# Biostat234_Lab 5

Name: Huiyu Hu

```
#LOAD NECESSARY PACKAGES
setwd("/Users/huiyuhu/Desktop/Study/UCLA_Biostat/BIOSTAT234/lab/Lab 5")
library(R2jags)
```

```
## Loading required package: rjags
```

```
## Loading required package: coda
```

```
## Linked to JAGS 4.3.0
```

```
## Loaded modules: basemod,bugs
```

```
##
## Attaching package: 'R2jags'
```

```
## The following object is masked from 'package:coda':
##
##     traceplot
```

```
load("AddBurnin.RData")
library(lattice)
load("model1data.RData")
```

```
# sink("model1.txt")
# cat("
# # Response is number of Exencephaly malformations
# # out of the total number of live fetuses
# # in the litter.
# # Model #1.
# model
# {
#         for (i in 1:N)
#         {
#                 y[i] ~ dbin(p[i], n[i])
#                 logit(p[i]) <-  alpha[1] +alpha[2]*(z[i]-695) + beta[i] +
#                                 x1[i]*alpha[3] +x2[i]*alpha[4]
#                 beta[i] ~ dnorm(0, tau)
#         }
#         for (j in 1:4)
#         {
#                 alpha[j] ~ dnorm(0, 0.001)
#         }
#         sigsq ~ dunif(0,6)
#         tau <- 1/sigsq
#         sigma <- sqrt(sigsq)
#   for(j in 1:npred)
#       {
#           lppred[j] <- alpha[1] + alpha[2]*(pred[j]-695) + betanew
#           logit(probpred[j]) <- lppred[j]
#       }
#       betanew ~ dnorm(0,tau)
# }
#     ",fill = TRUE)
#sink()

# Initial Values
model1inits = rep(
  list(list(alpha=c(0, 0, 0, 0), sigsq=1,
beta=c(0,0,0,1,0, 0,0,0,-1,0, 0,0,0,0,0, 0,0,0,0,0,
             0,0,0,0,0, 0,0,0,1,0, 0,0,0,0,0, 0,0,0,-1,0,
             0,0,0,0,1, 0,0,0,0,0, 0,0,0,0,0, 0,0,-1,0,0,
             0,0), betanew=0 )), 3)
```

```
# sink("model2.txt")
# cat("
# # Model #2
# model
# {
#         for (i in 1:N)
#         {
#                 y[i] ~ dbin(p[i], n[i])
#                 p[i] ~ dbeta(a[i],b)
#                 a[i] <- b * exp(alpha[1] +alpha[2]*(z[i]-695) +
#                         x1[i]*alpha[3] +x2[i]*alpha[4])
#         }
#         for (j in 1:4)
#         {
#                 alpha[j] ~dnorm(0, 0.001)
#         }
#         b ~ dgamma(ba,bb)
# }
#     ",fill = TRUE)
# sink()
```

```
model1parameters= c("alpha", "p", "probpred", "beta", "tau", "sigma")

#### Did you load the data yet???

run1=jags(model1data, model1inits, model1parameters, "model1.txt", n.chains=3, n.iter=11
000, n.burnin=0, n.thin=1, DIC=F)
```

```
## module glm loaded
```

```
## module dic loaded
```

```
## Compiling model graph
##    Resolving undeclared variables
##    Allocating nodes
## Graph information:
##    Observed stochastic nodes: 62
##    Unobserved stochastic nodes: 68
##    Total graph size: 673
##
## Initializing model
```

```
Output1=AddBurnin(run1$BUGSoutput$sims.array,burnin=1000,n.thin=1)
```

Lab 5 Homework

1. Print model 1 and on the printout, identify the following:

a. Where the likelihood is modeled;

```
logit(p[i]) <-  alpha[1] +alpha[2]*(z[i]-695) + beta[i] +  x1[i]*alpha[3] +x2[i]*alpha[4]
```

b. Where the priors are modeled;

```
for (j in 1:4)  {  alpha[j] ~ dnorm(0, 0.001)  }  sigsq ~ dunif(0,6)  tau <- 1/sigsq
sigma <- sqrt(sigsq)
```

c. Where the linear predictor is specified;

```
lppred[j] <- alpha[1] + alpha[2]*(pred[j]-695) + betanew
```

d. Where the predictions are modeled;

