# Problem 1:

Y.values<-c(975.51,1734.02,2177.37,2965.93,1250.45,2821.97,0.00,2322.25,0.00,

972.20,4352.78,2546.13,2186.04,0.00,869.86,1281.62,3557.40,1234.57,741.99)

W.values <-c(1,1,1,1,1,1,0,1,0,1,1,1,1,0,1,1,1,1,1)

# data for the observed 19 individuals

W.tot <- sum(W.values) # count of all "successes"

**######################################################################**

**#**

**# 1(a,b,c,d)**

**#**

**# The distribution of theta|W\_1,...W\_19,W20 is beta(20-W\_1-...-W\_20+1, W\_1+...+W20+1).**

**# The distribution of W\_20|theta is Bernoulli(1-theta), i.e., Binomial(1,1-theta).**

**# The distribution of Y\_20|theta,W\_20=1 is gamma(shape=4,rate=1/500).**

**# The distribution of Y\_20|theta,W\_20=0 is degenerate at 0, i.e., is 0 with probability 1.**

**#**

**######################################################################**

**######################################################################**

**#**

**# 1(e)**

**#**

**# Gibbs sampler below:**

**#**

**######################################################################**

tot.draws <- 20000 # number of sampled values from each full conditional

# Initial values for quantities of interest:

Y.20.init <- 2000

W.20.init <- 1

theta.init <- 0.5

# Dummy Vectors that will hold values for samples of interest:

Y.20.vec <- c(Y.20.init, rep(NULL, times = tot.draws) )

W.20.vec <- c(W.20.init, rep(NULL, times = tot.draws) )

theta.vec <- c(theta.init, rep(NULL, times = tot.draws) )

for (j in 2:(tot.draws+1) ) {

theta.vec[j] <- rbeta(n=1, 20 - W.tot - W.20.vec[j-1] + 1, W.tot + W.20.vec[j-1] + 1)

W.20.vec[j] <- rbinom(n=1, size=1, prob=1-theta.vec[j])

Y.20.vec[j] <- ifelse(test=(W.20.vec[j]==1),rgamma(n=1, shape=4, rate=1/500),0)

}

# Remove initial values:

theta.vec <- theta.vec[-1]

W.20.vec <- W.20.vec[-1]

# remove first 2000 sampled values as "burn-in":

theta.post <- theta.vec[-(1:2000)]

W.20.post <- W.20.vec[-(1:2000)]

Y.20.post <- Y.20.vec[-(1:2000)]

**######################################################################**

**#**

**# 1(f)**

**#**

**# A posterior estimate of theta is around 0.19**

**# A posterior estimate of E(Y) is around 1620 (this will vary, but a bit over 1600).**

**#**

**######################################################################**

## Posterior summary for theta:

#plot(density(theta.post)) # plot of estimated posterior for theta

print(paste("Posterior mean for theta: ",mean(theta.post))) # Posterior mean of the other 18000 sampled theta values

print(paste("Posterior median for theta: ",median(theta.post))) # Posterior median of the other 18000 sampled theta values

#quantile(theta.post, probs=c(0.025,0.975) ) # approximate 95% quantile-based interval for theta

#emp.hpd(theta.post, conf=0.95) # approximate 95% HPD interval for theta

## NOTE: The observed-data MLE for theta here is:

print(paste("MLE for theta: ",mean(1-W.values)))

# which is biased in this case because it lacks the missing value.

## Posterior summary for the missing data value Y.20:

print(paste("Posterior mean for E(Y\_20): ",mean(Y.20.post))) # Posterior mean for Y.20

## NOTE: The observed-data MLE for E(Y) here is:

print(paste("MLE for E(Y\_20): ",mean(Y.values)))

