSS....S...S...........
S..SS....S.S.S...........        SSSSSS.SS.S......S.......        SSSSSSSSSS..SSS..........
SSSSSS....SS.S...........        SSSSSSSS......S..S.......        SSSSSSSSS..S..S..........
S.S.SSS.S....S...........        SSSSSSS..SSSSS...S.......        SS.SSS..S......S.........
SSSS..S.S..S.S...........        .SSSSSSSSS...S....S......        SSSSSSS.S.SS.S..S........
SSSSSS.S....S.S..........        SSSSSSS.S....S....S......        SSSSS.S.S.S.S...S........
SSSSS..S..SS..S.S........        SSSS.SSSS...SS..S.S......        SSSSSSSSSS......S........
SSSSSSSSSS......S........        SSSSS.....S..S......S....        SSSSSS..S...S...SS.......
SS.S.S..........SS.......        SSSSSSS.SSSSSSS..S.SS....        SSSSSSS.S.S.S.SS.S.......
SSSS..SS...S.S.S.S.......        SSSSSSSS.SSS.S.S...S.S...        SSSSSS.S.S.........S.....
SSSSSSSS...S.SSS..S......        SS.SSSS..S.............S.        SSSSSSSSSSS.SS.....S.....
SSSSS.S.S..S.SSS.....S..S        SSSSSS.SSS...S.........S.        SSSSS.....S.............S
```

Figure 24.1 *On the left, sequence of shocks ("S") and avoidances (".") for 25 trials on each of 30 dogs. The dogs here are ordered by the time of the last shock, with ties broken randomly. In the middle and right, similar displays for 30 dogs simulated from the (classical) logistic and logarithmic regression models conditional on the estimated parameters for each model. See also Figure 24.5.*

where

$$
\begin{aligned}
X_{1jt} &= \sum_{k=0}^{t-1}(1 - y_{jk}) = \text{number of previous avoidances} \\
X_{2jt} &= \sum_{k=0}^{t-1} y_{jk} = \text{number of previous shocks.} \qquad (24.2)
\end{aligned}
$$

Upon reflection, we can realize that this model is not ideal for these data. In particular, the experiment is designed so that all dogs get shocked on trial 0 (which in fact happens, as is shown in the leftmost column of Figure 24.1), whereas the logistic regression model is structured to always have a nonzero probability of both outcomes. This problem could be addressed in many ways, for example, by fitting a logarithmic instead of a logistic link or simply by fitting the model to the data excluding trial 0.

To start with, however, we fit the logistic regression model to the entire dataset and examine what aspects of model misfit are uncovered by various predictive checks. We believe this is an interesting question because it is standard practice to fit a logistic regression model to binary data without seriously considering its appropriateness. (One reason for this is that for many problems there is no clearly preferable alternative to logistic regression, and simple tricks like changing the link function or discarding noninformative data are not generally sufficient to allow a simple model to fit.) In these cases of routine use, we would like to have routine model checks (by analogy to residual plots in normal regressions) that would give the user some idea of the model's problems. The goal of such methods is not to "accept" or "reject" a model, but rather to highlight important areas where it does not fit the data.

The model is easy to fit in Bugs, and the posterior medians of β_0, β_1, and β_2 are 1.80, -0.35, and -0.21, respectively. The negative coefficients β_1, β_2 imply that the probability of shock declines after either an avoidance or a shock, with $|\beta_1| > |\beta_2|$ implying that avoidances have a larger effect.

