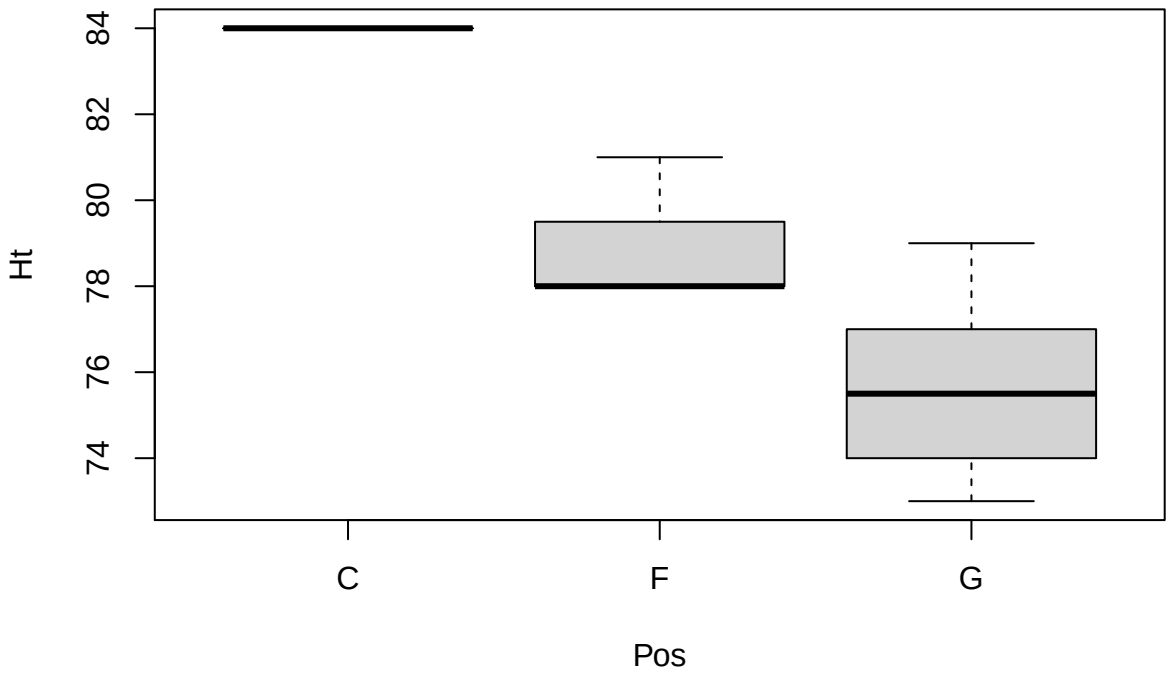
STAT 578 (Spring 2020) HW6 Solution

1. There is a relationship between Position and Height. Centers tend to be tallest, Forwards in the middle, and Guards tend to be shortest.

```
bb <- read.csv("~/UIUC/STAT578_20Spring/HW6/illinimensbb.csv",
               header = T)
bb$Pos = factor(bb$Pos)
plot(Ht ~ Pos, data=bb, xlab='Pos')
```



2. (a) JAGS model:

```
model {
  for (i in 1:length(FGM)) {
    FGM[i] ~ dbin(prob[i], FGA[i])
    logit(prob[i]) <- beta_P[Pos[i]] + beta_H * Htscaled[i]

    FGMrep[i] ~ dbin(prob[i], FGA[i])
  }

  for (j in 1:max(Pos)) {
    beta_P[j] ~ dt(0, 0.01, 1)
  }
  beta_H ~ dt(0, 0.16, 1)
}
```