# which is biased in this case because it lacks the missing value.

############

## Problem 2:

############

**######################################################################**

**#**

**# 2(a,b)**

**#**

**# Since we know beta > 1, then we know 0 < theta < 1.**

**# As the prior on theta, let's pick a Beta distribution with mean 1/10**

**# and most of the probability between 1/15 and 1/5. Answers may vary,**

**# but let's say a Beta(3,27) prior on theta.**

**#**

**######################################################################**

**######################################################################**

**#**

**# 2(c): I will post some details about this separately.**

**#**

**######################################################################**

**######################################################################**

**#**

**# 2(d): To adjust the acceptance rate, we would adjust the variance**

**# of the proposal distribution. In this algorithm, we would adjust**

**# the value of k up or down, but keeping k between 0 and 1.**

**#**

**######################################################################**

############

## Problem 3:

############

y <- c(2289,3393,3271,2648,2897,3327,2970,2535,3138,3573)/1000

x1 <- c(15,17,18,15,16,19,17,16,18,19)

X <- as.matrix( cbind(rep(1, length(x1)), x1) )

###### Setting up prior specification :

# Suppose that based on "expert opinion" we have the following guess:

xtilde.obs.1 <- 15

ytilde.obs.1 <- 2.700

xtilde.obs.2 <- 18

ytilde.obs.2 <- 3.000

# Making the matrix X~ :

prior.obs.stacked <- rbind(xtilde.obs.1, xtilde.obs.2)

xtilde <- cbind(rep(1, times=nrow(prior.obs.stacked) ), prior.obs.stacked)

# Making the vector Y~ :

ytilde <- c(ytilde.obs.1, ytilde.obs.2)

# The prior mean on the beta vector that we are inducing is:

prior.mean.beta <- as.vector(solve(xtilde) %\*% ytilde)

print("Prior mean for beta vector was:"); print(round(prior.mean.beta,3))

**#########################################################################**

**#**

**# 3(a)**

**#**

**# We only need two hypothetical observations because we only have one predictor variable**

**# in the model: If there are p predictors, we need p+1 hypothetical observations.**

**#**

**#########################################################################**

**#########################################################################**

**#**

**# 3(b)**

**#**

**# The prior mean vector for Beta is (1.2, 0.1).**

**#**

**#########################################################################**

# Diagonal matrix D contains weights that indicate how much worth we place on

# our hypothetical prior observations (note the weights could vary if we are more

# confident about some hypothetical prior observations than about others):

D <- diag(c(1,1))

**#########################################################################**

**#**

**# 3(c)**

**#**

**# This matrix D implies that our prior knowledge about the betas is worth the same**

**# as two sample observations.**

**#**

**#########################################################################**

### Choosing prior parameters a and b for gamma prior on tau:

# The parameter "a" reflects our confidence in our prior:

# Let's choose "a" to be 0.5, which implies that our

# prior knowledge about tau is "worth" about 1 sample observation.

a <- 0.5

**#########################################################################**

**#**

**# 3(d)**

**#**

**# The choice a=0.5 implies that our prior knowledge about tau is worth**

**# about one sample observation.**

**#**

**#########################################################################**

# Then set the prior guess for sigma to be (3.000-2.700)/1.645 = .1823708

# So the prior guess for sigma^2 is (0.1823708^2) = 0.03325911

# So the prior guess for tau is 1/0.03325911 = 30.06695

# Since the gamma prior mean is a/b, let

# b = a / tau.guess:

tau.guess <- 30.06695

b <- a / tau.guess

##### Posterior information for tau and beta:

# Calculate beta-hat :

betahat <- as.vector(solve(t(X) %\*% X + t(xtilde)%\*%solve(D)%\*%xtilde) %\*% (t(X)%\*%y + t(xtilde)%\*%solve(D)%\*%ytilde))

# Calculate s\* :

n <- length(y)

sstar <- as.numeric( (t(y-X%\*%betahat)%\*%(y-X%\*%betahat) + t(ytilde-xtilde%\*%betahat)%\*%solve(D)%\*%(ytilde-xtilde%\*%betahat) + 2\*b)/(n+2\*a) )

print(paste("mean of the conditional posterior for beta: ", betahat))

