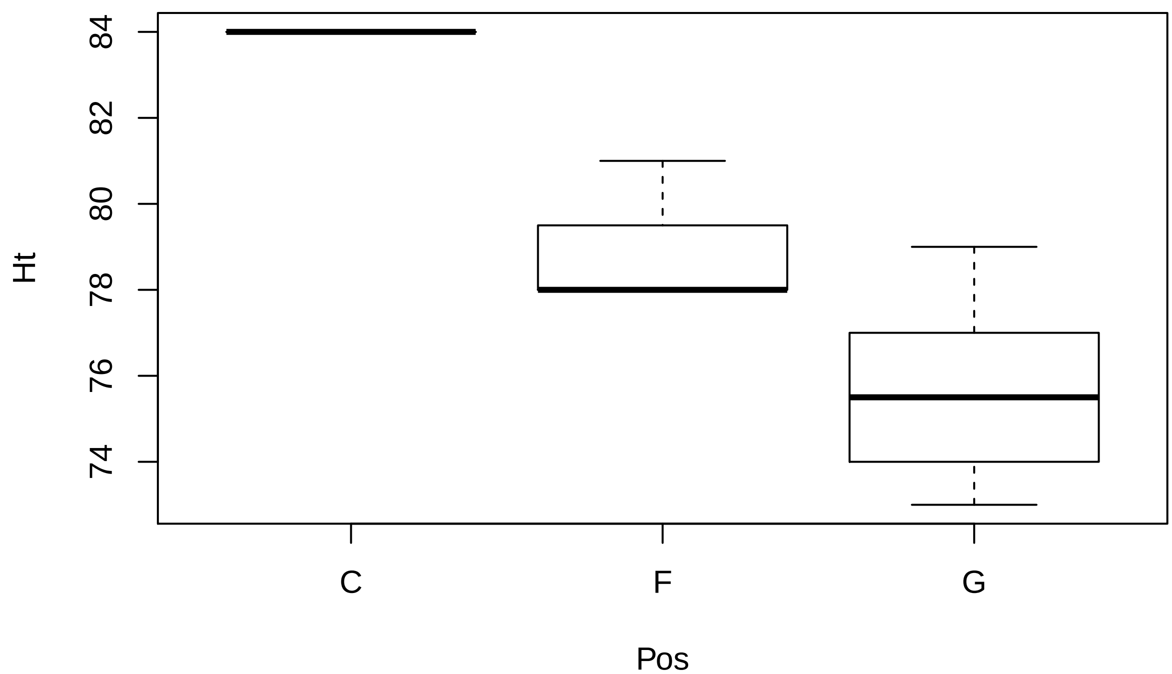
STAT 578 - Advanced Bayesian Modeling - Fall 2019
Assignment 6

Xiaoming Ji

Solution for Problem 1

```
perf_data = read.csv("illinimensbb.csv", header=TRUE)

plot(Ht ~ Pos, data= perf_data)
```



By checking the plot, we do see height and position are highly correlated. *center* has highest mean of height, *forward* has shortest mean of height and *forward* has in between these two. Their value ranges also don't seem to cross each other significantly.