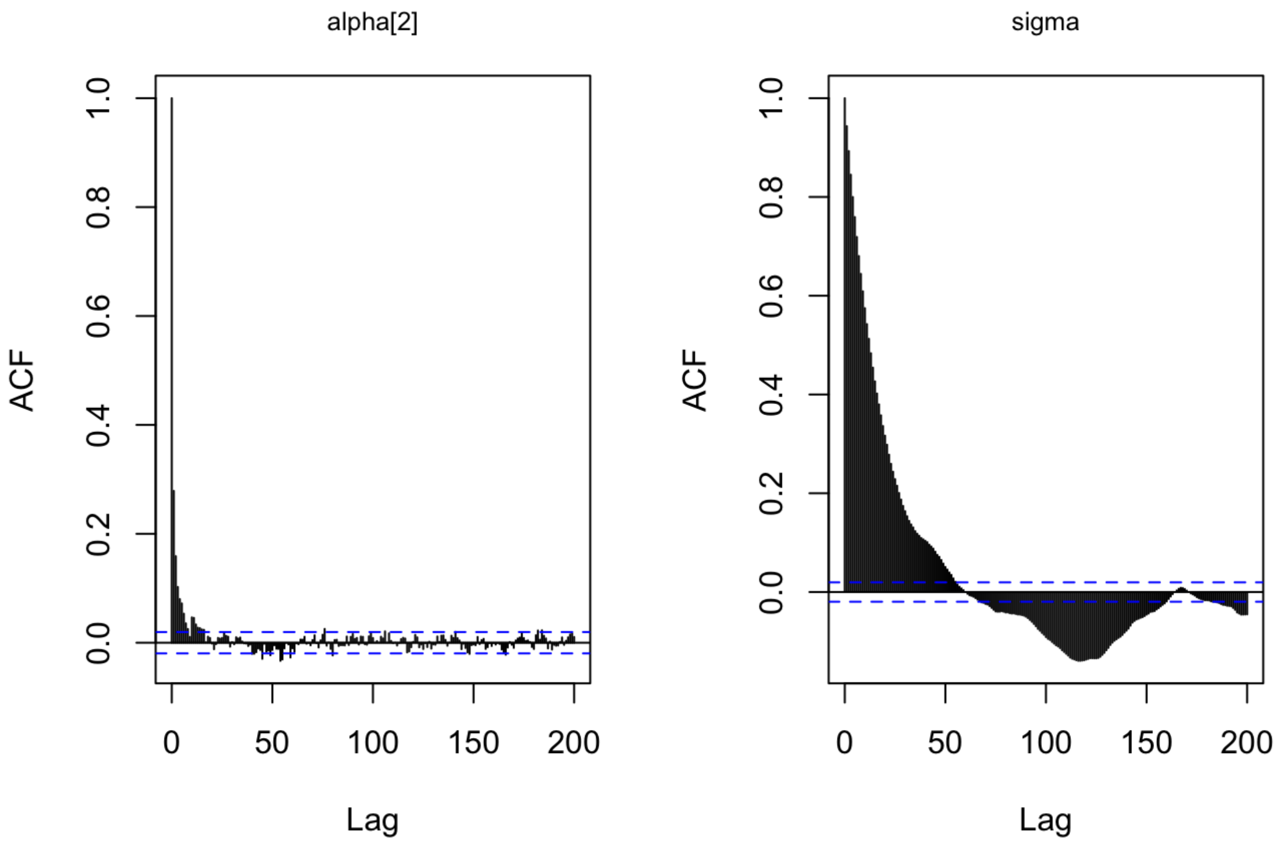
```
for(j in 1:npred)  {  lppred[j] <- alpha[1] + alpha[2]*(pred[j]-695) + betanew
logit(probpred[j]) <- lppred[j]  }  betanew ~ dnorm(0,tau)
```

e. Where what predictions we are interested in are given.

```
logit(probpred[j]) <- lppred[j]
```

2. Give time series plots and autocorrelation plots for sigma and for alpha[2]. Give an estimate of how many iterations are sufficient for good estimation.

```
par(mfrow=c(1,2))
acf(Output1$Burnin.sims.array[,1,2], main="", lag.max=200)
mtext("alpha[2]",side=3, line=1, cex=.8)
acf(Output1$Burnin.sims.array[,1,137], main="", lag.max = 200)
mtext("sigma", side=3, cex=.8, line=1)
```



- According to the plot, 100 iterations are sufficient for good estimation.