### Point estimates for tau (and thus for sigma^2):

p.mean.tau <- 1 / sstar

p.mean.sig.sq <- 1 / p.mean.tau

p.median.tau <- qgamma(0.50, shape=(n+2\*a)/2, rate=((n+2\*a)/2)\*sstar)

p.median.sig.sq <- 1 / p.median.tau

print(paste("posterior.mean for sigma^2=", round(p.mean.sig.sq,3),

"posterior.median for sigma^2=", round(p.median.sig.sq,3) ))

#[1] "posterior.mean for sigma^2= 0.041 posterior.median for sigma^2= 0.044"

### Marginal Interval estimate for tau (and sigma^2):

library(TeachingDemos) # loading TeachingDemos package, to use hpd function

hpd.95.tau <- hpd(qgamma, shape=(n+2\*a)/2, rate=((n+2\*a)/2)\*sstar )

hpd.95.sig.sq <- 1 / hpd.95.tau

print("95% posterior interval for sigma^2:")

round(sort(hpd.95.sig.sq), 3)

#[1] 0.022 0.153

### APPROACH TO GET POINT ESTIMATES FOR BETA:

# Randomly generate many tau values from its posterior distribution:

how.many <-30000

tau.values <- rgamma(n=how.many, shape=(n+2\*a)/2, rate=((n+2\*a)/2)\*sstar )

library(mvtnorm)

beta.values<- matrix(0,nr = how.many, nc=length(betahat))

for (j in 1:how.many){

beta.values[j,] <- rmvnorm(n=1, mean=betahat, sigma= (1/tau.values[j])\*solve(t(X)%\*%X + t(xtilde)%\*%solve(D)%\*%xtilde) )

}

# Posterior median for each regression coefficient:

post.medians<-apply(beta.values,2,median)

# 90% posterior interval for each regression coefficient:

post.lower <- apply(beta.values,2,quantile, probs=0.05)

post.upper <- apply(beta.values,2,quantile, probs=0.95)

## Summarizing:

names.predictors <- c("x")

beta.post.summary <- data.frame(cbind(post.lower, post.medians, post.upper), row.names=c("intercept", names.predictors))

names(beta.post.summary) <- c('0.05 Quantile', '0.5 Quantile', '0.95 Quantile')

print(beta.post.summary)

**#########################################################################**

**#**

**# (e)**

**#**

**# 0.05 Quantile 0.5 Quantile 0.95 Quantile**

**# intercept -1.9776475 -0.7441322 0.4992074**

**# x 0.1468315 0.2201494 0.2928889**

**#**

**# The fitted model is: Y-hat = -0.7441 + 0.2201x**

**#**

**# 90% credible interval for beta\_1: (0.147, 0.293)**

**#**

**# The predicted birth weight for a baby born to a 17.5-year-old mother is 3.108 kg.**

**#**

**#########################################################################**

**#########################################################################**

**#**

**# 3(f)**

**#**

**# Answers will vary for checking model fit, but some reasonable approach should be taken.**

**#**

**#########################################################################**

##################

# Problem 4

##################

my.datafile <- tempfile()

cat(file=my.datafile, "