Solution for Problem 2

(a)

```
model {
    for (i in 1:length(FGM)) {
        FGM[i] ~ dbin(prob[i], FGA[i])
```

```
        logit(prob[i]) <- beta_pos[Pos[i]] + beta_ht * Ht_Scaled[i]
        FGM_rep[i] ~ dbin(prob[i], FGA[i])
    }
    for (j in 1:max(Pos)) {
        beta_pos[j] ~ dt(0, 0.01, 1)
    }

    beta_ht ~ dt(0, 0.16, 1)
}
```

```
library(rjags)

df_jags_1 <- list( FGM = perf_data$FGM, FGA = perf_data$FGA,
                   Pos = unclass(perf_data$Pos),
                   Ht_Scaled = as.vector(scale(perf_data$Ht, scale=2*sd(perf_data$Ht))))

initial_vals_1 <- list(list(beta_pos = c(10,10,10), beta_ht=10),
                       list(beta_pos = c(10,10,-10), beta_ht=-10),
                       list(beta_pos = c(10,-10,10), beta_ht=-10),
                       list(beta_pos = c(10,-10,-10), beta_ht=10))

model_1 <- jags.model("perf_1.bug", df_jags_1, initial_vals_1, n.chains = 4,
                      n.adapt = 1000)
update(model_1, 1000)
x1 <- coda.samples(model_1, c("beta_pos","beta_ht","prob","FGM_rep"),
                              n.iter = 1000)
```

```
gelman.diag(x1, autoburnin=FALSE, multivariate = FALSE)
```

```
## Potential scale reduction factors:
##
##              Point est. Upper C.I.
## FGM_rep[1]         1.00       1.00
## FGM_rep[2]         1.00       1.00
## FGM_rep[3]         1.00       1.00
## FGM_rep[4]         1.00       1.01
## FGM_rep[5]         1.00       1.00
## FGM_rep[6]         1.00       1.00
## FGM_rep[7]         1.00       1.00
## FGM_rep[8]         1.00       1.00
## FGM_rep[9]         1.00       1.01
## FGM_rep[10]        1.00       1.00
## FGM_rep[11]        1.00       1.00
## FGM_rep[12]        1.00       1.01
## FGM_rep[13]        1.00       1.00
## FGM_rep[14]        1.00       1.01
## FGM_rep[15]        1.00       1.00
## beta_ht            1.01       1.02
## beta_pos[1]        1.01       1.02
## beta_pos[2]        1.00       1.01
## beta_pos[3]        1.01       1.02
## prob[1]            1.00       1.00
## prob[2]            1.00       1.01
## prob[3]            1.00       1.00
```

```
## prob[4]           1.00       1.01
## prob[5]           1.00       1.00
## prob[6]           1.00       1.01
## prob[7]           1.00       1.01
## prob[8]           1.00       1.01
## prob[9]           1.01       1.02
## prob[10]          1.00       1.01
## prob[11]          1.00       1.00
## prob[12]          1.00       1.01
## prob[13]          1.00       1.01
## prob[14]          1.00       1.00
## prob[15]          1.00       1.00
```

```
coef_sample_1 <- coda.samples(model_1, c("beta_pos","beta_ht","prob","FGM_rep"),
                              n.iter = 10000, thin = 5)
effectiveSize(coef_sample_1[,c("beta_pos[1]", "beta_pos[2]", "beta_pos[3]", "beta_ht")])
```

```
## beta_pos[1] beta_pos[2] beta_pos[3]     beta_ht
##    5816.727    6462.546    5452.309    4825.122
```

(b)

```
summary(coef_sample_1[, c("beta_pos[1]", "beta_pos[2]", "beta_pos[3]", "beta_ht")])
```

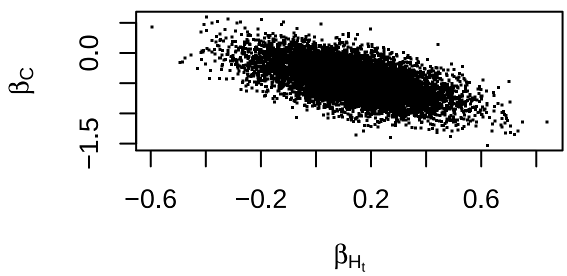
```
##
## Iterations = 3005:13000
## Thinning interval = 5
## Number of chains = 4
## Sample size per chain = 2000
##
## 1. Empirical mean and standard deviation for each variable,
##    plus standard error of the mean:
##
##                 Mean      SD  Naive SE Time-series SE
## beta_pos[1] -0.44833 0.28416 0.0031770      0.0037286
## beta_pos[2] -0.06084 0.11277 0.0012608      0.0014058
## beta_pos[3] -0.33532 0.07153 0.0007997      0.0009704
## beta_ht      0.13502 0.18077 0.0020211      0.0026066
##
## 2. Quantiles for each variable:
##
##                2.5%      25%      50%      75%   97.5%
## beta_pos[1] -1.0071 -0.63930 -0.44655 -0.25805  0.1001
## beta_pos[2] -0.2801 -0.13662 -0.05929  0.01493  0.1597
## beta_pos[3] -0.4765 -0.38287 -0.33468 -0.28749 -0.1955
## beta_ht     -0.2168  0.01184  0.13494  0.25505  0.4918
```

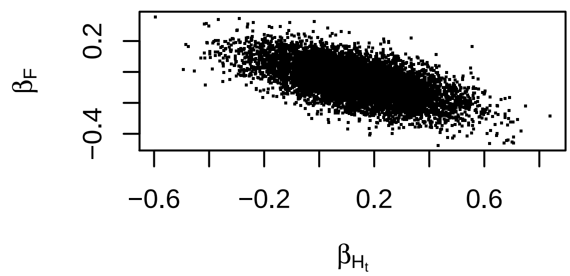
(c)