3. Which p[i] is largest? Discuss (1-2 sentences) how you decide this, as there is more than one way to determine this.

- I use which.max function to find the maximum from output. p[25] is the largest and the boxplots can also showed that p[25] is the largest.

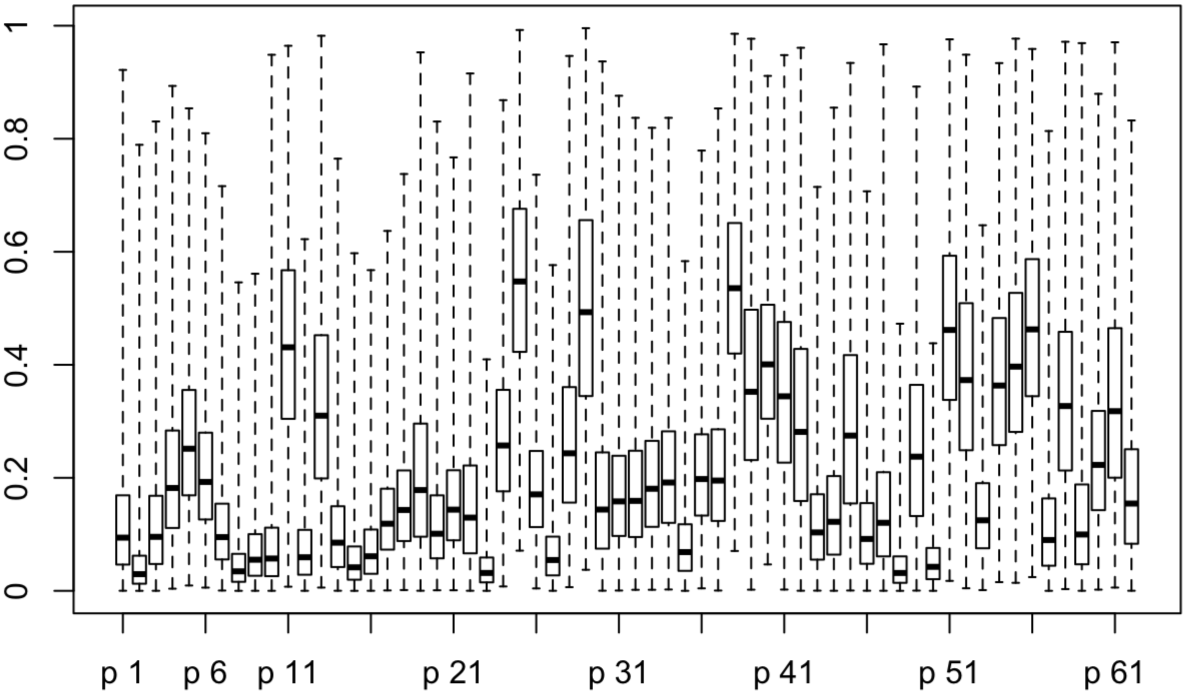
```
names <- paste0("p[", 1:62, "]")
p <- Output1$Burnin.Summary[names, ]
which.max(p[,1])
```

```
## p[25]
##    25
```

```
max(p[,1])
```

```
## [1] 0.5492187
```

```
#Boxplots for p
par(mfrow=c(1,1)) # sets plots in a 1x1 grid
#You may need to change the window size of the plot
boxplot(Output1$Burnin.sims.matrix[,c(seq(67,128))], range=0, axes=F)
axis(side=1,at=seq(1,62,5), labels=paste("p",seq(1,62,5)))
axis(side=2, at=0:5/5, labels=0:5/5)
box()
```



4. Which beta[i] is largest? Does it correspond to the same case as in Question 2? Explain: Do the largest beta[i] and the largest p[i] have to correspond to the same case [i]?

