# 15 Poisson Generalized linear model or Poisson GLMM

# We assume that pair counts over 30 years were available in each of 16 shrike populations. Our intent is to model population trends. 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.

# 15.2 Data generation

# -------------------------------------------

set.seed(15)

nPops <- 16

nYears <- 30

n <- nPops \* nYears

pop <- gl(n = nPops, k = nYears)

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

orig.year <- rep(1:nYears, nPops)

year <- (orig.year-1) / 29

# We build a design matrix without the intercept and look at the top 91 rows --- make sure you understand what the design matrix means.

Xmat <- model.matrix(~pop \* year - 1 - year)

print(Xmat[1:91,], 2) # Print top 91 rows

# we draw the intercept and slope parameter values from their respective two normal distributions, and we need to pick values for the hyperparameters first.

**# Choose values for hyperparams and draw Normal random numbers**

intercept.mean <- 3

intercept.sd <- 1

slope.mean <- -2

slope.sd <- 0.6

intercept.effects <- rnorm(n = nPops, mean = intercept.mean, sd = intercept.sd)

slope.effects <- rnorm(n = nPops, mean = slope.mean, sd = slope.sd)

all.effects <- c(intercept.effects, slope.effects) # All together

**# Save true parameter values**

truth <- c(intercept.mean=intercept.mean, slope.mean=slope.mean,

intercept.sd=intercept.sd, slope.sd=slope.sd)

# we assemble the counts C 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

xyplot(C ~ orig.year | pop, ylab = "Red-backed shrike counts", xlab = "Year", pch = 16, cex = 1.2, col = rgb(0, 0, 0, 0.4))

# 15.3 Likelihood analysis with canned functions in R

#We specify a model without correlation between intercepts and slopes by using the '||' operator in the model formula. If this is new, then please read up on it in the documentation of that function.

library('lme4')

out15.3 <- glmer(C ~ year + (year || pop), family = poisson) # Fit GLMM

summary(out15.3) # Inspect results

**# Compare estimates with truth**

sds <- sqrt(as.numeric(summary(out15.3)$varcor))

glmer\_est <- c(fixef(out15.3), sds)

tmp <- cbind(truth=truth, glmer=glmer\_est)

print(tmp, 4)

#15.4 Bayesian analysis with JAGS

# ---------------------------------

**# Bundle and summarize data**

str(dataList <- list(C = C, pop = as.numeric(pop), year = year, nPops = nPops, n = n) )

**# Write JAGS model file**

cat(file="model15.4.txt", "

model {

**# Priors**

for (i in 1:nPops){

alpha[i] ~ dnorm(intercept.mean, intercept.tau) # Intercepts

beta[i] ~ dnorm(slope.mean, slope.tau) # Slopes

}

intercept.mean ~ dnorm(0, 0.001) # Hyperparam. for random intercepts

intercept.tau <- pow(intercept.sd, -2)

intercept.sd ~ dunif(0, 10)

slope.mean ~ dnorm(0, 0.001) # Hyperparameter for random slopes

slope.tau <- pow(slope.sd, -2)

slope.sd ~ dunif(0, 10)

**# 'Likelihood'**

for (i in 1:n){

C[i] ~ dpois(lambda[i])

lambda[i] <- exp(alpha[pop[i]] + beta[pop[i]] \* year[i])

# log(lambda[i]) <- alpha[pop[i]] + beta[pop[i]]\* year[i] # same

}

}

")

**# Function to generate starting values**

inits <- function(){

list(intercept.mean = rnorm(1), slope.mean=rnorm(1),

intercept.sd = runif(1), slope.sd=runif(1))

}

**# Parameters to estimate**

params <- c("intercept.mean", "slope.mean", "intercept.sd",

"slope.sd", "alpha", "beta")

**# MCMC settings**

na <- 1000 ; ni <- 3000 ; nb <- 1000 ; nc <- 3 ; nt <- 1

**# Call JAGS (ART <1 min), check convergence and summarize posteriors**

out15.4 <- jags(dataList, inits, params, "model15.4.txt", n.iter = ni, n.burnin = nb, n.chains = nc, n.thin = nt, n.adapt = na, parallel = TRUE)

par(mfrow=c(2, 2)); traceplot(out15.4) # not shown

print(out15.4, 3) # not shown

**# Compare likelihood with Bayesian estimates and with truth**

jags\_est <- unlist(out15.4$mean[1:4])

tmp <- cbind(truth=truth, glmer=glmer\_est, JAGS=jags\_est)

print(tmp, 4)

# 15.6 Bayesian analysis with Stan

library(rstan)

**# Bundle and summarize data**

str(dataList <- list(C = C, pop = as.numeric(pop), year = year, nPops = nPops, n = n) )

**# Write Stan model**

cat(file="model15\_6.stan", "

data{

int n;

int nPops;

int C[n];

vector[n] year;

int pop[n];

}

parameters{

real intercept\_mean;

real slope\_mean;

real<lower=0> intercept\_sd;

real<lower=0> slope\_sd;

vector[nPops] alpha;

vector[nPops] beta;

}

model{

vector[n] lambda;

for (i in 1:nPops){

alpha[i] ~ normal(intercept\_mean, intercept\_sd);

beta[i] ~ normal(slope\_mean, slope\_sd);

}

for (i in 1:n){

lambda[i] = exp(alpha[pop[i]] + beta[pop[i]] \* year[i]);

C[i] ~ poisson(lambda[i]);

}

}

")

**# HMC settings**

ni <- 2000 ; nb <- 1000 ; nc <- 3 ; nt <- 1

**# Call STAN (ART 42/15 sec), assess convergence and print results table**

system.time(

out15.6 <- stan(file = "model15\_6.stan", data = dataList,

warmup = nb, iter = ni, chains = nc, thin = nt) )

rstan::traceplot(out15.6) # Wilted-flower plots: not shown

print(out15.6, dig = 3) # not shown

**# Compare estimates with truth**

stan\_est <- summary(out15.6)$summary[1:4,1]

tmp <- cbind(truth=truth, glmer=glmer\_est, JAGS=jags\_est, Stan=stan\_est)

print(tmp, 4)