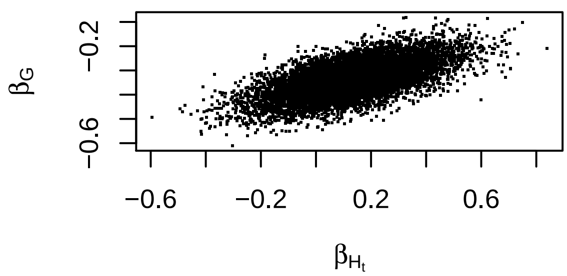
```
par(mfrow=c(2, 2))

plot(as.matrix(coef_sample_1)[,"beta_pos[1]"] ~ as.matrix(coef_sample_1)[,"beta_ht"],
```

```
     xlab = expression(paste(beta[H[t]])), ylab = expression(paste(beta[C])), pch='.')
plot(as.matrix(coef_sample_1)[,"beta_pos[2]"] ~ as.matrix(coef_sample_1)[,"beta_ht"],
     xlab = expression(paste(beta[H[t]])), ylab = expression(paste(beta[F])), pch='.')
plot(as.matrix(coef_sample_1)[,"beta_pos[3]"] ~ as.matrix(coef_sample_1)[,"beta_ht"],
     xlab = expression(paste(beta[H[t]])), ylab = expression(paste(beta[G])), pch='.')
```



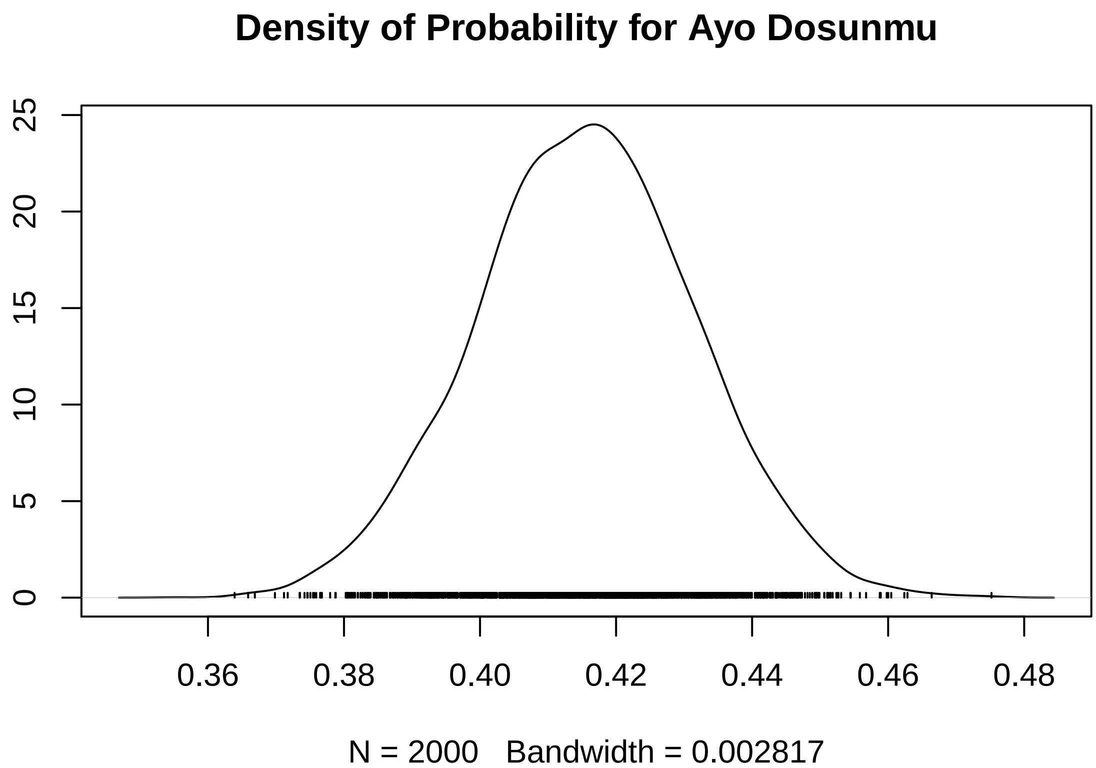




According to the plots, β_C, β_F, β_G are correlated with β_{H_t}.