R code:

```
library(rjags)
```

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

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

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

```
d1 <- list(FGM = bb$FGM,
           FGA = bb$FGA,
           Pos = unclass(bb$Pos),
           Htscaled = as.vector(scale(bb$Ht, scale=2*sd(bb$Ht))))
inits1 <- list(list(beta_H=10,  beta_P=c(10,10,10),
                    .RNG.name="base::Wichmann-Hill", .RNG.seed=12),
               list(beta_H=-10, beta_P=c(-10,10,10),
                    .RNG.name="base::Wichmann-Hill", .RNG.seed=34),
               list(beta_H=10,  beta_P=c(10,-10,10),
                    .RNG.name="base::Wichmann-Hill", .RNG.seed=56),
               list(beta_H=-10, beta_P=c(10,10,-10),
                    .RNG.name="base::Wichmann-Hill", .RNG.seed=78))
m1 <- jags.model("~/UIUC/STAT578_20Spring/HW6/m1.bug",
                 d1, inits1,
                 n.chains=4, n.adapt=1000)
```

```
## Compiling model graph
##    Resolving undeclared variables
##    Allocating nodes
## Graph information:
##    Observed stochastic nodes: 15
##    Unobserved stochastic nodes: 19
##    Total graph size: 110
##
## Initializing model
```

```
update(m1, 2000)  # burn-in
x1 <- coda.samples(m1, c("beta_H","beta_P","prob","FGMrep"),n.iter=10000)
gelman.diag(x1[,c("beta_H","beta_P[1]","beta_P[2]","beta_P[3]")],
            autoburnin=FALSE)
```

```
## Potential scale reduction factors:
##
##           Point est. Upper C.I.
## beta_H             1          1
## beta_P[1]          1          1
## beta_P[2]          1          1
## beta_P[3]          1          1
##
## Multivariate psrf
##
## 1
```

```
effectiveSize(x1[,c("beta_H","beta_P[1]","beta_P[2]","beta_P[3]")])
```

```
##    beta_H beta_P[1] beta_P[2] beta_P[3]
##  5696.005  8912.652  9029.307  7672.402
```

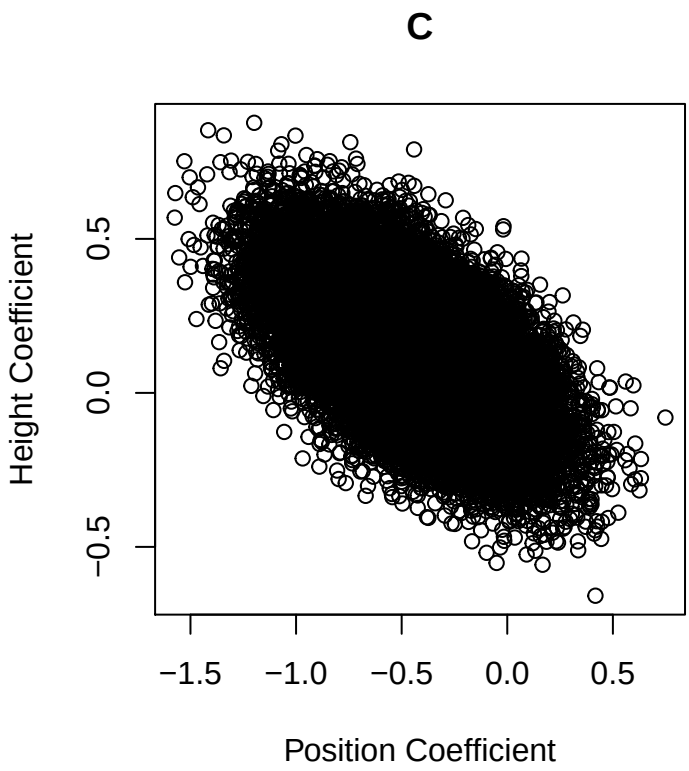
(b)

```
summary(x1[,c("beta_H","beta_P[1]","beta_P[2]","beta_P[3]")])
```