We can write the Bugs model in three parts: first, calculation of the number of previous shocks and avoidances recursively in terms of `y[j,t]`, the indicator for a shock of dog j at time t; second, the classical logistic regression model; and, third, the noninformative prior distributions: For convenience, we enclose both of the first two parts in the same `for (j in 1:n.dogs)` loop:

Bugs code

```
model {
  for (j in 1:n.dogs){
    n.avoid[j,1] <- 0
    n.shock[j,1] <- 0
    for (t in 2:n.trials){
      n.avoid[j,t] <- n.avoid[j,t-1] + 1 - y[j,t-1]
      n.shock[j,t] <- n.shock[j,t-1] + y[j,t-1]
    }
    for (t in 1:n.trials){
      y[j,t] ~ dbin (p[j,t], 1)
      logit(p[j,t]) <- b.0 + b.1*n.avoid[j,t] + b.2*n.shock[j,t]
    }
  }
  b.0 ~ dnorm (0, .0001)
  b.1 ~ dnorm (0, .0001)
  b.2 ~ dnorm (0, .0001)
}
```

Defining predictive replications for the dog example

To perform model checks, the data must be compared to a reference distribution of possible replicated datasets. In a usual logistic regression model, this would be performed by fixing the matrix X of predictors and then, for each simulated parameter vector β^l, drawing the 25×30 responses $y_{jt}^{\text{rep}\,(s)}$ independently,

$$\Pr(y_{jt}^{\text{rep}\,(s)}) = \text{logit}^{-1}(X_{jt}\beta^{(s)}), \tag{24.3}$$

to yield a simulated dataset $y^{\text{rep}\,(s)}$. (The notation X_{jt} indicates the vector of predictors, $(1, X_{1jt}, X_{2jt})$ defined in (24.2).) Computing this for n_{sims} parameter vectors yields n_{sims} simulated datasets.

A stochastic learning model is more complicated, however, because the predictor variables X depend on previous outcomes y. Simulation of replicated data for a new dog must thus be performed sequentially. For each simulated parameter vector $\beta^{(s)}$:

- For each dog, $j = 1, \ldots, 30$:
  - For trial $t = 0, \ldots, 24$:

1. Compute the vector of predictors, $X_{jt}^{\text{rep}\,(s)}$, based on the previous t trials for dog j.
2. Simulate $y_{jt}^{\text{rep}\,(s)}$ as in (24.3).

Predictive replications in Bugs. We can create the simulations in Bugs or directly in R. The Bugs model has two parts: recursive definition of the number of shocks/avoidances, and the probability model for the individual outcomes.

Bugs code
```
for (j in 1:n.dogs){
  n.avoid.rep[j,1] <- 0
  n.shock.rep[j,1] <- 0
  for (t in 2:n.trials){
    n.avoid.rep[j,t] <- n.avoid.rep[j,t-1] + 1 - y.rep[j,t-1]
    n.shock.rep[j,t] <- n.shock.rep[j,t-1] + y.rep[j,t-1]
  }
  for (t in 1:n.trials){
    y.rep[j,t] ~ dbin (p.rep[j,t], 1)
    logit(p.rep[j,t]) <- b.0+b.1*n.avoid.rep[j,t]+b.2*n.shock.rep[j,t]
  }
}
```

Once this is set up in Bugs, we can save y^{rep} along with everything else. The only difficulty is that we are now saving 750 additional parameters, and if we save thousands of iterations, we will run into computer storage problems. (Nowadays, 750,000 bytes are nothing on the computer, but in R, computations with a matrix of this size can be slow.) We set up and call the Bugs model as follows, using the `n.thin` option in Bugs to save only every 100^{th} iteration:

R code
```
n.dogs <- nrow(y)
n.trials <- ncol(y)
data <- list ("y", "n.dogs", "n.trials")
inits <- function (){
  list (b.0=rnorm(1), b.1=rnorm(1), b.2=rnorm(1))
}
parameters <- c ("b.0", "b.1", "b.2", "y.rep", "y.mean.rep")
fit.logit.1 <- bugs (data, inits, parameters, "dogs.logit.1.bug",
  n.chains=3, n.iter=2000, n.thin=100)
```

With 3 chains of 2000 iterations each, saving every 100^{th} iteration of the last half of each chain, we are left with 30 simulation draws, which are enough for reasonable estimates, standard errors, and predictive checks.

Predictive replications in R. Alternatively, we can write an R function to simulate predictive replications, given the simulations of $\beta_0, \beta_1, \beta_2$ from the Bugs model. We first set up an empty array for the n_{sims} replicated datasets and then fill it up, one dog at a time and one trial at a time. All the computations are done in vector form, simulating all n_{sims} random replications at once.