6.7 62 81 2.59 200 2.3010

5.1 59 66 1.70 101 2.0043

7.4 57 83 2.16 204 2.3096

6.5 73 41 2.01 101 2.0043

7.8 65 115 4.30 509 2.7067

5.8 38 72 1.42 80 1.9031

5.7 46 63 1.91 80 1.9031

3.7 68 81 2.57 127 2.1038

6.0 67 93 2.50 202 2.3054

3.7 76 94 2.40 203 2.3075

6.3 84 83 4.13 329 2.5172

6.7 51 43 1.86 65 1.8129

5.8 96 114 3.95 830 2.9191

5.8 83 88 3.95 330 2.5185

7.7 62 67 3.40 168 2.2253

7.4 74 68 2.40 217 2.3365

6.0 85 28 2.98 87 1.9395

3.7 51 41 1.55 34 1.5315

7.3 68 74 3.56 215 2.3324

5.6 57 87 3.02 172 2.2355

5.2 52 76 2.85 109 2.0374

3.4 83 53 1.12 136 2.1335

6.7 26 68 2.10 70 1.8451

5.8 67 86 3.40 220 2.3424

6.3 59 100 2.95 276 2.4409

5.8 61 73 3.50 144 2.1584

5.2 52 86 2.45 181 2.2577

11.2 76 90 5.59 574 2.7589

5.2 54 56 2.71 72 1.8573

5.8 76 59 2.58 178 2.2504

3.2 64 65 0.74 71 1.8513

8.7 45 23 2.52 58 1.7634

5.0 59 73 3.50 116 2.0645

5.8 72 93 3.30 295 2.4698

5.4 58 70 2.64 115 2.0607

5.3 51 99 2.60 184 2.2648

2.6 74 86 2.05 118 2.0719

4.3 8 119 2.85 120 2.0792

4.8 61 76 2.45 151 2.1790

5.4 52 88 1.81 148 2.1703

5.2 49 72 1.84 95 1.9777

3.6 28 99 1.30 75 1.8751

8.8 86 88 6.40 483 2.6840

6.5 56 77 2.85 153 2.1847

3.4 77 93 1.48 191 2.2810

6.5 40 84 3.00 123 2.0899

4.5 73 106 3.05 311 2.4928

4.8 86 101 4.10 398 2.5999

5.1 67 77 2.86 158 2.1987

3.9 82 103 4.55 310 2.4914

6.6 77 46 1.95 124 2.0934

6.4 85 40 1.21 125 2.0969

6.4 59 85 2.33 198 2.2967

8.8 78 72 3.20 313 2.4955

", sep=" ")

mydata <- read.table(my.datafile, header=FALSE, col.names=c("x1","x2","x3", "x4","y.untransf", "y"))

attach(mydata)

**#########################################################################**

**#**

**# 4(a): Using the transformed survival time is helpful**

**# because it more closely resembles a normal distribution**

**# (the untransformed survival times are highly skewed to the right)**

**#**

**hist(y)**

**hist(y.untransf)**

**#**

**#########################################################################**

X <- as.matrix( cbind(rep(1, length(x1)), x1,x2,x3,x4) )

# Gill's function to produce posterior medians

# and quantile based credible intervals for the

# regression parameters (based on multivariate-t posterior)

t.ci.table <- function(coefs, cov.mat, level=0.95, degrees=Inf, quantiles=c(0.025, 0.50, 0.975) ) {

quantile.mat <- cbind(coefs, sqrt(diag(cov.mat)), t(qt(quantiles, degrees) %o% sqrt(diag(cov.mat))) +

matrix(rep(coefs,length(quantiles)), ncol=length(quantiles)) )

quantile.names <- c("mean", "Std. Error")

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

quantile.names <- c(quantile.names, paste(quantiles[i], "Quantile"))

}

dimnames(quantile.mat)[2] <- list(quantile.names)

return(round(quantile.mat,4))

}

### Noninformative Prior Analysis:

bhat <- solve(t(X) %\*% X) %\*% t(X) %\*% y

sig2hat <- t(y - X %\*% bhat) %\*% (y - X %\*% bhat) / (nrow(X) - ncol(X))

my.cov.mat <- solve(t(X) %\*% X) \* ((nrow(X)-ncol(X))\*sig2hat / (nrow(X) - ncol(X) - 2))[1,1]

