CHAPTER

18

Binomial Analysis of Covariance

OUTLINE

18.1 INTRODUCTION

We can specify a binomial analysis of covariance (ANCOVA) by adding discrete and continuous covariates to the linear predictor of a binomial generalized linear model (GLM). Once again, to stress the structural similarity with the normal linear model in Chapter 11, we modify the asp viper example just slightly. Instead of modeling a continuous measurement such as body mass in Chapter 11, we model a count governed by an underlying probability; specifically, we model the proportion of black individuals in adder populations. The adder has an all-black and a zigzag morph, where females are brown and males are gray (Fig. 18.1).

It has been hypothesized that the black color confers a thermal advantage, and therefore, the proportion of black individuals should be greater in cooler or wetter habitats. We simulate data that bear on this question and "study," by simulation, 10 adder populations each in the Jura mountains, the Black Forest, and the Alps. We capture a number of snakes in these populations and record the proportion of black adders. Then, we relate these proportions to the mountain range and to a combined index of low

FIGURE 18.1 Male adder (*Vipera berus*) of the zigzag morph, Germany, 2007. (*Photo T. Ott*)

temperature, wetness, and northerliness of the site. Our expectation will of course be that there are relatively more black adders at cool and wet sites. As always, a count is the result of a true number (here, of black and zigzag adders) and a detection probability. Hence, in the following analyses, we make the implicit assumption that the detectability of black and zigzag adders neither differs between each other nor among populations.

We model the number of black adders C_i among N_i captured animals in population i. Here is the description of the model.

1. Distribution: $C_i \sim \text{Binomial}\,(p_i, N_i)$
2. Link function: logit, i.e., $\text{logit}(p_i) = \log\left(\frac{p_i}{1-p_i}\right)$ = linear predictor
3. Linear predictor: $\alpha_{\text{Jura}} + \beta_1 * x_{\text{BlackF}} + \beta_2 * x_{\text{Alps}} + \beta_3 * x_{\text{wet}} + \beta_4 * x_{\text{wet}} * x_{\text{BlackF}} + \beta_5 * x_{\text{wet}} * x_{\text{Alps}}$

Note that the number of animals captured, N_i, is not a parameter of the binomial distribution in this example but is known (in contrast to the model of Chapter 21 where N_i will be estimated). The link function is the logit, as is customary for a binomial distribution, though other links are possible (e.g., the complementary log–log, which is asymmetrical; see GLM textbooks such as McCullagh and Nelder, 1989). Finally, the linear predictor is made up of an intercept α_{Jura} for the expected proportion, on the logit scale, of black adders in the Jura and parameters β_1 and β_2 for the difference in the intercept between Black Forest and the Jura and the Alps and the Jura, respectively. The parameters β_3, β_4, and β_5 specify the slope of the (logit-linear) relationship between the proportion of black adders and the wetness indicator of a site in the Jura and the difference from β_3 of these slopes in the Black Forest and the Alps, respectively.

18.2 DATA GENERATION

```
n.groups <- 3
n.sample <- 10
n <- n.groups * n.sample
x <- rep(1:n.groups, rep(n.sample, n.groups))
pop <- factor(x, labels = c("Jura", "Black Forest", "Alps"))
```

We construct a continuous wetness index: 0 denotes wet sites lacking sun and 1 is the converse. For ease of presentation, we sort this covariate; this has no effect on the analysis.

```
wetness.Jura <- sort(runif(n.sample, 0, 1))
wetness.BlackF <- sort(runif(n.sample, 0, 1))
wetness.Alps <- sort(runif(n.sample, 0, 1))
wetness <- c(wetness.Jura, wetness.BlackF, wetness.Alps)
```

We also need the number of adders examined in each population (N_i), i.e., the binomial totals, also called sample or trial size of the binomial distribution. We assume that the total number of snakes examined in each population is a random variable drawn from a uniform distribution, but this is not essential.

```
N <- round(runif(n, 10, 50) )        # Get discrete Uniform values
```