R code
```
y.rep <- array (NA, c(n.sims, n.dogs, n.trials))
for (j in 1:n.dogs){
  n.avoid.rep <- rep (0, n.sims)
  n.shock.rep <- rep (0, n.sims)
  for (t in 1:n.trials){
    p.rep <- invlogit (b.0 + b.1*n.avoid.rep + b.2*n.shock.rep)
    y.rep[,j,t] <- rbinom (n.sims, 1, p.rep)
    n.avoid.rep <- n.shock.rep + 1 - y.rep[,j,t]
    n.shock.rep <- n.shock.rep + y.rep[,j,t]
  }
}
```

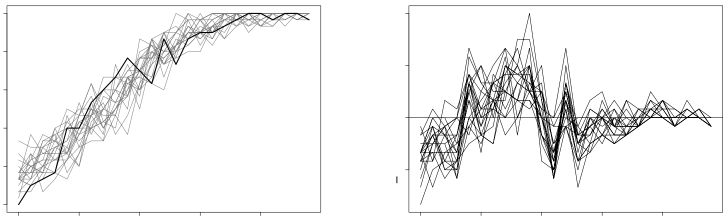

Figure 24.2 *(a) The proportion of avoidances among the dogs in the shock-avoidance experiment, as a function of the trial number. The solid line shows the data, and the light lines represent 20 simulated replications from the model. This plot can be seen as a predictive check of a vector test statistic, $T(y)$, compared to replications $T(y^{\rm rep})$.*
(b) Plots of $T(y^{\rm rep}) - T(y)$ for the 20 simulations of $y^{\rm rep}$. The systematic differences from the horizontal line represent aspects of the data that are not captured by the model.

Direct comparison of simulated to real data

The most basic predictive check is a visual comparison of the observed data to a replication under the assumed model. Figure 24.1 shows the observed data (in the left part of the figure), along with a single replicated dataset, $y^{\rm rep}$ (in the center part; ignore the right part of the figure for now). The visual display shows some interesting differences between the real and simulated dogs.

The visual comparison is aided by ordering the 30 dogs in each dataset in order of the time of their last shock. We program this as a function in R:

R code

```
dogsort <- function (y){
  n.dogs <- nrow(y)
  n.trials <- ncol(y)
  last.shock <- rep (NA, n.dogs)
  for (j in 1:n.dogs){
    last.shock[j] <- max ((1:n.trials)[y[j,]==1])}
  y[order(last.shock),]
}
```

To make the left and center displays in Figure 24.1, we then simply print `y` and `y.rep[1,,]`, with the latter being the first of the $n_{\rm sims}$ random replications of the data. (Since the simulations from the Bugs output are randomly ordered before being returned to R, the first simulated replication matrix is as good as any other.)

Strictly speaking, a posterior predictive check should compare y to several draws of $y^{\rm rep}$, but in this case a single draw is informative because of the internal replication of 30 independent dogs in a single dataset.

More focused model checks

The graphical comparison in Figure 24.1 can be used to suggest more focused diagnostics. For example, the replicated dogs appear to have too few shocks in the early trials, compared to the real dogs. This is substantively relevant because the purpose of the model is to understand the learning behavior of the dogs. To check this pattern more formally, we display in Figure 24.2a the proportion of avoidances among the 30 dogs (that is, $1-\bar{y}_{.t}$) versus time t. Overlain on the graph are the corresponding time series for 20 random draws $y^{{\rm rep}\,(s)}$ from the predictive distribution under the estimated model. Compared to the data, the model predicts too many avoidances in the first two trials and too slow an improvement in the first five trials. The model thus does not capture the rate of learning at the beginning of the experiment.

We programmed this model check in R by first writing a function to generate the test variable (in this case, the mean number of avoidances over time) and then displaying it for the replications and observed data:

R code
```
test <- function (data){
  colMeans (1-data)
}
plot (c(0,n.trials-1), c(0,1), xlab="Time",
  ylab="Proportion of avoidances", type="n")
mtext ("data and replicated data\n(logit link)", 3)
for (s in 1:20){
  lines (0:(n.trials-1), test (y.sim[s,,]), lwd=.5, col="gray")
}
lines (0:(n.trials-1), test (y), lwd=3)
```

To sharpen the contrast between replications and data, we display in Figure 24.2b the difference between the replicated and observed proportions of avoidances over time. In R, we first create the function for the difference of the test variables,

R code
```
test.diff <- function (data, data.rep){
  test (data) - test (data.rep)
}
```

then determine the range of these differences (to use in scaling the plot),

R code
```
diff.range <- NULL
for (s in 1:20){
  diff.range <- range (diff.range, test.diff (y, y.rep[s,,]))
}
```

and then set up the plot, graph the differences between the data and each of 20 simulations, and graph the zero line, which shows the standard of comparison.