# Getting posterior information about the betas:

table.beta.info <- t.ci.table(bhat, my.cov.mat, degrees = nrow(X) - ncol(X))

table.beta.info

**#########################################################################**

**#**

**# X4 (the liver measurement) seems less likely to be important**

**# since the credible interval for beta\_4 contains 0:**

**#**

**# mean Std. Error 0.025 Quantile 0.5 Quantile 0.975 Quantile**

**# 0.4888 0.0513 0.3857 0.4888 0.5918**

**# x1 0.0685 0.0056 0.0574 0.0685 0.0797**

**# x2 0.0093 0.0004 0.0084 0.0093 0.0102**

**# x3 0.0095 0.0004 0.0087 0.0095 0.0103**

**# x4 0.0019 0.0099 -0.0180 0.0019 0.0218**

**#**

**#########################################################################**

# Getting posterior information about sigma^2:

library(TeachingDemos) # loading TeachingDemos package, to use hpd function

library(pscl) # loading pscl package, to use inverse gamma distribution

my.alpha <- (nrow(X) - ncol(X) - 1)/2

my.beta <- 0.5 \* sig2hat \* (nrow(X) - ncol(X))

hpd.sig.sq <- hpd(qigamma, alpha=my.alpha, beta=my.beta)

round(hpd.sig.sq, 3)

**#########################################################################**

**#**

**#(c)**

**#**

**c(1,6,65,2,150) %\*% bhat**

**#**

**# 1.809157**

**#**

**#**

**10^1.81**

**#**

**#[1] 64.56542**

**#**

**#########################################################################**

**#**

**#**

**#(d)**

**#**

**10^.009475**

**#[1] 1.022057**

**#**

**#########################################################################**

# Model selection:

#

lpy.X <- function(y,X,g=length(y),nu0=1,s20=try(summary(lm(y~-1+X))$sigma^2,silent=TRUE)) {

n<-dim(X)[1]; p<-dim(X)[2]

if(p==0) {Hg<-0; s20<-mean(y^2)}

if(p>0) {Hg<-(g/(g+1))\*X%\*%solve(t(X)%\*%X)%\*%t(X)}

SSRg<- t(y)%\*%( diag(1,nrow=n) - Hg)%\*%y

-.5\*(n\*log(pi)+p\*log(1+g)+(nu0+n)\*log(nu0\*s20+SSRg)-nu0\*log(nu0\*s20))+lgamma((nu0+n)/2)-lgamma(nu0/2)

}

### Starting values for Gibbs Sampler:

z<-rep(1,dim(X)[2]) # starting with z = all 1's (all terms in model)

lpy.c<-lpy.X(y,X[,z==1,drop=FALSE])

S <- 3000 # number of Monte Carlo iterations

Z<-matrix(NA,S,dim(X)[2])

### The Gibbs Sampler:

for(s in 1:S)

{

for(j in sample(1:dim(X)[2]))

{

zp<-z; zp[j] <- 1-zp[j]

lpy.p<-lpy.X(y,X[,zp==1,drop=FALSE])

r<- (lpy.p - lpy.c)\*(-1)^(zp[j]==0)

z[j]<-rbinom(1,1,1/(1+exp(-r)))

if(z[j]==zp[j]) {lpy.c<-lpy.p}

}

Z[s,]<-z

}

#########

poss.z.vectors <- matrix( c(

1,0,0,0,0,

1,1,0,0,0,

1,0,1,0,0,

1,0,0,1,0,

1,0,0,0,1,

1,1,1,0,0,

1,1,0,1,0,

1,1,0,0,1,

1,0,1,1,0,

1,0,1,0,1,

1,0,0,1,1,

1,1,1,1,0,

1,1,0,1,1,

1,1,1,0,1,

1,0,1,1,1,

1,1,1,1,1

), ncol=5, byrow=T)

z.probs <- rep(0, times= nrow(poss.z.vectors))

for(i in 1:nrow(poss.z.vectors)) {

z.probs[i] <- sum(apply(Z,1,identical, y=poss.z.vectors[i,]))

}

z.probs <- z.probs/sum(z.probs)

cbind(poss.z.vectors, z.probs)[rev(order(z.probs)),]