(d)

```
Dosunmu_index = which(perf_data$X==11)
densplot(coef_sample_1[, paste("prob[",Dosunmu_index,"]",sep="")], main = "Density of Probability for A
```



(e)

Probability of $\beta_F > \beta_G$,

```
beta_F = as.matrix(coef_sample_1)[, "beta_pos[2]"]
beta_G = as.matrix(coef_sample_1)[, "beta_pos[3]"]
mean(beta_F > beta_G)
```

```
## [1] 0.961875
```

Bayes factor favoring $\beta_F > \beta_G$ versus $\beta_F < \beta_G$,

```
mean(beta_F > beta_G) / mean(beta_F < beta_G)
```

```
## [1] 25.22951
```

Give the bayes factor is between 20 to 150, we can say that the data has **Strong** evidence that $\beta_F > \beta_G$.

(f)

```
probs <- as.matrix(coef_sample_1)[, paste("prob[",1:nrow(perf_data),"]", sep="")]
FGM_rep <- as.matrix(coef_sample_1)[, paste("FGM_rep[",1:nrow(perf_data),"]", sep="")]


Tchi <- numeric(nrow(FGM_rep))
Tchirep <- numeric(nrow(FGM_rep))
```

```r
for(s in 1:nrow(FGM_rep)){
  Tchi[s] <- sum((perf_data$FGM - perf_data$FGA * probs[s,])^2 /
                   (perf_data$FGA * probs[s,] * (1 - probs[s,])))
  Tchirep[s] <- sum((FGM_rep[s,] - perf_data$FGA * probs[s,])^2 /
                      (perf_data$FGA * probs[s,] * (1 - probs[s,])))
 }

mean(Tchirep >= Tchi)
```

```
## [1] 0.046125
```

The posterior predictive p-value is suspiciously small, although not exceedingly so. Given we don't find any outliers, we conclude that there could be a bit of overdispersion.

(g)

(i)

```
model {
    for (i in 1:length(FGM)) {
        FGM[i] ~ dbin(prob[i], FGA[i])
        logit(prob[i]) <- beta_pos[Pos[i]] + beta_ht * Ht_Scaled[i] + epsilon[i]
        epsilon[i] ~ dnorm(0, 1 / sigma_epsilon^2)
        FGM_rep[i] ~ dbin(prob[i], FGA[i])
    }
    for (j in 1:max(Pos)) {
        beta_pos[j] ~ dt(0, 0.01, 1)
    }

    beta_ht ~ dt(0, 0.16, 1)
    sigma_epsilon ~ dunif(0,10)
}
```

```r
df_jags_2 <- list( FGM = perf_data$FGM, FGA = perf_data$FGA,
                   Pos = unclass(perf_data$Pos),
                   Ht_Scaled = as.vector(scale(perf_data$Ht, scale=2*sd(perf_data$Ht))))

initial_vals_2 <- list(list(beta_pos = c(10,10,10), beta_ht=10, sigma_epsilon = 0.01),
                       list(beta_pos = c(10,10,-10), beta_ht=-10, sigma_epsilon = 9),
                       list(beta_pos = c(10,-10,10), beta_ht=-10, sigma_epsilon = 0.01),
                       list(beta_pos = c(10,-10,-10), beta_ht=10, sigma_epsilon = 9))

model_2 <- jags.model("perf_2.bug", df_jags_2, initial_vals_2, n.chains = 4,
                      n.adapt = 1000)
update(model_2, 1000)
x2 <- coda.samples(model_2, c("beta_pos","beta_ht","prob","FGM_rep", "sigma_epsilon"),
                           n.iter = 2000)
```

```r
gelman.diag(x2, autoburnin=FALSE, multivariate = FALSE)
```

```
## Potential scale reduction factors:
##
##                  Point est. Upper C.I.
## FGM_rep[1]             1.00       1.01
```