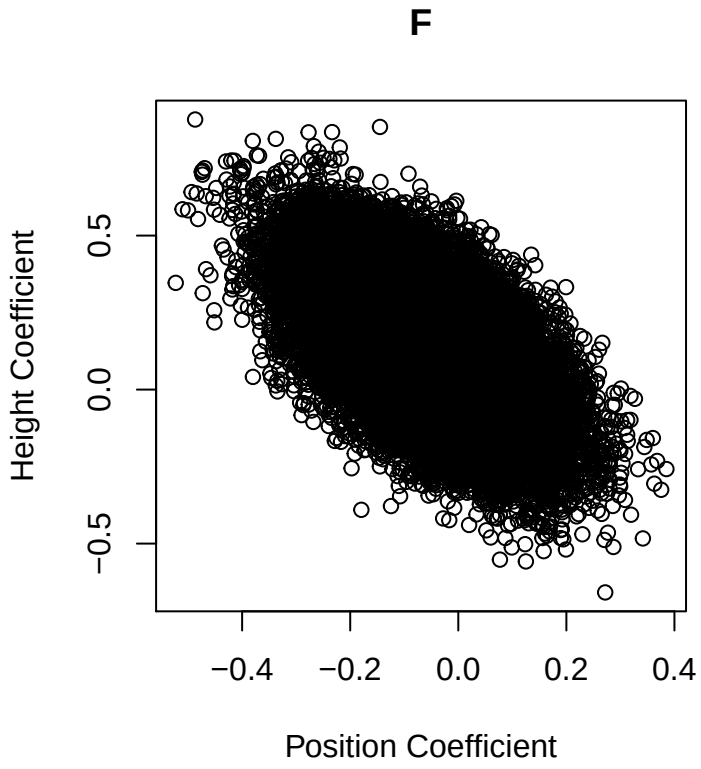
```
##
## Iterations = 3001:13000
## Thinning interval = 1
## Number of chains = 4
## Sample size per chain = 10000
##
## 1. Empirical mean and standard deviation for each variable,
##    plus standard error of the mean:
##
##                Mean      SD  Naive SE Time-series SE
## beta_H     0.13759 0.17799 0.0008899      0.0023667
## beta_P[1] -0.45331 0.28952 0.0014476      0.0030703
## beta_P[2] -0.06254 0.11193 0.0005597      0.0011828
## beta_P[3] -0.33380 0.07037 0.0003519      0.0008039
##
## 2. Quantiles for each variable:
##
##               2.5%      25%      50%      75%   97.5%
## beta_H    -0.2105  0.01816  0.13726  0.25698  0.4865
## beta_P[1] -1.0219 -0.64666 -0.45361 -0.25639  0.1072
## beta_P[2] -0.2819 -0.13782 -0.06314  0.01326  0.1571
## beta_P[3] -0.4715 -0.38171 -0.33357 -0.28620 -0.1954
```

(c) There are (posterior) correlations between each of the three Position coefficients and the Height coefficient.

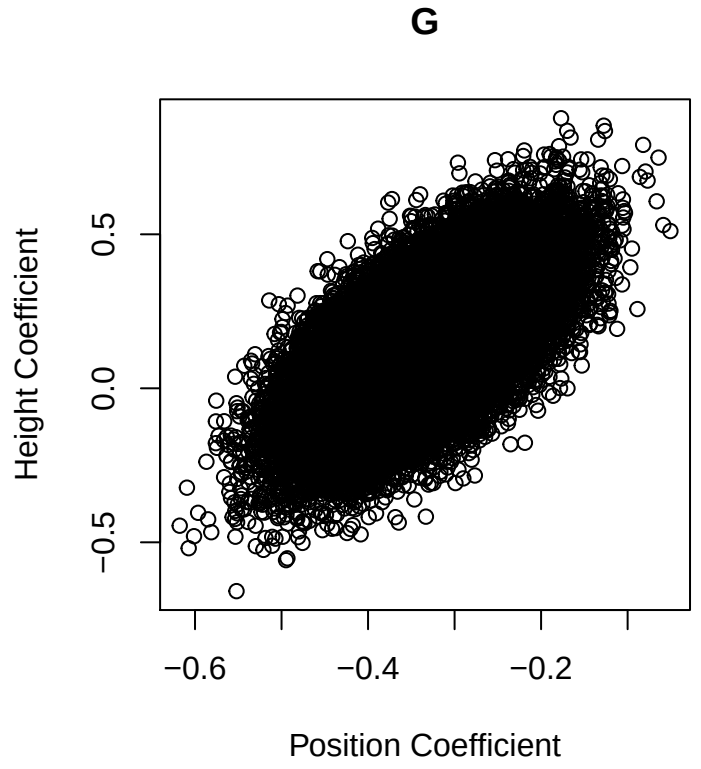
```
plot(as.matrix(x1)[,"beta_P[1]"], as.matrix(x1)[,"beta_H"],
     xlab='Position Coefficient', ylab='Height Coefficient', main='C')
```




```
plot(as.matrix(x1)[,"beta_P[2]"], as.matrix(x1)[,"beta_H"],
     xlab='Position Coefficient', ylab='Height Coefficient', main='F')
```