#

**#########################################################################**

**#**

**#(e)**

**#**

**# Best model is: The one with x1, x2, x3 (but not x4).**

**# This matches the idea from (a) that x4 may not be an important predictor.**

**#**

**#**

**# Posterior model probabilities:**

**#**

**# z.probs**

**# [1,] 1 1 1 1 0 0.407287449**

**# [2,] 1 0 1 1 0 0.296356275**

**# [3,] 1 0 1 1 1 0.142510121**

**# [4,] 1 1 1 1 1 0.054251012**

**#**

**#########################################################################**

**#########################################################################**

**## Problem 5:**

**#**

**# Much of the specific answer to this problem will vary based on the analysis chosen.**

**# There should be some discussion of (1) the prior specification used,**

**# (2) model selection, (3) assessing model fit, (4) overall conclusions.**

**#**

**#########################################################################**

**#########################################################################**

**#**

**# term estimate std.error conf.low conf.high**

**# <chr> <dbl> <dbl> <dbl> <dbl>**

**# 1 (Intercept) 0.650 0.444 -0.256 1.50**

**# 2 x 0.505 0.328 -0.130 1.15**

**# 3 xsq -0.103 0.0574 -0.216 0.00713**

**# 4 urban1 -0.501 0.194 -0.878 -0.132**

**#**

**# We see the estimated coefficient of "urban" is -0.5, so the expected offspring count for urban**

**# sparrows is exp(-0.5) = 0.607 times that of rural sparrows (of the same age),**

**# i.e, a 40% reduction for urban sparrows compared to rural.**

**#**

**# Using just the fitted model with the point estimates, the expected offspring count**

**# for a 3-year-old rural sparrow is**

**# exp(0.65+0.505\*3-0.103\*9-0.501\*0)**

**# = 3.448709**

**#**

**# Perhaps a better approach is to find the mean of the posterior predictive distribution**

**# for such a sparrow:**

**# Posterior prediction of offspring of 3-yr-old rural sparrow:**

**# example\_prediction <- posterior\_predict(**

**# sparrow\_model, newdata = data.frame(x=3,xsq=9,urban==0))**

**# A posterior predictive distribution for this individual:**

**# example\_prediction**

**# 0 1 2 3 4 5 6 7 8 9 10 11 12**

**# 0.083 0.185 0.225 0.198 0.140 0.085 0.046 0.022 0.010 0.004 0.002 0.001 0.000**

**# 13 14 15**

**# 0.000 0.000 0.000**

**# Posterior predictive mean for this individual:**

**# print(post.pred.mean)**

**# [1] 2.782012**

**#**

**#**

**#**

**#########################################################################**

####################################################

##### (Sparrow data with extra predictor)

####################################################

library(mvtnorm)

# May need to install the mvtnorm package first?

# If so, type at the command line: install.packages("mvtnorm", dependencies=T)

# while plugged in to the internet.

sparrow.data <- read.table("https://people.stat.sc.edu/hitchcock/sparrowdatamore.txt", header=T)

y <- sparrow.data[,1]

x <- sparrow.data[,2]; xsq <- x^2

urban <- sparrow.data[,3]

X <- cbind(rep(1,times=length(x)), x, xsq, urban)

p <- dim(X)[2] # number of columns of matrix X

beta.prior.mean <- c(0,0,0,-0.5)

beta.prior.sd <- rep(10,times=p)

proposal.cov.matrix <- var(log(y+1/2))\*solve(t(X)%\*%X)

# should be reasonable in this case: should be close to (sigma^2)\*(X'X)^{-1}

S <- 30000

beta.current <- rep(0,times=p) # initial value for M-H algorithm

acs <- 0 # will be to track "acceptance rate"

beta.values <- matrix(0,nrow=S,ncol=p) # will store sampled values of beta vector

for (s in 1:S) {

beta.proposed <- t(rmvnorm(1, beta.current, proposal.cov.matrix))

log.accept.ratio <- sum(dpois(y,exp(X %\*% beta.proposed), log=T)) - sum(dpois(y,exp(X %\*% beta.current), log=T)) +

