CHAPTER

16

Poisson Mixed-Effects Model (Poisson GLMM)

OUTLINE

16.1 INTRODUCTION

The Poisson generalized linear mixed model (GLMM) is an extension of the Poisson generalized linear model (GLM) to include at least one additional source of random variation over and above the random variation intrinsic to a Poisson distribution. Here, we adopt a Poisson GLMM to analyze a set of long-term population surveys of Red-backed shrikes (Fig. 16.1).

We assume that pair counts over 30 years were available in each of 16 shrike populations (again, the balanced design is for convenience only). Our intent is to model population trends. First, we write down the random-coefficients model without correlation between the intercepts and slopes. This model is very similar to that for the normal linear case that we examined extensively in Chapter 12. Thanks to how we specify models in the BUGS language, this similarity is more evident than when fitting the model with a canned routine such as in R.

Introduction to WinBUGS for Ecologists
DOI: 10.1016/B978-0-12-378605-0.00016-8

FIGURE 16.1 Male Red-backed shrike (*Lanius collurio*), Switzerland, 2004. (*Photo A. Saunier*)

We model C_i, the number pairs of Red-backed shrikes counted in year i in study area j:

1. Distribution: $C_i \sim \text{Poisson}(\lambda_i)$
2. Link function: log, i.e., $\log(\lambda_i) = \log(E(C_i)) = \text{linear predictor}$
3. Linear predictor: $\alpha_{j(i)} + \beta_{j(i)} * x_i$
4. Submodel for parameters/distribution of random effects:

$$\alpha_j \sim \text{Normal}\ (\mu_\alpha, \sigma^2_\alpha)$$
$$\beta_j \sim \text{Normal}\ (\mu_\beta, \sigma^2_\beta)$$

So the GLMM is just the same as a simple GLM, but with the added submodels for the log-linear intercept and slope parameters that we use to describe the population trends. We don't add a year-specific "residual" to the linear predictor. This could be done to account for random year effects and would be a first step towards modeling serial autocorrelation, e.g., by imposing an autoregressive structure on successive random year effects (Littell et al., 2008), but we omit this added complexity here. Also, we don't model a correlation between intercepts and slopes.

In conducting this analysis, we implicitly make one of two assumptions. Either we assume that we find all shrike pairs in every year and study area or we assume that at least the proportion of pairs overlooked does not vary among years or study areas in a systematic way, i.e., that there is no time

trend in detection probability. Otherwise, the interpretation of these data would be difficult or impossible. For an alternative protocol for data collection and associated analysis method that doesn't require these potentially restrictive assumptions, see the binomial mixture model in Chapter 21.

16.2 DATA GENERATION

We generate data under the random-coefficients model without correlation between the intercepts and slopes.

```
n.groups <- 16
n.years <- 30
n <- n.groups * n.years
pop <- gl(n = n.groups, k = n.years)
```

We standardize the year covariate to a range from zero to one.

```
original.year <- rep(1:n.years, n.groups)
year <-  (original.year-1)/29
```

We build a design matrix without the intercept.

```
Xmat <- model.matrix(~pop*year-1-year)
print(Xmat[1:91,], dig = 2)          # Print top 91 rows
```

Next, we draw the intercept and slope parameter values from their normal distributions, whose hyperparameters we need to select.

```
intercept.mean <- 3       # Choose values for the hyperparams
intercept.sd <- 1
slope.mean <- -2
slope.sd <- 0.6
intercept.effects<-rnorm(n = n.groups, mean = intercept.mean, sd = intercept.sd)
slope.effects <- rnorm(n = n.groups, mean = slope.mean, sd = slope.sd)
all.effects <- c(intercept.effects, slope.effects)  # Put them all together
```

We assemble the counts C_i by first computing the linear predictor, then exponentiating it and finally adding Poisson noise. Then, we look at the data.

```
lin.pred <- Xmat[,] %*% all.effects       # Value of lin.predictor
C <- rpois(n = n, lambda = exp(lin.pred)) # Exponentiate and add Poisson noise
hist(C, col = "grey")                     # Inspect what we've created
```