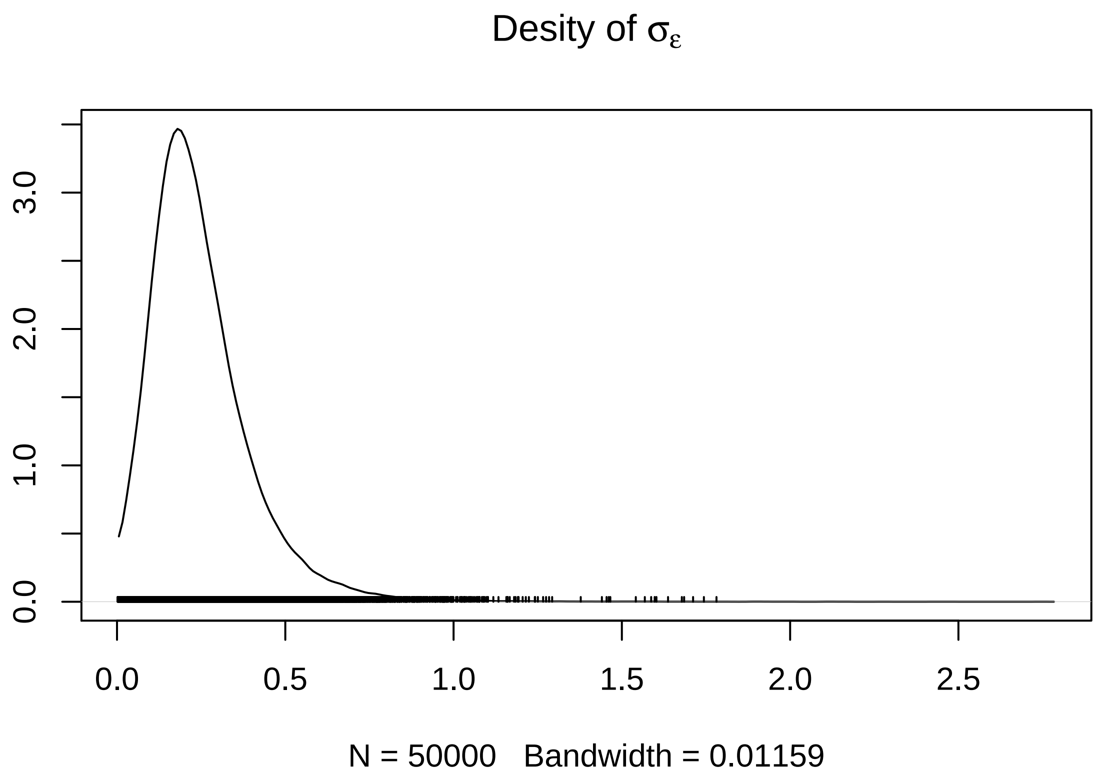
```
## FGM_rep[2]          1.00       1.00
## FGM_rep[3]          1.00       1.00
## FGM_rep[4]          1.00       1.00
## FGM_rep[5]          1.00       1.01
## FGM_rep[6]          1.00       1.00
## FGM_rep[7]          1.00       1.00
## FGM_rep[8]          1.00       1.00
## FGM_rep[9]          1.00       1.01
## FGM_rep[10]         1.01       1.02
## FGM_rep[11]         1.00       1.00
## FGM_rep[12]         1.00       1.01
## FGM_rep[13]         1.00       1.00
## FGM_rep[14]         1.00       1.00
## FGM_rep[15]         1.00       1.01
## beta_ht             1.03       1.08
## beta_pos[1]         1.02       1.05
## beta_pos[2]         1.02       1.04
## beta_pos[3]         1.04       1.13
## prob[1]             1.01       1.02
## prob[2]             1.00       1.01
## prob[3]             1.00       1.00
## prob[4]             1.00       1.00
## prob[5]             1.01       1.02
## prob[6]             1.00       1.01
## prob[7]             1.01       1.02
## prob[8]             1.01       1.02
## prob[9]             1.01       1.03
## prob[10]            1.02       1.05
## prob[11]            1.01       1.02
## prob[12]            1.01       1.03
## prob[13]            1.01       1.02
## prob[14]            1.01       1.02
## prob[15]            1.01       1.03
## sigma_epsilon       1.03       1.09
```

```
coef_sample_2 <- coda.samples(model_2, c("beta_pos","beta_ht","prob","FGM_rep", "sigma_epsilon"),
                              n.iter = 50000)
effectiveSize(coef_sample_2[,c("beta_pos[1]", "beta_pos[2]", "beta_pos[3]", "beta_ht", "sigma_epsilon")]
```

```
##   beta_pos[1]   beta_pos[2]   beta_pos[3]       beta_ht sigma_epsilon
##      6154.259      5159.298      6056.208      4013.969      3886.420
```