- By using which.max function and boxplot, beta[29] is the largest. No, the largest beta[i] and the largest p[i] do not necessarily have to correspond to the same case [i].

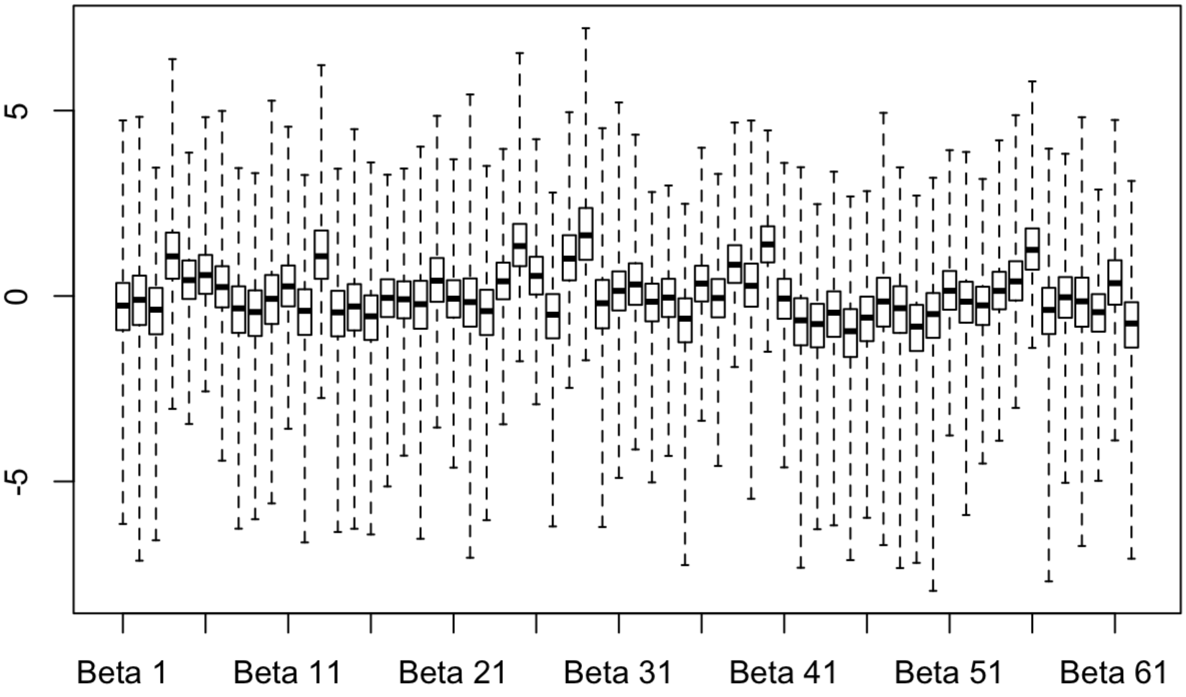
```
names <- paste0("beta[", 1:62, "]")
beta <- Output1$Burnin.Summary[names, ]
which.max(beta[,1])
```

```
## beta[29]
##       29
```

```
max(beta[,1])
```