We build the design matrix of an interactive combination of population and wetness.

```
Xmat <- model.matrix(~ pop*wetness)
print(Xmat, dig = 2)
```

Select the parameter values, i.e., choose values for α_{Jura}, β_1, β_2, β_3, β_4, and β_5.

```
beta.vec <- c(-4, 1, 2, 6, 2, -5)
```

We assemble the number of black adders captured in each population in three steps:

1. We add up all components of the linear model to get the value of the linear predictor,

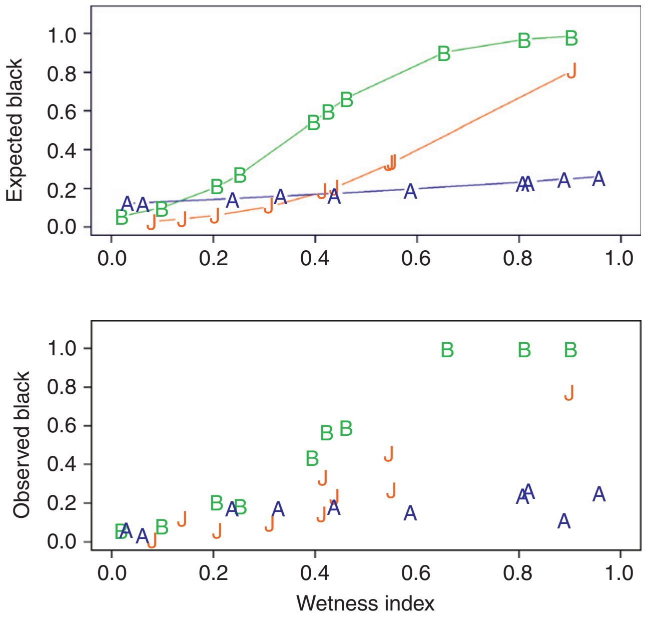

FIGURE 18.2 Relationship between the wetness index and the expected (top) and observed proportion (bottom) of black adders in the Jura mountains (red J), the Black Forest (green B), and the Alps (blue A). The discrepancy between the two graphs is due to binomial sampling variation.

2. We apply the inverse logit transformation to get the expected proportion (p_i) of black adders in each population (Fig. 18.2; top), and finally,
3. We add binomial noise, i.e., use p_i and N_i to draw binomial random numbers representing the count of black adders in each sample of N_i snakes (Fig. 18.2; bottom).

The value of the linear predictor is again obtained by matrix multiplication of the design matrix (`Xmat`) and the parameter vector (`beta.vec`). (For ecological purists the usual reality disclaimer applies again.).

```
lin.pred <- Xmat[,] %*% beta.vec                  # Value of lin.predictor
exp.p <- exp(lin.pred) / (1 + exp(lin.pred))      # Expected proportion

C <- rbinom(n = n, size = N, prob = exp.p)        # Add binomial noise
hist(C)                                           # Inspect what we've created

par(mfrow = c(2,1))
matplot(cbind(wetness[1:10], wetness[11:20], wetness[21:30]), cbind(exp.p[1:10],
exp.p[11:20], exp.p[21:30]), ylab = "Expected black", xlab = "", col =
c("red","green","blue"), pch = c("J","B","A"), lty = "solid", type = "b", las = 1,
cex = 1.2, main = "", lwd = 1)
```

```
matplot(cbind(wetness[1:10], wetness[11:20], wetness[21:30]),
cbind(C[1:10]/N[1:10], C[11:20]/N[11:20], C[21:30]/N[21:30]), ylab = "Observed
black", xlab = "Wetness index", col = c("red","green","blue"), pch =
c("J","B","A"), las = 1, cex = 1.2, main = "")
par(mfrow = c(1,1))
```

Hence, the proportion of black adders increases with wetness most steeply in the Black Forest, less so in the Jura, and hardly at all in the Alps (Fig. 18.2).