(ii)

```
densplot(coef_sample_2[, "sigma_epsilon"], main = expression(paste("Desity of ", sigma[epsilon])))
```



(iii)

```
beta_F = as.matrix(coef_sample_2)[, "beta_pos[2]"]
beta_G = as.matrix(coef_sample_2)[, "beta_pos[3]"]
mean(beta_F > beta_G)
```

```
## [1] 0.77747
```

This posterior probability is smaller than previous model.

```
mean(beta_F > beta_G) / mean(beta_F < beta_G)
```

```
## [1] 3.493776
```

This Bayes factor favoring $\beta_F > \beta_G$ versus $\beta_F < \beta_G$ is much smaller than previous model, and we can only say the data has **Positive** (between 3 to 30) evidence that $\beta_F > \beta_G$.

Also Chi-square discrepancy,

```
## [1] 0.38253
```

Thus we says no overdispersion problems for this model.

Solution for Problem 3

(a)

```
model {
    for (i in 1:length(BLK)) {
        BLK[i] ~ dpois(lambda[i])
        log(lambda[i]) <- log_MIN[i] + beta_pos[Pos[i]] + beta_ht * Ht_Scaled[i]
        BLK_rep[i] ~ dpois(lambda[i])
    }

    for (j in 1:max(Pos)) {
        beta_pos[j] ~ dnorm(0, 0.0001)
    }

    beta_ht ~ dnorm(0, 0.0001)
}
```

```
df_jags_3 <- list( BLK = perf_data$BLK,
                   Pos = unclass(perf_data$Pos),
                   log_MIN = log(perf_data$MIN),
                   Ht_Scaled = as.vector(scale(perf_data$Ht, scale=sd(perf_data$Ht))))

initial_vals_3 <- list(list(beta_pos = c(100,100,100), beta_ht=100),
                       list(beta_pos = c(100,100,-100), beta_ht=-100),
                       list(beta_pos = c(100,-100,100), beta_ht=-100),
                       list(beta_pos = c(100,-100,-100), beta_ht=100))

model_3 <- jags.model("perf_3.bug", df_jags_3, initial_vals_3, n.chains = 4,
                      n.adapt = 1000)
update(model_3, 1000)
x3 <- coda.samples(model_3, c("beta_pos","beta_ht", "lambda","BLK_rep"),
                              n.iter = 1000)
```

```
gelman.diag(x3, autoburnin=FALSE, multivariate = FALSE)
```

```
## Potential scale reduction factors:
##
##              Point est. Upper C.I.
## BLK_rep[1]         1.00       1.00
## BLK_rep[2]         1.00       1.00
## BLK_rep[3]         1.00       1.00
## BLK_rep[4]         1.00       1.00
## BLK_rep[5]         1.00       1.00
## BLK_rep[6]         1.00       1.00
## BLK_rep[7]         1.00       1.00
## BLK_rep[8]         1.00       1.00
## BLK_rep[9]         1.00       1.00
## BLK_rep[10]        1.00       1.00
## BLK_rep[11]        1.00       1.00
## BLK_rep[12]        1.00       1.01
## BLK_rep[13]        1.00       1.00
## BLK_rep[14]        1.00       1.00
## BLK_rep[15]        1.00       1.00
```

```
## beta_ht           1.01       1.03
## beta_pos[1]       1.01       1.03
## beta_pos[2]       1.01       1.03
## beta_pos[3]       1.00       1.01
## lambda[1]         1.00       1.00
## lambda[2]         1.01       1.02
## lambda[3]         1.00       1.00
## lambda[4]         1.00       1.01
## lambda[5]         1.01       1.02
## lambda[6]         1.01       1.02
## lambda[7]         1.00       1.01
## lambda[8]         1.01       1.02
## lambda[9]         1.01       1.03
## lambda[10]        1.00       1.01
## lambda[11]        1.00       1.00
## lambda[12]        1.01       1.02
## lambda[13]        1.00       1.00
## lambda[14]        1.01       1.02
## lambda[15]        1.00       1.01
```

```
coef_sample_3 <- coda.samples(model_3, c("beta_pos","beta_ht", "lambda","BLK_rep"),
                              n.iter = 20000, thin = 5)
effectiveSize(coef_sample_3[,c("beta_pos[1]", "beta_pos[2]", "beta_pos[3]", "beta_ht")])
```

```
## beta_pos[1] beta_pos[2] beta_pos[3]     beta_ht
##    4582.977    5071.813    9939.009    4337.851
```