sum(dnorm(beta.proposed, beta.prior.mean, beta.prior.sd, log=T)) -

sum(dnorm(beta.current, beta.prior.mean, beta.prior.sd, log=T) )

if (log.accept.ratio > log(runif(1)) ) {

beta.current <- beta.proposed

acs <- acs + 1

}

beta.values[s,] <- beta.current

}

acs/S # gives the acceptance rate

par(mfrow=c(2,2))

acf(beta.values[,1]) # plot autocorrelation values for beta\_0

acf(beta.values[,2]) # plot autocorrelation values for beta\_1

acf(beta.values[,3]) # plot autocorrelation values for beta\_2

acf(beta.values[,4]) # plot autocorrelation values for beta\_3

par(mfrow=c(1,1))

# Seems to be an issue with serial dependence.

# Thinning out the sampled values by taking every 10th row:

beta.values.thin <- beta.values[10\*(1:(S/10) ),]

par(mfrow=c(2,2))

acf(beta.values.thin[,1]) # plot autocorrelation values for beta\_0

acf(beta.values.thin[,2]) # plot autocorrelation values for beta\_1

acf(beta.values.thin[,3]) # plot autocorrelation values for beta\_2

acf(beta.values.thin[,4]) # plot autocorrelation values for beta\_3

par(mfrow=c(1,1))

# Looks much better...

# Trace plots for the sampled beta\_0 and beta\_1 values:

par(mfrow=c(2,2))

plot(beta.values.thin[,1], type='l')

plot(beta.values.thin[,2], type='l')

plot(beta.values.thin[,3], type='l')

plot(beta.values.thin[,4], type='l')

par(mfrow=c(1,1))

# Removing burn-in:

beta.values.thin.b <- beta.values.thin[-(1:200),]

### Posterior summary:

apply(beta.values.thin.b,2,median) # Posterior medians for each regression coefficent

rbind( emp.hpd(beta.values.thin.b[,1], conf=0.95),

emp.hpd(beta.values.thin.b[,2], conf=0.95),

emp.hpd(beta.values.thin.b[,3], conf=0.95),

emp.hpd(beta.values.thin.b[,4], conf=0.95) ) # approximate 95% HPD intervals

## Posterior prediction for model fit:

num.draws<-nrow(beta.values.thin.b)

y.hats <- matrix(0,nrow=num.draws,ncol=nrow(X))

for (i in 1:nrow(beta.values.thin.b)){

y.hats[i,] <- exp( X %\*% as.matrix(beta.values.thin.b[i,]))

}

matrix.of.ys <- matrix(y, nrow=nrow(y.hats), ncol=ncol(y.hats), byrow=T)

# In-sample MAE:

mae <- mean(abs(matrix.of.ys - y.hats));

print(mae)

### Poisson regression using the 'rstanarm' package:

# Load packages

library(bayesrules)

library(rstanarm)

library(bayesplot)

library(tidyverse)

library(tidybayes)

library(broom.mixed)

urban <- as.factor(urban)

sparrow\_data <- data.frame(y,x,xsq,urban)

# stan\_glm wants to start out with the data in a data frame...

sparrow\_model <- stan\_glm(y ~ x+xsq+urban,

data=sparrow\_data,

family = poisson,

prior\_intercept = normal(0, 0.5),

prior = normal(c(0,0,-0.5), 2.5, autoscale = TRUE),

chains = 2, iter = 5000\*2,

prior\_PD = FALSE)

# Some built-in MCMC checks:

mcmc\_trace(sparrow\_model)

mcmc\_dens\_overlay(sparrow\_model)

mcmc\_acf(sparrow\_model)

# A little autocorrelation, but nothing bad.

# Plotting 5 posterior predictive histograms/bar charts (not densities since Y is a discrete random variable here):

pp\_check(sparrow\_model, plotfun = "hist", nreps = 5) +

xlab("offspring")