We use a lattice graph to plot the shrike counts against time for each population (Fig. 16.2).

```
library("lattice")
xyplot(C ~  original.year | pop, ylab = "Red-backed shrike counts", xlab = "Year")
```

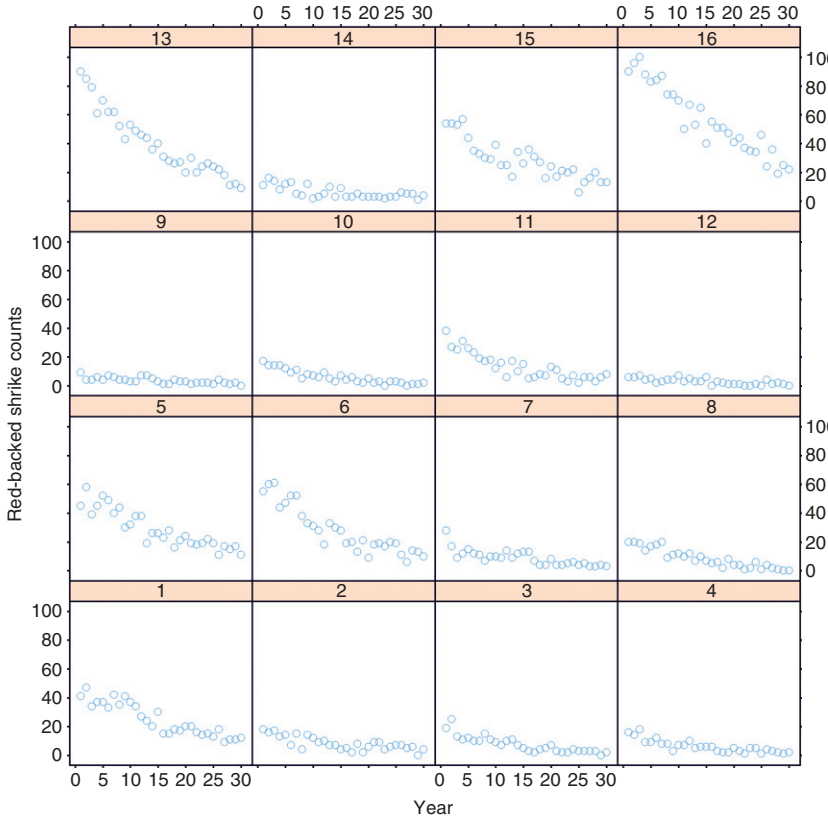

FIGURE 16.2 Trellis plot of pair counts in 16 populations of Red-backed shrikes over 30 years.

16.3 ANALYSIS UNDER A RANDOM-COEFFICIENTS MODEL

We could analyze the shrike counts under the assumption that all shrike populations have the same trend, but at different levels, corresponding to a random-intercepts model, as we did for the normal case (see Section 12.3). However, we only show here the analysis under the random-coefficients model. We assume that each population has a specific trend, i.e., that both slopes and intercepts are independent random variables governed by common hyperparameters.

16.3.1 Analysis Using R

```
library('lme4')
glmm.fit  <- glmer(C ~ year + (1 | pop) + ( 0+ year | pop), family = poisson)
glmm.fit                    # Inspect results
```

```
> glmm.fit# Inspect results
Generalized linear mixed model fit by the Laplace approximation
Formula: C ~ year + (1 | pop) + (0 + year | pop)
    AIC   BIC  logLik deviance
  602.7 619.3  -297.3     594.7
Random effects:
  Groups Name        Variance  Std.Dev.
  pop    (Intercept) 0.67721   0.82293
  pop    year        0.13311   0.36484
Number of obs: 480, groups: pop, 16

Fixed effects:
            Estimate  Std. Error  z value  Pr(>|z|)
(Intercept)   3.2406      0.2071    15.65    <2e-16  ***
year         -1.8233      0.1055   -17.28    <2e-16  ***
- - -
Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Correlation of Fixed Effects:
     (Intr)
year -0.044
```