(b)

```
summary(coef_sample_3[, c("beta_pos[1]", "beta_pos[2]", "beta_pos[3]", "beta_ht")])
```

```
##
## Iterations = 3005:23000
## Thinning interval = 5
## Number of chains = 4
## Sample size per chain = 4000
##
## 1. Empirical mean and standard deviation for each variable,
##    plus standard error of the mean:
##
##               Mean     SD Naive SE Time-series SE
## beta_pos[1] -5.298 0.6060 0.004791       0.008980
## beta_pos[2] -4.512 0.2863 0.002263       0.004023
## beta_pos[3] -4.450 0.1799 0.001422       0.001812
## beta_ht      1.006 0.2761 0.002183       0.004198
##
## 2. Quantiles for each variable:
##
##                2.5%     25%    50%    75%  97.5%
## beta_pos[1] -6.5083 -5.7022 -5.284 -4.888 -4.139
## beta_pos[2] -5.0852 -4.7022 -4.507 -4.314 -3.972
## beta_pos[3] -4.8192 -4.5683 -4.445 -4.327 -4.112
## beta_ht      0.4741  0.8156  1.000  1.190  1.559
```

(c)

```
beta_ht = as.matrix(coef_sample_3)[, "beta_ht"]
quantile(exp(beta_ht), c(0.025, 0.975))
```

```
##     2.5%    97.5%
## 1.606508 4.754442
```

The values within 95% central posterior credible interval are all greater than 1 and thus we can conclude that greater height is associated with a higher rate of blocking shots.

(d)

```
lambdas <- as.matrix(coef_sample_3)[, paste("lambda[",1:nrow(perf_data),"]", sep="")]
BLK_rep <- as.matrix(coef_sample_3)[, paste("BLK_rep[",1:nrow(perf_data),"]", sep="")]

Tchi <- numeric(nrow(BLK_rep))
Tchirep <- numeric(nrow(BLK_rep))

for(s in 1:nrow(BLK_rep)){
  Tchi[s] <- sum((perf_data$BLK - lambdas[s,])^2 / lambdas[s,])
  Tchirep[s] <- sum((BLK_rep[s,] - lambdas[s,])^2 / lambdas[s,])
 }

mean(Tchirep >= Tchi)
```

```
## [1] 0.00725
```

The posterior predictive p-value is extremely small. Thus this could could have overdispersion problem.

(e)

(i)

```
p_sample <- matrix(0, nrow = nrow(BLK_rep), ncol = nrow(perf_data))
for(s in 1:nrow(BLK_rep)){
  p_sample[s,] <- BLK_rep[s,] > perf_data$BLK
}

p = apply(p_sample, 2, mean)
p_df = data.frame(name=perf_data$Player, p_value=p)
p_df
```

```
##                       name   p_value
## 1  Bezhanishvili, Giorgi 0.5368125
## 2            Cayce, Drew 0.0580000
## 3      De La Rosa, Adonis 0.9857500
## 4           Dosunmu, Ayo 0.7021875
## 5          Feliz, Andres 0.8362500
## 6         Frazier, Trent 0.8174375
## 7          Griffin, Alan 0.0075625
## 8         Griffith, Zach 0.1808125
## 9          Jones, Tevian 0.9043750
## 10         Jordan, Aaron 0.1313750
```

```
## 11           Kane, Samba 0.0024375
## 12        Nichols, Kipper 0.2306875
## 13      Oladimeji, Samson 0.1804375
## 14       Underwood, Tyler 0.2654375
## 15     Williams, Da'Monte 0.0466875
```

(ii)

```
p_df[p_df$p_value < 0.05,]
```

```
##                  name   p_value
## 7       Griffin, Alan 0.0075625
## 11        Kane, Samba 0.0024375
## 15 Williams, Da'Monte 0.0466875
```

(iii)

```
p_df[p_df$p_value > 0.95,]
```

```
##                  name p_value
## 3 De La Rosa, Adonis 0.98575
```

Adonis played in center position and was 84 height. He played 225 minutes but was blocked only 1 shot. Similar to him, Samba also played in center position and was 84 height. For 86 minutes he played, 10 shots were blocked. Thus Adonis real performance is exceptional compared to our model. Thus the p-value is very high.