# The predictive data sets match the observed data set pretty well ... Poisson model looks good.

# Plotting 50 posterior plausible models:

sparrow\_data %>%

add\_fitted\_draws(sparrow\_model, n = 50) %>%

ggplot(aes(x = x, y = y, color = urban)) +

geom\_line(aes(y = .value, group = paste(urban, .draw)),

alpha = .1) +

geom\_point(data = sparrow\_data, size = 0.1)

# Summary of point estimates and interval estimates for the betas:

tidy(sparrow\_model, conf.int = TRUE, conf.level = 0.95)

# Posterior prediction of offspring of 3-yr-old rural sparrow:

example\_prediction <- posterior\_predict(

sparrow\_model, newdata = data.frame(x=3,xsq=9,urban==0))

table(example\_prediction)

# A posterior predictive distribution for this individual:

round(table(example\_prediction)/length(example\_prediction),3)

# Posterior predictive mean for this individual:

post.pred.mean <- as.numeric(

as.numeric(row.names(table(example\_prediction))) %\*% as.numeric(table(example\_prediction)/length(example\_prediction))

)

print(post.pred.mean)

# Histogram of posterior predictive distribution for this individual:

mcmc\_hist(example\_prediction, binwidth = 1) +

xlab("Predicted number of offspring of 3-yr-old rural sparrow")

# Evaluating model fit:

poisson\_predictions <- posterior\_predict(sparrow\_model, newdata = sparrow\_data)

# Plot the posterior predictive models for each student:

ppc\_intervals\_grouped(sparrow\_data$y, yrep = poisson\_predictions,

x = sparrow\_data$x,

group = sparrow\_data$urban,

prob = 0.5, prob\_outer = 0.95,

facet\_args = list(scales = "fixed"))

# Do the dark blue circles (observed counts) fall near the light blue circles (predicted counts)?

# Do most of the dark blue circles (observed counts) fall within the 95% prediction intervals?

# A numerical summary of in-sample model fit:

prediction\_summary(model = sparrow\_model, data = sparrow\_data)

# A (better?) numerical summary of out-of-sample model fit:

poisson\_cv <- prediction\_summary\_cv(model = sparrow\_model,

data = sparrow\_data, k = 10)

poisson\_cv$cv

# Could fit a different model (like without x1? or with an interaction term?)

# and compare MAE's across models ... smaller would be better.

**#########################################################################**

**## Problem 6:**

**#########################################################################**

**#########################################################################**

**#**

**#(a)**

**pnorm(6,mean=5,sd=5)**

**#[1] 0.5792597**

**#**

**pnorm(6,mean=5,sd=5)/(1-pnorm(6,mean=5,sd=5))**

**# 1.376763**

**#**

**# The posterior probability that the mean is less than 6 is 1.38 times the**

**# posterior probability that the mean is not less than 6.**

**#**

**#########################################################################**

**#########################################################################**

**#**

**#(b)**

**pnorm(6,mean=10,sd=10)/(1-pnorm(6,mean=10,sd=10))**

**#[1] 0.5257352**

**#**

**# The prior probability that the mean is less than 6 is 0.53 times the**

**# prior probability that the mean is not less than 6.**

**#**

**#########################################################################**

**#**

**#**

**#(c)**

**( pnorm(6,mean=5,sd=5)/(1-pnorm(6,mean=5,sd=5)) )/( pnorm(6,mean=10,sd=10)/(1-pnorm(6,mean=10,sd=10)) )**

**#[1] 2.618739**

**#**

**# The odds of the alternative hypothesis after seeing the data are 2.6 times**

**# greater than the odds of the alternative hypothesis before seeing the data.**

**# Seeing the data increased the odds that mu is less than 6 by a factor of 2.6.**

**#**

**#########################################################################**

**#########################################################################**

**#**

**#(d)**

**( (1-pnorm(6,mean=5,sd=5))/pnorm(6,mean=5,sd=5) )/( (1-pnorm(6,mean=10,sd=10))/pnorm(6,mean=10,sd=10) )**

**# [1] 0.3818632**

**#**

**#########################################################################**