R code
```
plot (c(0,24), diff.range, xlab="Time", ylab="Proportion of avoidances",
  type="n")
mtext ("data minus replicated data\n(logit link)", 3)
for (s in 1:20){
  lines (0:(n.trials-1), test.diff (y, y.sim[s,,]), lwd=.5, col="gray")
}
abline (0, 0, lwd=3)
```

Figure 24.2b shows the data to be consistently lower than the simulations at some points and higher at others, indicating a statistically significant discrepancy between model and data.

Numerical test statistics. For another example, Figure 24.3 displays predictive checks for two simple test statistics: the mean and standard deviation of the number of shocks per dog. In each plot, the observed value of $T(y)$ is shown as a vertical bar in a histogram representing 1000 draws of $T(y^{\text{rep}})$ from the posterior distribution. From Figure 24.3a, we see that the mean number of shocks is fit well by the model. Figure 24.3b shows that the observed standard deviation is a bit higher than expected under the model, but the discrepancy is not statistically significant; that is, we could expect to see such discrepancies occasionally just by chance, even if the model were correct.

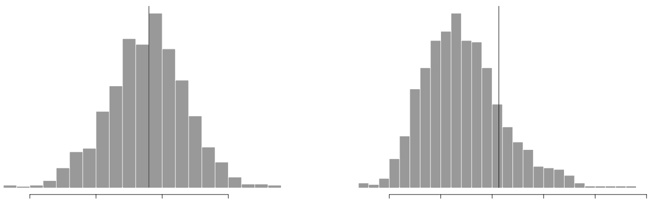

Figure 24.3 *Predictive checks for the (a) mean and (b) standard deviation of the number of shocks among the 30 dogs. The vertical bars indicate the observed values of the test statistics $T(y)$, and the histograms display $T(y^{\rm rep})$ from 1000 draws of $y^{\rm rep}$ under the logistic model.*

Thus, these two aspects of the data are fit reasonably well; however, the systematic problem we have found in the early trials indicates a problem with the model.

Fitting and checking a logarithmic regression model

We now move to a more reasonable logarithmic regression model for the same data (which was in fact fit by the psychologists who performed the early data analyses):

$$\Pr(y_{jt} = 1) = \exp(\beta_1 X_{1jt} + \beta_2 X_{2jt}), \qquad (24.4)$$

with X_{1jt} and X_{2jt} the number of previous avoidances and shocks, respectively, as defined in (24.2). Unlike the logistic model (24.1), this model has no constant term because the probability of shock is fixed at 1 at the beginning of the experiment. In addition, β_1 and β_2 are restricted to be negative.

The Bugs model is similar to the logistic regression on page 517 except with the `logit(p[j,t])` line changed to:

```
log(p[j,t]) <- b.1*n.avoid[j,t] + b.2*n.shock[j,t]
```
Bugs code

The log model omits the intercept term, b_0, so that the probability of a shock is 1 for the first trial. In addition, the coefficients β_1, β_2 must be constrained to be negative, so we give them the following noninformative distributions:

```
b.1 ~ dunif (-100, 0)
b.2 ~ dunif (-100, 0)
```
Bugs code

The median estimates of the parameters β_1 and β_2 in the logarithmic model are -0.24 and -0.08, with standard errors of 0.02 and 0.01, respectively. The coefficient for avoidances, β_1, is estimated to be more negative than β_2, indicating that avoidances have a larger effect than shocks in reducing the probability of future shocks. Transforming back to the probability scale, the median estimates for $(e^{\beta_1}, e^{\beta_2})$ are $(0.79, 0.92)$, indicating that an avoidance or a shock multiplies the predicted probability of shock by an estimated factor of 0.79 or 0.92, respectively.

Having fit this improved model, we check its fit using predictive replications, which we simulate from the model as described earlier in this section (except using the logarithmic rather than the logistic link). A single random draw from the predictive distribution of 30 new dogs is displayed on the right side of Figure 24.1. A check of the average number of avoidances over time—Figure 24.4, replicating Figure 24.2—shows no apparent discrepancies with this aspect of the data: the logarithmic link has fixed the problem with the early trials.