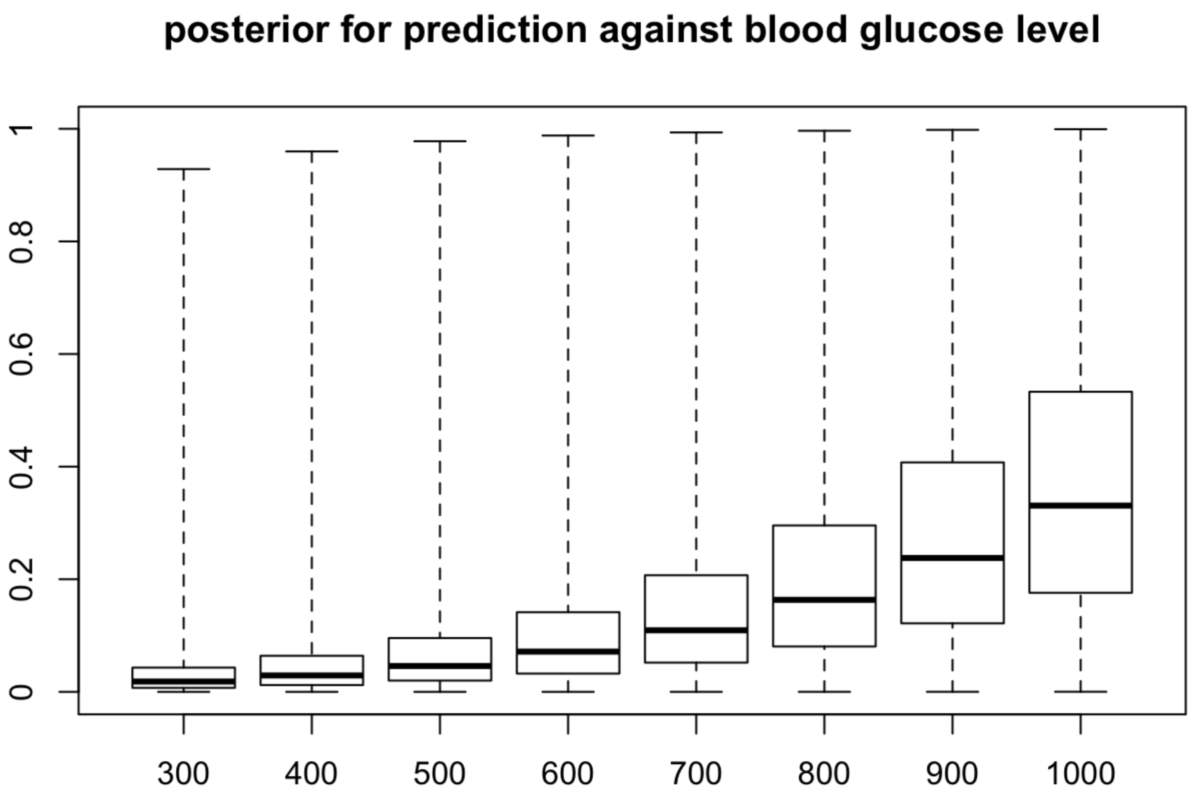
```
## [1] 1.725273
```

```
#Boxplots for Beta ## should include in the hw2
boxplot(Output1$Burnin.sims.matrix[,c(seq(5,66))], range=0, axes=F)
axis(side=1,at=seq(1,62,5), labels=paste("Beta",seq(1,62,5)))
axis(side=2, at=c(-5,0,5), labels=c(-5,0,5))
box()
```



5. Plot posterior estimates of (.05, median, .95) for each prediction against blood glucose level. Connect the dots between neighboring glucose levels. Label and document your plot carefully.

```
#Plot of the predictions at each maternal blood glucose level
boxplot(Output1$Burnin.sims.matrix[,129:136], range=0, axes=F)
axis(side=1,at=1:8, labels=3:10*100)
axis(side=2, at=0:5/5, labels=0:5/5)
title(main = 'posterior for prediction against blood glucose level')
box()
```



```
names <- paste0("probpred[", 1:8, "]")
glucose.level <- c(300,400,500,600,700,800,900,1000)
table1 <- cbind(glucose.level, round(Output1$Burnin.Summary[names, ], 4))
knitr::kable(table1, caption = "Posterior of for each prediction against blood glucose l
evel")
```

