# 7 Normal linear regression

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

# As a motivating example for a linear regression analysis we take a Swiss survey of the Wallcreeper (Fig. 7-1), a spectacular little cliff-inhabiting bird that appears to have declined greatly in Switzerland in recent years. Assume that we had data on the proportion of sample quadrats in which the species was observed in Switzerland for the years 1990–2005 and that we were willing to assume that the random deviations about a linear time trend were normally distributed. This is for illustration only, usually, we would use logistic regression (chapters 16–18) or a site-occupancy model (see chapter 19) to make inference about such data that have to do with the distribution of a species and represent a proportion (i.e., number occupied/number surveyed).

Importantly, in this chapter we will also introduce posterior predictive model checking, including computation of the Bayesian p-value (Gelman *et al.* 1996; Gelman and Hill 2007, chap. 24). This is a very general concept for checking the goodness-of-fit of a model, i.e. of making a judgment of how good a model is in absolute terms .

# 7.2 Data generation

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

set.seed(77) # Seed 7 yields too strange pattern...

n <- 16 # Number of years

a <- 40 # Intercept

b <- -0.5 # Slope

sigma2 <- 25 # Residual variance

x <- 1:16 # Values of covariate year

eps <- rnorm(n, mean = 0, sd = sqrt(sigma2))

y <- a + b\*x + eps # Assemble data set

plot((x+1989), y, xlab = "Year", las = 1, ylab = "Prop. occupied (%)", cex = 1.2, pch = 16, col = rgb(0,0,0,0.4), frame = FALSE) # not shown

**# Save true values for later comparisons**

truth <- c(alpha=a, beta=b, sigma=sqrt(sigma2))

# 7.3 Likelihood analysis using canned functions in R

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

summary(out7.3 <- lm(y ~ x))

lines(x+1989, predict(out7.3), col = "blue", lwd = 2)

**# Compare estimates with truth**

lm\_est <- c(coef(out7.3), sigma=sigma(out7.3))

cbind(truth=truth, lm=lm\_est)

7.4 Bayesian analysis with JAGS

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

**# Bundle and summarize data**

str(dataList <- list(y = y, x = x, n = n))

**# Write JAGS model file**

cat(file="model7.4.txt", "

model {

**# Priors**

alpha ~ dnorm(0,0.001)

beta ~ dnorm(0,0.001)

sigma ~ dunif(0, 100)

**# Likelihood**

for (i in 1:n) {

y[i] ~ dnorm(mu[i], tau)

mu[i] <- alpha + beta\*x[i]

}

**# Derived quantities**

tau <- pow(sigma, -2)

p.decline <- 1-step(beta) # Probability of decline

}

")

**# Function to generate starting values**

inits <- function(){ list(alpha=rnorm(1), beta=rnorm(1), sigma=rlnorm(1))}

**# Parameters to estimate**

params <- c("alpha","beta", "sigma", "p.decline")

**# MCMC settings**

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

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

out7.4 <- jags(dataList, inits, params, "model7.4.txt", n.iter = ni, n.burnin = nb, n.chains = nc, n.thin = nt, n.adapt = na, parallel = TRUE)

par(mfrow=c(2, 2)); jagsUI::traceplot(out7.4) # not shown

print(out7.4, 3)

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

jags\_est <- out7.4$summary[1:3,1]

cbind(truth=truth, lm=lm\_est, JAGS=jags\_est)

# Forming predictions

**# Fig. 7-3 (left)**

par(mfrow = c(1, 2), mar = c(6,6,6,3), cex.lab = 1.5, cex.axis = 1.5, cex.main = 2)

plot((x+1989), y, xlab = "Year", las = 1, ylab = "Prop. occupied (%)", cex = 2, pch = 16, frame = FALSE, col = rgb(0,0,0, 0.5))

abline(lm(y~ I(x+1989)), col = "blue", lwd = 2)

pred.y <- out7.4$mean$alpha + out7.4$mean$beta \* x

points(1990:2005, pred.y, type = "l", col = "red", lwd = 2)

legend('bottomleft', legend = "Maximum likelihood\n Posterior mean", cex = 1.2, bty = 'n', lty = 1, col = c('blue', 'red'), lwd = 3)

# We set up an R data structure to hold the predictions, fill them, then determine the appropriate percentile points and produce a plot (Fig. 7-3 right):

predictions <- array(dim = c(length(x), length(out7.4$sims.list$alpha)))

for(i in 1:length(x)){

predictions[i,] <- out7.4$sims.list$alpha + out7.4$sims.list$beta\*i

}

LPB <- apply(predictions, 1, quantile, probs = 0.025) # Lower bound

UPB <- apply(predictions, 1, quantile, probs = 0.975) # Upper bound

**# Fig. 7-3 (right)**

plot(1990:2005, y, xlab = "Year", las = 1, ylab = "Prop. occupied (%)", cex = 1.2, pch = 16, ylim = c(20, 50), col = rgb(0,0,0,0.5), frame = FALSE)

points(1990:2005, out7.4$mean$alpha + out7.4$mean$beta \* x, type = "l", lwd = 2, col = 'red')

polygon(c(1990:2005, rev(1990:2005)), c(LPB, rev(UPB)), col = rgb(1, 0, 0, 0.2), border = NA)

# 7.6 Bayesian analysis with Stan

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

**# Load Stan R package**

library(rstan)

**# Bundle and summarize data (same as before)**

str(dataList <- list(y = y, x = x, n = n)) # not shown

**# Write text file with model description in BUGS language**

cat(file = "model7\_6.stan", # This line is R code

"data { // This is the first line of Stan code

int<lower=0> n; // Define the format of all data

vector[n] y; // ... including the dimension of vectors

vector[n] x; //

}

parameters { // Define format for all parameters

real alpha;

real beta;

real<lower=0> sigma;

}

model {

// Priors

alpha ~ normal(0, 1000);

beta ~ normal(0, 1000);

sigma ~ cauchy(0, 10);

// 'Likelihood'

y ~ normal(alpha + beta \* x, sigma);

}

} // This is the last line of Stan code

" )

**# HMC settings**

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

**# Call STAN (ART 30/3 sec)**

system.time(

out7.6 <- stan(file = "model7\_6.stan", data=dataList,

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

rstan::traceplot(out7.6) # not shown

print(out7.6, dig = 2) # not shown

**# Compare estimates with truth**

stan\_est <- summary(out7.6)$summary[1:3,1]

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

print(tmp, 4)

# 7.7 Do-it-yourself MLEs

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

# The likelihood to be maximized is the joint likelihood over all units in the data set, where the contribution from each datum comes from a Normal density which differs in terms of the expectation mu.

**# Definition of NLL for a OLS with one covariate**

NLL <- function(param, y, x) {

alpha <- param[1]

beta <- param[2]

sigma <- param[3]

mu <- alpha + beta \* x

L <- dnorm(y, mu, sigma) # Likelihood contr. for 1 observation

LL <- log(L) # Loglikelihood contr. for 1 observation

NLL <- -sum(LL) # NLL for all observations

return(NLL)

}

**# Minimize that NLL to find MLEs and also get SEs**

inits <- c('alpha' = 50, 'beta' = -2, 'sigma' = 10)

out7.7 <- optim(inits, NLL, y = y, x = x, hessian=TRUE, method = 'BFGS')

getMLE(out7.7, 5)

**# Compare estimates with truth and previous estimates**

diy\_est <- out7.7$par

tmp <- cbind(truth=truth, lm=lm\_est, JAGS=jags\_est, Stan=stan\_est, DIY=diy\_est)

print(tmp, 4)