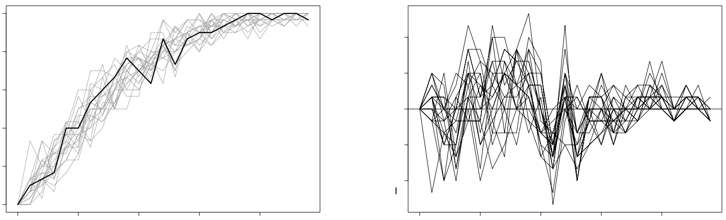

Figure 24.4 *Replication of Figure 24.2 for the logarithmic regression model. Under this estimated model, the simulated patterns of the rate of avoidances over time fit reasonably well to the data.*

Fitting and checking a multilevel model with no additional learning from avoidances

The logarithmic regression model fits the data reasonably well, but a key issue not addressed so far is separating between-dog variability from stochastic learning. The data from the 30 dogs vary quite a bit. In the stochastic learning model this is explained by the fact that dogs learn more from avoidances than from shocks (that is, $\beta_1 < \beta_2 < 0$) so that a dog that gets lucky at the beginning of the experiment is likely to perform well throughout, even in the absence of any real differences between dogs. However, one may consider an alternative explanation that the dogs indeed differ and, perhaps, there may be no additional learning associated with avoidances.

These two hypotheses—extra learning from avoidances, or between-dog variability—can be distinguished by a multilevel model of the form (24.4) but with parameters that vary by dog:

$$\Pr(y_{jt} = 1) = \exp(\beta_{1j} X_{1jt} + \beta_{2j} X_{2jt}), \tag{24.5}$$

with the parameters β_1, β_2 both constrained to be negative for each dog (to ensure that $\Pr(y_{jt} = 1)$ is always less than 1).

We can easily fit this model all at once, but we fit it in stages in order to understand the workings of each part. The model we have already fit corresponds to extra learning from avoidances. The other extreme is to allow no extra learning from avoidances but to allow dogs to vary in their ability to perform the task. This model could be written as

$$\Pr(y_{jt} = 1) = \exp(\beta_j (X_{1jt} + X_{2jt})), \tag{24.6}$$

where $X_{1jt} + X_{2jt}$ is the number of previous trials, or simply $t - 1$.

In Bugs, we write the model as before, but applying the same coefficient β to both shocks and avoidances, and allowing it to vary by dog. The parameters β_j are restricted to be negative (so that the probabilities p_{jt} fall below 1). To form a multilevel model, we work with the negative of the β_j's, which we assign a log-normal distribution (that is, the values of $\log(-\beta_j)$'s are modeled with a normal distribution) with hyperparameters μ_β and σ_β. Here is the rewritten fragment of the Bugs model:

Bugs code
```
      log(p[j,t]) <- b[j]*(n.avoid[j,t]+n.shock[j,t])
    }
```

```
        REAL DOGS                FAKE (simple multilevel model)   FAKE (full multilevel model)

trial number                     trial number                     trial number

0    5    10   15   20 24        0    5    10   15   20 24        0    5    10   15   20 24
SS.S.S...................        S.S......................        SS.S.....................
SSSS.S...................        SSSSS....................        SSSS.....................
SSSSSSS..................        SSSSS....................        SSSSS....................
SSSSSSS..................        S.S...S..................        SSSSS....................
SSSSSSSS.................        SSS.S..S.................        SSS.S....................
SSS.S..S.................        SSSSS..S.................        SSS.S.S..................
SSSS.S.S.................        SSSSS.SS.................        SSS.SSS..................
SSS..S.SS................        SSSSSS..S................        SSSSSS.S.................
SSSS.S..S................        S.SS.....S...............        SSS...S.S................
SSS.....S................        SS.SSSS...S..............        SSS.....S................
S.SSSS.SSS...............        SS.SS.S.S..S.............        SSSSSS...S...............
SSSSSSS..SS..............        SS..S.S....S.............        SSSSSSSSSS...............
SS.S...S..S..............        SSSS.......S.............        S.SSSS..S.S..............
SSSSS.SSSSSS.............        SSSSS.S.SS..S............        SSS.S..S..S..............
SSSS.SS..S.S.............        SSS....S.S...S...........        SSS........S.............
SSSSS.....SS.............        SSS.S.SS.S.S.S...........        SSSSS.SS.SSS.............
SSS.S.S...S.S............        SS..SS...S.....S.........        SSS.SSS.....S............
SSSS......S.S............        SSS.SS.S.......S.........        SSSSSSSSSSS.S............