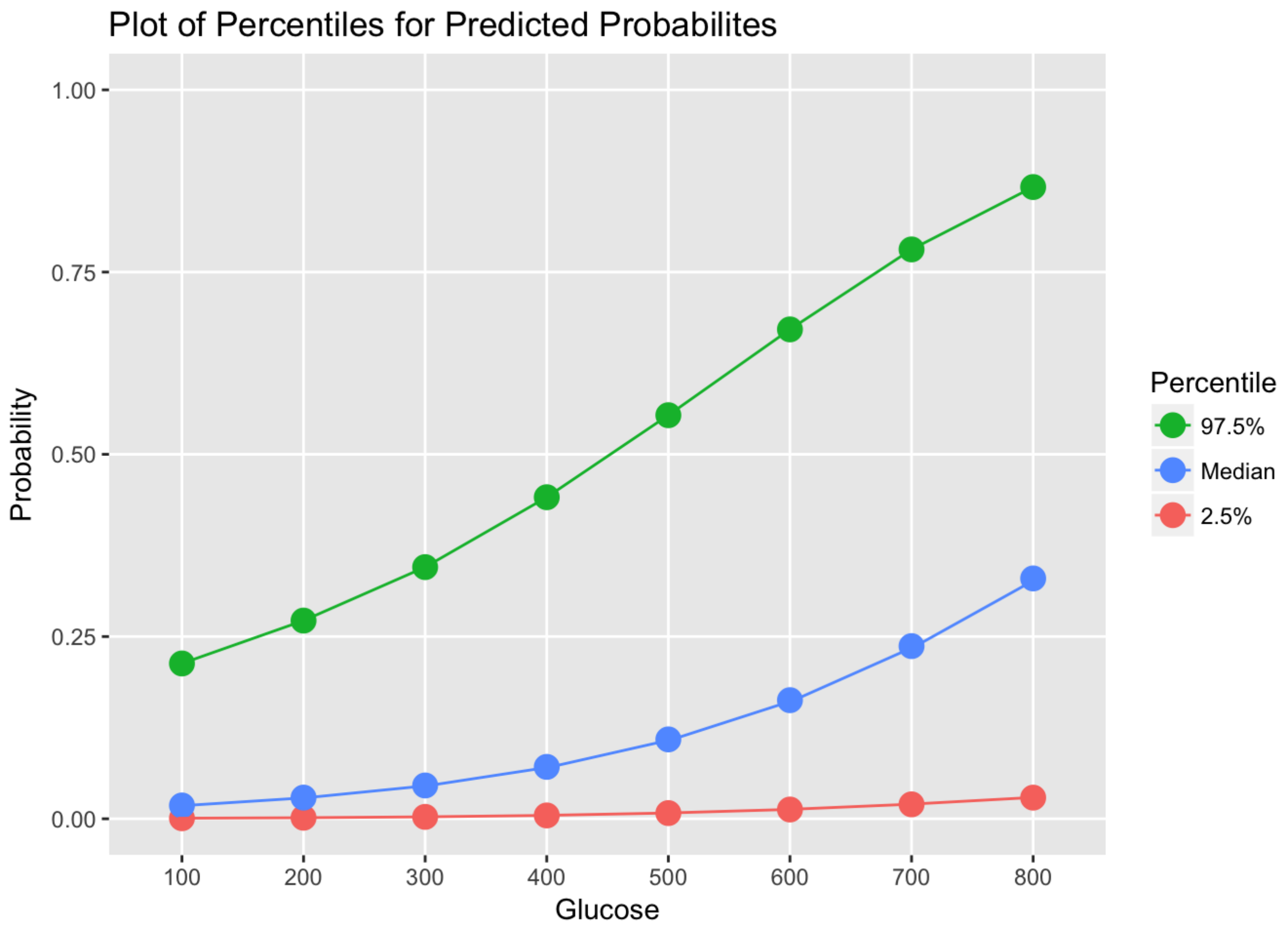
Posterior of for each prediction against blood glucose level

| | glucose.level | mu.vect | sd.vect | 2.5% | 97.5% | P>0 |
|---|---:|---:|---:|---:|---:|---:|
| probpred[1] | 300 | 0.0391 | 0.0644 | 0.0008 | 0.2145 | 1 |
| probpred[2] | 400 | 0.0545 | 0.0776 | 0.0015 | 0.2727 | 1 |
| probpred[3] | 500 | 0.0769 | 0.0949 | 0.0027 | 0.3452 | 1 |
| probpred[4] | 600 | 0.1089 | 0.1170 | 0.0047 | 0.4415 | 1 |
| probpred[5] | 700 | 0.1533 | 0.1441 | 0.0079 | 0.5554 | 1 |
| probpred[6] | 800 | 0.2118 | 0.1749 | 0.0127 | 0.6718 | 1 |
| probpred[7] | 900 | 0.2842 | 0.2063 | 0.0198 | 0.7833 | 1 |
| probpred[8] | 1000 | 0.3671 | 0.2339 | 0.0289 | 0.8679 | 1 |

```
#Extract the 95% CI and median from our BUGS summary table
r1<-as.matrix(run1$BUGSoutput$summary)
r1 <-r1[129:136,c(3,5,7)]

# Restructure the data so it's easy to plot
r2 <- data.frame(seq(100,800,100),rep(c("2.5%","Median","97.5%"),each=8),c(r1[,1],r1[,2
],r1[,3]))
names(r2) <- c("Predicted","Percentile","Value")
r2$Predicted <- as.factor(r2$Predicted)

# Use ggplot2 to make it fancy
library(ggplot2)
ggplot(data=r2,aes(x=Predicted,y=Value,colour=Percentile)) + geom_point(size = 4) +
  geom_line(position=position_dodge(.1),aes(group=Percentile)) +
  theme(panel.grid.minor=element_blank()) +
  labs(title = "Plot of Percentiles for Predicted Probabilites",
       x = "Glucose", y = "Probability") +
  scale_colour_discrete(breaks=c("97.5%","Median","2.5%")) + ylim(0,1.0)
```