18.3 ANALYSIS USING R

As usual, the analysis in R is concise.

```
summary(glm(cbind(C, N-C) ~ pop * wetness, family = binomial))
beta.vec
```

```
> summary(glm(cbind(C, N-C) ~ pop * wetness, family = binomial))

Call:
glm(formula = cbind(C, N-C) ~ pop * wetness, family = binomial)

Deviance Residuals:
    Min         1Q     Median        3Q       Max
-1.8911    -0.5785    0.1309    0.8162    1.8866

Coefficients:
                         Estimate  Std. Error  z value  Pr(>|z|)
(Intercept)               -3.7255     0.3905   -9.541   < 2e-16 ***
popBlack Forest            0.1342     0.5894    0.228   0.81990
popAlps                    1.5118     0.4946    3.057   0.00224 **
wetness                    5.5286     0.7215    7.663  1.81e-14 ***
popBlack Forest:wetness    3.6046     1.3070    2.758   0.00582 **
popAlps:wetness           -4.3748     0.8607   -5.083  3.72e-07 ***
- - -
Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance:  374.63  on 29  degrees of freedom
Residual deviance:   26.41  on 24  degrees of freedom
AIC: 123.75

Number of Fisher Scoring iterations: 4

> print(beta.vec)
[1] -4  1   2  6  2  -5
>
```

Owing to the small sample size, we observe only a moderate correspondence with the input values.

18.4 ANALYSIS USING WinBUGS

In the following WinBUGS code, we add a few lines to compute Pearson residuals. For a binomial response, these can be computed as $(C_i - N_i p_i)/\sqrt{N_i p_i (1 - p_i)}$, where C_i is the binomial count for unit i and N_i and p_i are the trial size and success probability of the associated binomial distributions. The Pearson residual has the form of a raw residual divided by the standard deviation of unit i.

```
# Define model
sink("glm.txt")
cat("
model {

# Priors
 for (i in 1:n.groups){
    alpha[i] ~ dnorm(0, 0.01)      # Intercepts
    beta[i] ~ dnorm(0, 0.01)       # Slopes
 }

# Likelihood
 for (i in 1:n) {
    C[i] ~ dbin(p[i], N[i])
    logit(p[i]) <- alpha[pop[i]] + beta[pop[i]]* wetness[i] # Baseline Jura

# Fit assessments: Pearson residuals and posterior predictive check
   Presi[i] <- (C[i]-N[i]*p[i]) / sqrt(N[i]*p[i]*(1-p[i])) # Pearson resi
   C.new[i] ~ dbin(p[i], N[i])              # Create replicate data set
   Presi.new[i] <- (C.new[i]-N[i]*p[i]) / sqrt(N[i]*p[i]*(1-p[i]))
   D[i] <- pow(Presi[i], 2)                 # Squared Pearson residuals
   D.new[i] <- pow(Presi.new[i], 2)
 }

# Derived quantities
# Recover the effects relative to baseline level (no. 1)
 a.effe2 <- alpha[2] - alpha[1]        # Intercept Black Forest vs. Jura
 a.effe3 <- alpha[3] - alpha[1]        # Intercept Alps vs. Jura
 b.effe2 <- beta[2] - beta[1]          # Slope Black Forest vs. Jura
 b.effe3 <- beta[3] - beta[1]          # Slope Alps vs. Jura

# Custom tests
 test1 <- beta[3] - beta[2]            # Difference slope Alps -Black Forest

# Add up discrepancy measures
 fit <- sum(D[])
 fit.new <- sum(D.new[])
}
",fill=TRUE)
sink()

# Bundle data
win.data <- list(C = C, N = N, pop = as.numeric(pop), n.groups = n.groups, wetness
= wetness, n = n)

# Inits function
inits <- function(){ list(alpha = rlnorm(n.groups, 3, 1), beta = rlnorm(n.groups,
2, 1))}           # Note log-normal inits here

# Parameters to estimate
params <- c("alpha", "beta", "a.effe2", "a.effe3", "b.effe2", "b.effe3", "test1",
"Presi", "fit", "fit.new")

# MCMC settings
ni <- 1500
nb <- 500
nt <- 5
nc <- 3

# Start Gibbs sampler
out <- bugs(data = win.data, inits = inits, parameters.to.save = params, model.file
= "glm.txt", n.thin = nt, n.chains = nc, n.burnin = nb, n.iter = ni, debug = TRUE)
```