SSSSSSS.SSSSSS...........        S.S...S.SSS....S.........        SSSSSS.......S...........
S..SS....S.S.S...........        SS..S....S......S........        SSS..SSSS.....S..........
SSSSSS....SS.S...........        SS..SSS.SS......S........        SSSSS..S.S.S..S..........
S.S.SSS.S....S...........        SSSSSS.SS..SS..SS........        SSSS.SSSSS.S....S........
SSSS..S.S..S.S...........        SS.SS..S...SS.S..S.......        SSSSSSSSS.SS.SSSS........
SSSSSS.S....S.S..........        SSS.S.......S....S.......        SSS.....S.....SS.S.......
SSSSS..S..SS..S.S........        SSSS.....S..S....S.......        SSS..SS.S.S.S....S.......
SSSSSSSSSS......S........        SSSS..SSSS.S......S......        SSSS.SS.S........S.......
SS.S.S..........SS.......        SSSSS.....SSS....S.S.....        SSSSS.S...........S......
SSSS..SS...S.S.S.S.......        SSSS..S..S...........S...        SSSSSSS.S.S..S.....S.....
SSSSSSSS...S.SSS..S......        SSS.SS................S..        SSSS................S....
SSSSS.S.S..S.SSS.....S..S        S.SSS..S..........S....S.        SSSSSSSS.........S.SS....
```

Figure 24.5 *On the left, sequence of shocks ("S") and avoidances (".") for 25 trials on each of 30 dogs, copied from the left panel of Figure 24.1. The dogs here are ordered by the time of the last shock, with ties broken randomly. In the middle and right, similar displays for 30 dogs simulated from the multilevel model, with and without no extra learning from avoidances. Compare to Figure 24.1 on page 516.*

```
    b[i] <- -b.neg[i]
    b.neg[i] ~ dlnorm (mu.b, tau.b)
  }
  mu.b ~ dnorm (0, .0001)
  tau.b <- pow(sigma.b, -2)
  sigma.b ~ dunif (0, 100)
```

When this model is fit to the dog data, the average of the 30 estimated values of e^{β_j} is 0.86—thus, the probability of a shock is multiplied by about 0.86 after each trial. This estimate is broadly consistent with our previous results: it falls between the estimates of e^{β_1} and e^{β_2} in the classical logarithmic regression model with differential learning. The standard deviation of the 30 dog parameters e^{β_j} in the multilevel model is estimated to be 0.04, indicating a fairly small range of variation in the "abilities" of the dogs.

The left and middle panels of Figure 24.5 show the data y (repeated from the left panel of Figure 24.1) and a simulated replicated dataset y^{rep} from the multilevel model (24.6) with no differential learning. The model clearly does not fit the data—the actual data show many cases of a dense series of shocks with no further avoidances, whereas the fake data shows many dogs with a long series of avoidances followed by a single late shock.

Fitting and checking the full multilevel model

Thus, it appears that the variation between dogs in the data is not simply explained by varying abilities; rather, there is evidence that the experience of avoidances introduces additional learning. We combine the features of between-dog variation and differential learning with the multilevel model (24.5), in which both the learning-after-avoidance and learning-after-shock parameters vary by dog. The regression part of the Bugs model becomes

Bugs code
```
log(p[j,t]) <- b.1[j]*n.avoid[j,t] + b.2[j]*n.shock[j,t])
```

with a normal distribution for $(\log(-\beta_{1j}), \log(-\beta_{2j}))$, a bivariate version of the lognormal distribution in the Bugs model on page 522.

The average of the 30 estimated values of $e^{\beta_{1j}}$ and $e^{\beta_{2j}}$ are 0.78 and 0.91 (with estimated between-dog standard deviations of 0.05 and 0.03, respectively). Hence, as with the non-multilevel log model, the probability of a shock is multiplied by about 0.8 following each avoidance and 0.9 following each shock, with relatively little between-dog variation in the parameters.

The group pooling factor (21.14) from page 479 is 0.73 for the parameters β_{1j} and 0.77 for the parameters β_{2j}—thus, the estimates are shrunk approximately three-fourths toward the group-level model, or to put it another way, the estimates are close to complete pooling.

Having fit the model, we can as usual check its fit by displaying a replicated dataset, which is displayed in the rightmost panel of Figure 24.5. The replications look generally comparable to the real dogs, and, as with Figure 24.4, plots of average avoidances over time show no systematic discrepancies between model and data. Given the relatively small size of this dataset, it is difficult to say much more about this comparison.