Extra credit. 6. Repeat questions 1-4 for model 2. One question does not make sense. Which is it? Why doesn't it make sense?

- There is no probpred in model 2.

```
load("model2data.RData")

# Initial values.
model2inits = rep(list(list(alpha=c(0,0,0,0),b=1,
p=c(.5,.5,.5,.4,.5, .5,.5,.5,.5,.5,
.5,.5,.5,.5,.5,.5,.5,.6,.5,.5,
.5,.5,.5,.4,.5,.5,.5,.5,.5,.5,
.5,.5,.5,.6,.5,.5,.5,.5,.5,.5,
.5,.5,.5,.4,.5,.5,.5,.5,.5,.5,
.5,.5,.5,.6,.5,.5,.5,.5,.5,.5, .5,.5))), 5)

#Rerun the model but with more iterations and a larger thinning parameter
run2=jags(model2data, model2inits, model1parameters, "model2.txt", n.chains=5, n.iter=20
1000, n.burnin=0, n.thin=20, DIC=F)
```

```
## Compiling model graph
##    Resolving undeclared variables
##    Allocating nodes
## Graph information:
##    Observed stochastic nodes: 62
##    Unobserved stochastic nodes: 67
##    Total graph size: 685
##
## Initializing model
```

```
## Warning in FUN(X[[i]], ...): Failed to set trace monitor for probpred
## Variable probpred not found
```

```
## Warning in FUN(X[[i]], ...): Failed to set trace monitor for beta
## Variable beta not found
```

```
## Warning in FUN(X[[i]], ...): Failed to set trace monitor for tau
## Variable tau not found
```

```
## Warning in FUN(X[[i]], ...): Failed to set trace monitor for sigma
## Variable sigma not found
```

```
Output2=AddBurnin(run2$BUGSoutput$sims.array,burnin=1000,n.thin=20)
```

a. Where the likelihood is modeled;

```
p[i] ~ dbeta(a[i],b)
```

b. Where the priors are modeled;

```
alpha[j] ~dnorm(0, 0.001)  b ~ dgamma(ba,bb)
```

c. Where the linear predictor is specified;

```
a[i] <- b * exp(alpha[1] +alpha[2]*(z[i]-695) + x1[i]*alpha[3] +x2[i]*alpha[4])
```