The model converges rapidly. To practice good statistical behavior, we now first assess model fit before rushing on to inspect the parameter estimates (or indulge in some activity related to significance testing). After all, only when a model adequately reproduces the salient features in the data set should we believe what it says about the parameters. We first plot the Pearson residuals and, naturally, find no obvious remaining pattern related to order or wetness of a site (Fig. 18.3).

```
par(mfrow = c(1,2), cex = 1.5)
plot(out$mean$Presi, ylab = "Residual", las = 1)
abline(h = 0)
plot(wetness, out$mean$Presi, ylab = "Residual", las = 1)
abline(h = 0)
```

Next, we conduct a posterior predictive check for overall goodness of fit of the model (Fig. 18.4). Remember that our discrepancy measure is the sum of squared Pearson residuals.

```
plot(out$sims.list$fit, out$sims.list$fit.new, main = "", xlab = "Discrepancy
actual data", ylab = "Discrepancy ideal data")
abline(0,1, lwd = 2, col = "black")
```

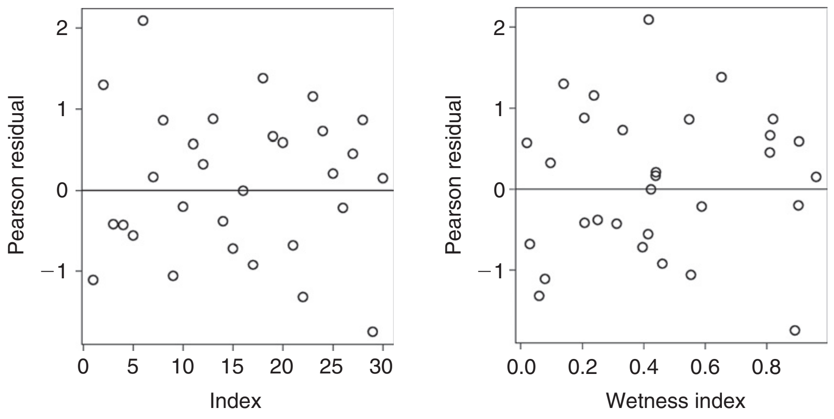

FIGURE 18.3 Pearson residuals plotted against the order of each observation (left) and against the value of the wetness indicator (right).

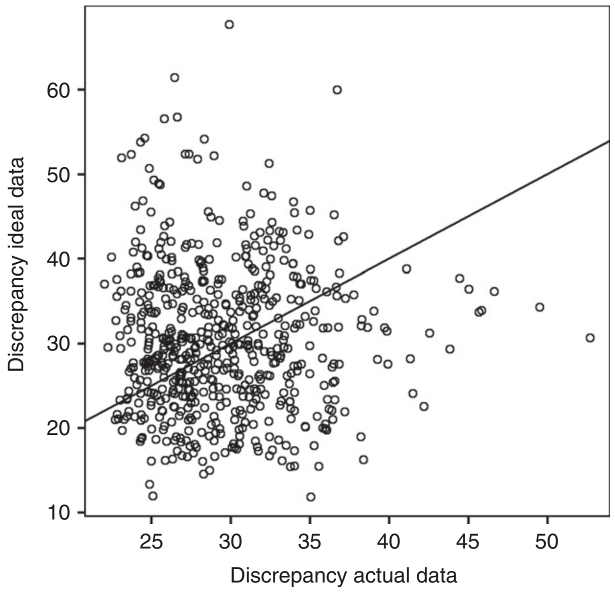

FIGURE 18.4 Posterior predictive check for the black adder analysis. The Bayesian p-value (here, 0.52) is the proportion of points above the line.