16.3.2 Analysis Using WinBUGS

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

# Priors
  for (i in 1:n.groups){
    alpha[i] ~ dnorm(mu.int, tau.int)     # Intercepts
    beta[i] ~ dnorm(mu.beta, tau.beta)    # Slopes
}
  mu.int ~ dnorm(0, 0.001)    # Hyperparameter for random intercepts
  tau.int <- 1 / (sigma.int * sigma.int)
  sigma.int ~ dunif(0, 10)
```

```
  mu.beta ~ dnorm(0, 0.001)   # Hyperparameter for random slopes
  tau.beta <- 1 / (sigma.beta * sigma.beta)
  sigma.beta ~ dunif(0, 10)

# Poisson likelihood
  for (i in 1:n) {
    C[i] ~ dpois(lambda[i])
    lambda[i] <- exp(alpha[pop[i]] + beta[pop[i]]* year[i])
 }
}
",fill=TRUE)
sink()

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

# Inits function
inits <- function(){ list(alpha = rnorm(n.groups, 0, 2), beta = rnorm(n.groups, 0,
2), mu.int = rnorm(1, 0, 1), sigma.int = rlnorm(1), mu.beta = rnorm(1, 0, 1),
sigma.beta = rlnorm(1))}

# Parameters to estimate
parameters <- c("alpha", "beta", "mu.int", "sigma.int", "mu.beta", "sigma.beta")

# MCMC settings
ni <- 2000
nb <- 500
nt <- 2
nc <- 3

# Start Gibbs sampling
out <- bugs(win.data, inits, parameters, "glmm.txt", n.thin=nt, n.chains=nc,
n.burnin=nb, n.iter=ni, debug = TRUE)
```

This GLMM converges easily.

```
print(out, dig = 2)
> print(out, dig = 2)
Inference for Bugs model at "glmm.txt", fit using WinBUGS,
 3 chains, each with 2000 iterations (first 500 discarded), n.thin = 2
 n.sims = 2250 iterations saved
             mean    sd   2.5%    25%    50%    75%  97.5%  Rhat n.eff
[ ... ]
mu.int       3.24  0.24   2.77   3.08   3.23   3.40   3.75  1.00  2200
sigma.int    0.94  0.20   0.64   0.79   0.91   1.04   1.44  1.01   280
mu.beta     -1.83  0.12  -2.08  -1.90  -1.82  -1.74  -1.60  1.00   870
sigma.beta   0.41  0.10   0.25   0.34   0.40   0.47   0.65  1.00   970
[ ... ]
```

```
DIC info (using the rule, pD = var(deviance)/2)
pD = 31.7 and DIC = 2478.9
DIC is an estimate of expected predictive error (lower deviance is better).

# Compare with input values
intercept.mean ; intercept.sd ; slope.mean ; slope.sd
> intercept.mean ; intercept.sd ; slope.mean ; slope.sd
[1] 3
[1] 1
[1] -2
[1] 0.6
>
```

This comparison looks good.

16.4 SUMMARY

We have seen how the concept of random effects, which we first met in Chapters 9 and 12 for the normal linear case, can be carried over fairly smoothly to the non-normal case. Both frequentist and Bayesian analyses are fairly straightforward, but the model fitted in WinBUGS appears more transparent.

EXERCISES

1. *Swiss hare data 1*: Fit the random-coefficient Poisson regression to the counts without regard to land use. Take a single count per site and year. Don't forget to account for variable area by inclusion of an offset.
2. *Swiss hare data 2*: Now add a separate pair of hyperparameters (for intercept and slope) for arable and grassland areas; this should definitely get you published somewhere really nice. You could also add other explanatory variables, such as an index for fox abundance if you had that. Do not forget that you are not modeling *hare abundance*, just the *expected hare counts*. Nobody really knows what this means, but expected counts presumably represent an unknown, and you hope, constant, proportion of the true number of hares out there.