Review of example

We have fit several models to the dog data and found that the variation between the response patterns of the dogs can largely be explained by learning more from avoidances than from shocks. This path-dependence or positive-feedback pattern implies that when a dog gets lucky and avoids the shock early, learning, and hence further avoidances, will proceed faster. There is evidence for a small amount of between-dog variation, but less than the difference between the effects of shocks and avoidances.

We used predictive simulations to check the fit of the different models to data, using raw data displays and also summaries of the rate of avoidances over time. The predictive checks do not automatically lead us to an appropriate model and will not necessarily "reject" a poorly fitting model—for example, the mean and standard-deviation summaries in Figure 24.3 on page 521 do not point out the serious failings of the logistic model. Rather, predictive checking can be a useful tool in revealing aspects of disagreement between model and data, which can motivate a search for more reasonable models.

24.3 Model comparison and deviance

When fitting several models to the same dataset, it can be helpful to compare them using summary measures of fit. A standard summary that is programmed in Bugs is the *deviance*, which is -2 times the log-likelihood; that is, -2 times the logarithm of the probability of the data given the estimated model parameters.

Deviance and AIC in classical generalized linear models

In classical generalized linear models, adding a parameter to a model is expected to increase the fit—even if the new parameter represents pure noise. Adding a noise predictor is expected to reduce the deviance by 1, and adding k predictors that are pure noise is expected to reduce the deviance by k. More precisely, adding k noise predictors will reduce the deviance by an amount corresponding to a χ^2 distribution with k degrees of freedom.

Thus, if k predictors are added and the deviance declines by significantly *more* than k, then we can conclude that the observed improvement in predictive power is statistically significant. Thus,

$$\text{adjusted deviance} = \text{deviance} + \text{number of predictors} \tag{24.7}$$

can be used as an adjusted measure that approximately accounts for the increase in fit attained simply by adding predictors to a model. (The analogy in simple linear regression is the adjusted R^2.)

The next step, beyond checking whether an improvement in deviance is statistically significant, is to see if it is estimated to increase out-of-sample predictive power. On average, a predictor needs to reduce the deviance by 2 in order to improve the fit to new data. The *Akaike information criterion* is defined as

$$\begin{aligned} \text{AIC} &= \text{deviance} + 2 \cdot (\text{number of predictors}) \\ &= \text{adjusted deviance} + \text{number of predictors}. \end{aligned} \tag{24.8}$$

In classical regression or generalized linear modeling, a new model is estimated to reduce out-of-sample prediction error if the AIC decreases.

Deviance and DIC in multilevel models

The ideas of deviance and AIC apply to multilevel models also, but with the difficulty that the "number of parameters" is not so clearly defined. Roughly speaking, the number of parameters in a multilevel model depends on the amount of pooling—a batch of J parameters corresponds to one parameter if there is complete pooling, J independent parameters if there is no pooling, and something in between with partial pooling. For example, with the varying-intercept radon models, the coefficients for the 85 county indicators represent something fewer than 85 independent parameters. Especially for the counties with small sample sizes, the group-level regression explains much of the variation in the intercepts, so that in the multilevel model they are not estimated independently. When the model is improved and the group-level variance decreases, the effective number of independent parameters also decreases.

In multilevel models, the *mean deviance* (that is, the deviance averaged over all the n_{sims} simulated parameter vectors) plays the role of the adjusted deviance (24.7). The effective number of parameters is called p_D, and the measure of out-of-sample predictive error is the *deviance information criterion*,

$$\text{DIC} = \text{mean deviance} + 2p_D, \tag{24.9}$$

which plays the role of the Akaike information criterion in (24.8). We shall not further discuss the computation of p_D here, except to say that it is unstable to estimate—even from Bugs simulations that have otherwise converged—and so we currently use it only in a provisional sense.