Of course, this looks good and so does the Bayesian *p*-value.

```
mean(out$sims.list$fit.new > out$sims.list$fit)
> mean(out$sims.list$fit.new > out$sims.list$fit)
[1] 0.52
```

Hence, we feel justified in comparing the results of the Bayesian analysis with the truth from the data-generating random process and with the frequentist inference from R's function `glm()`.

```
print(out, dig = 3)                                  # Bayesian analysis
beta.vec                                             # Truth in data generation
print(glm(cbind(C, N-C) ~ pop * wetness, family = binomial)$coef, dig = 4)
                                                     # The ML solution
```

We get rather similar estimates. Parameters in the output of the Bayesian analysis printed in boldface correspond to the quantities used for data generation and also to those of the model parameterization in the frequentist analysis in R.

```
> print(out, dig = 3)                               # Bayesian analysis
Inference for Bugs model at "glm.txt", fit using WinBUGS,
 3 chains, each with 1500 iterations (first 500 discarded), n.thin = 5
 n.sims = 600 iterations saved
            mean    sd    2.5%    25%    50%    75%  97.5%  Rhat n.eff
alpha[1]  -3.797 0.383  -4.514 -4.056 -3.782 -3.518 -3.048 1.009   190
alpha[2]  -3.591 0.421  -4.408 -3.896 -3.580 -3.302 -2.825 1.002   600
alpha[3]  -2.225 0.301  -2.837 -2.422 -2.235 -2.021 -1.651 1.031    68
beta[1]    5.655 0.701   4.372  5.164  5.632  6.131  7.004 1.009   200
beta[2]    9.139 1.086   7.148  8.362  9.102  9.927 11.300 1.000   600
beta[3]    1.162 0.477   0.180  0.843  1.192  1.476  2.086 1.034    60
a.effe2    0.207 0.597  -0.972 -0.190  0.201  0.631  1.289 1.012   200
a.effe3    1.572 0.480   0.639  1.256  1.559  1.894  2.536 1.018   110
b.effe2    3.484 1.337   0.960  2.543  3.490  4.472  6.148 1.009   360
b.effe3   -4.492 0.832  -6.134 -5.000 -4.471 -3.916 -3.022 1.012   150
test1     -7.977 1.164 -10.501 -8.788 -7.897 -7.218 -5.741 1.007   280
Presi[1]  -1.109 0.185  -1.520 -1.235 -1.103 -0.979 -0.805 1.009   190
[ ... ]
DIC info (using the rule, pD = Dbar-Dhat)
pD = 6.0 and DIC = 123.8
DIC is an estimate of expected predictive error (lower deviance is better).

> beta.vec                              # Truth in data-generation
[1] -4  1  2  6  2 -5
```

```
> print(glm(cbind(C, N-C) ~ pop * wetness, family = binomial)$coef, dig = 4)
                                        # The ML solution
          (Intercept)          popBlack Forest              popAlps
              -3.7255                   0.1342               1.5118
              wetness  popBlack Forest:wetness      popAlps:wetness
               5.5286                   3.6046              -4.3748
```

18.5 SUMMARY

Moving from the normal and the Poisson to a binomial ANCOVA involves only minor changes in the code of WinBUGS (and also R). Similarly, the concepts of residuals and posterior predictive distributions carry over to this class of models. We see examples for both.

EXERCISES

1. *Multiple binomial regression*: Create another habitat variable `X` and, using the data set created in this chapter, use WinBUGS to fit the following model `prop.black ~ pop*wet + X + wet:X`. That is, add the main effect of `X` plus the interaction between `wet` and `X`.
2. *Swiss hare data*: Model the probability of a "large" count as a function of land use and year (continuous), i.e., fit the following model:

   ```
   large.pop ~ land-use * year
   ```

   You may choose a threshold of about 5. You may also want to check for nonlinearity (on the scale of the logit link) by adding a quadratic effect of year.