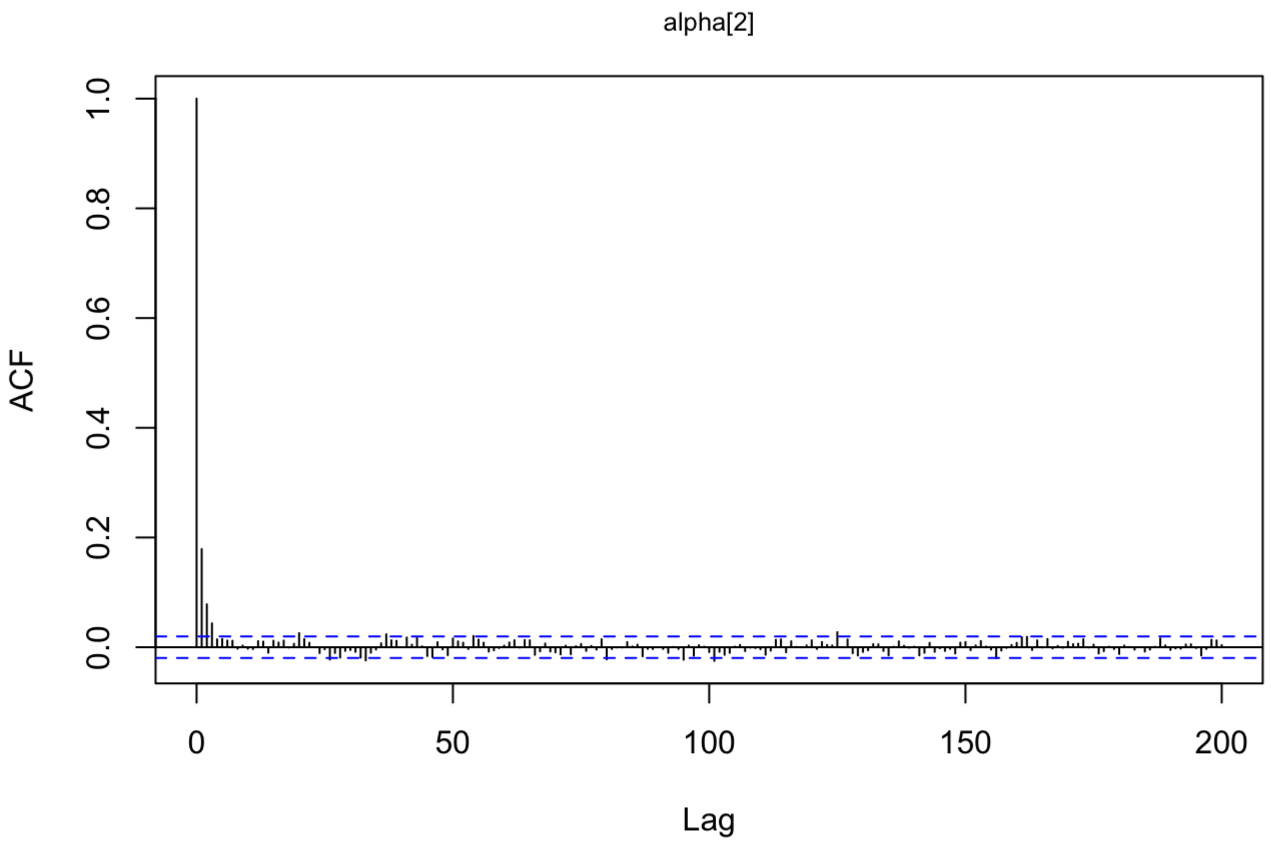
d. Where the predictions are modeled; (There is no prediction part in the model)

e. Where what predictions we are interested in are given. `p[i]`

Give time series plots and autocorrelation plots for sigma and for alpha[2]. Give an estimate of how many iterations are sufficient for good estimation.

- There is no sigma. But for alpha[2], 30 iterations are sufficient for good estimation.

```
acf(Output2$Burnin.sims.array[,1,2], main="", lag.max=200)
mtext("alpha[2]",side=3, line=1, cex=.8)
```



Which p[i] is largest? Discuss (1-2 sentences) how you decide this, as there is more than one way to determine this.

- p[38] is the largest.

```
names <- paste0("p[", 1:62, "]")
p <- Output2$Burnin.Summary[names, ]
which.max(p[,1])
```

```
## p[38]
##    38
```

```
max(p[,1])
```

```
## [1] 0.4720734
```

Which beta[i] is largest?

- beta[29] is the largest.

```
names <- paste0("beta[", 1:62, "]")
beta <- Output1$Burnin.Summary[names, ]
which.max(beta[,1])
```

```
## beta[29]
##       29
```

```
max(beta[,1])
```

```
## [1] 1.725273
```

7. The notes suggest possibly taking 1/(b+1) ~ uniform(0,1). Run this variation of model 2. Do this by defining c = 1/(b+1), then calculating b as a function of c. Provide your model code, the two posteriors for b (preferably on same plot, but not required), and the posterior summary info for parameter b. Indicate on your model code the parts 1a through 1e.