We illustrate the use of deviance and DIC by comparing the fit of the models fit to the dog data in Section 24.2. Figure 24.6 shows the mean deviance, effective

Model	Mean deviance	Effective # params, p_D	DIC
Classical logistic	570	4	574
Classical logarithmic	550	5	555
Simple multilevel	544	21	565
Full multilevel	533	29	562

Figure 24.6 *Average deviance, estimated effective number of parameters, and deviance information criterion (DIC) for each of four models fit to the dog data in Section 24.2. Model fit improves as we go down the table, as can be seen from the decreasing values of mean deviance. However, the effective number of parameters increases for the larger models. The best model for out-of-sample predictions, as measured by DIC, is the classical logarithmic model.*
We would still prefer the full multilevel model here, since we expect the dogs to vary in their underlying parameters. But the improvement in fit, compared to the classical logarithmic regression, is not estimated to result in more accurate predictions.

number of parameters, and DIC for each. As the models get more complicated, the mean deviance decreases, which makes sense—with more structure, we can fit the data better. The largest jump is from the logit to the log model, which makes sense, since as we saw in Section 24.2, the logarithmic link fit the data much better.

However, the improvement in fit when moving to the multilevel models is counterbalanced by the increase in p_D, the effective number of parameters. As a result, the estimated out-of-sample prediction error, DIC, actually increases slightly for these models. The multilevel models do fit the data better, an improvement greater than would be expected by chance—as we can see from the mean deviances—but the DIC values suggest that they would not actually do as well in predicting for new dogs.

In summary, we are *not* saying that the classical logarithmic model is "best" here. We prefer the full multilevel model as a more complete description of the data. However, the DIC results are interesting in suggesting that further improvement is possible, perhaps constraining the β_{1j} and β_{2j} parameters more than is done by the multilevel model we have fit so far.

Finally, the values of p_D in Figure 24.6 illustrate the instability of the estimates of this quantity: the classical models are estimated to have 4 and 5 "effective parameters" each, even though they only have 3 and 2 parameters, respectively. The multilevel models are estimated to have 21 (out of a possible 30) and 29 (out of a possible 60) parameters each, even though we have seen that their pooling factors are close to 1 (that is, the models are close to complete pooling). So we treat these p_D and DIC results as suggestive rather than definitive.

24.4 Bibliographic note

The simulation-based model checking approach presented in Chapter 8 and here is based on ideas of Box (1980), Rubin (1984), Gelman, Meng, and Stern (1996), Gelman et al. (2003, chapter 6), and Gelman (2003); see also Stone (1974) and Gelfand, Dey, and Chang (1992) for related ideas. Sinharay and Stern (2003) apply predictive checks to hierarchical models. See Cook and Weisberg (1999) for a comprehensive overview of classical regression diagnostics. Pardoe (2001) and Pardoe and Cook (2002) extend these ideas to simulation-based inference.

The logarithmic model for the dog data comes from Bush and Mosteller (1955), who also performed simulation-based model checking. Further analyses of these data were considered by Sternberg (1963) and Gelman et al. (2000).

AIC comes from Akaike (1973) and is related to C_p (Mallows, 1973); DIC comes from Spiegelhalter et al. (2002); see also Hodges and Sargent (2001), Vaida and Blanchard (2002), and Spiegelhalter (2006) for related work and discussion.

24.5 Exercises

1. Download the data in the folder `dogs` and fit some other models, for example using as a predictor the result from the previous trial, or the previous two trials, rather than the total number of shocks and avoidances.
   (a) Fit this model, as usual building up from simpler versions (first a single-level model, then varying intercepts, then varying slopes, then adding other predictors as appropriate). Plot the data and fitted model to make sure that your model makes sense.
   (b) Use Bugs to simulate replicated datasets from your model, and make various plots to compare the replicated with the actual data.
2. Model checking with non-nested levels: the folder `supreme.court` contains data regarding U.S. Supreme Court votes for all justices across several issues.
   (a) Fit an ideal-point model to these data (see Section 14.3) and then use replicated data and graphical displays to check the model fit. (This is a huge dataset, and so the model will certainly not fit in many ways. Your goal here is not simply to "reject" the model but rather to understand the ways in which it does not fit.)
   (b) How might you expand the model to fix these problems?
3. Model checking for multilevel logistic regression:
   (a) Do some simulation-based graphical checking for the logistic regression model that you fit in Exercises 14.5–14.6 to the data from the speed-dating experiment.
   (b) How might you expand the model to fix the problems you have found?
4. Model checking for ordered categorical regression:
   (a) Do some simulation-based graphical checking for the ordered logistic regression model that you fit in Exercise 17.11 to the data from the storable-voting experiment.
   (b) How might you expand the model to fix the problems you have found?