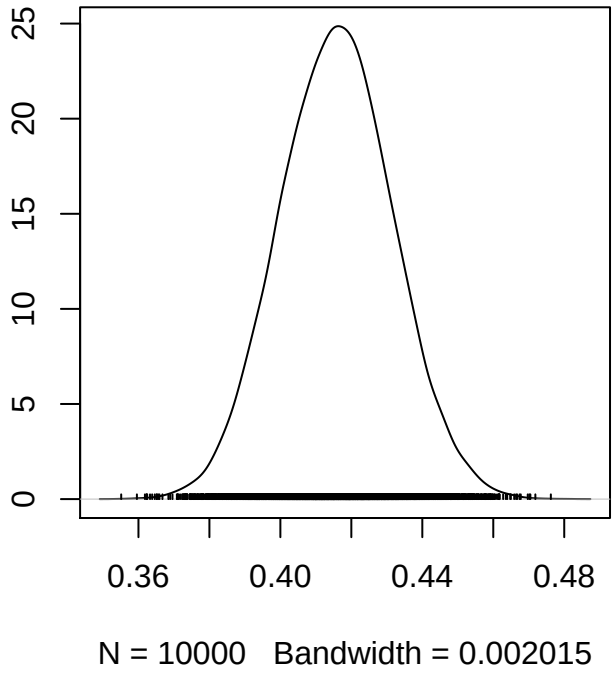



```
plot(as.matrix(x1)[,"beta_P[3]"], as.matrix(x1)[,"beta_H"],
     xlab='Position Coefficient', ylab='Height Coefficient', main='G')
```



(d)

```
densplot(x1[,"prob[4]"])
```



(e) $P(\beta_F > \beta_G \mid Data) = 0.96$.

Bayes factor favoring $\beta_F > \beta_G$ versus $\beta_F < \beta_G$: $\frac{P(\beta_F > \beta_G | Data)}{P(\beta_F < \beta_G | Data)} = 26.40$.

The data strongly supports the hypothesis that $\beta_F > \beta_G$.

```
mean(unlist(x1[,"beta_P[2]"]) > unlist(x1[,"beta_P[3]"]))
```

```
## [1] 0.9635
```

```
mean(unlist(x1[,"beta_P[2]"]) > unlist(x1[,"beta_P[3]"])) / (1 - mean(unlist(x1[,"beta_P[2]"]) > unlist
```

```
## [1] 26.39726
```

(f) Chi-square discrepancy posterior predictive p-value is less than 0.05, indicating evidence of overdispersion.

```
probs <- as.matrix(x1)[, paste("prob[",1:nrow(bb),"]", sep="")]
FGMrep <- as.matrix(x1)[, paste("FGMrep[",1:nrow(bb),"]", sep="")]
Tchi <- numeric(nrow(FGMrep))
Tchirep <- numeric(nrow(FGMrep))
for(s in 1:nrow(FGMrep)){
  Tchi[s] <- sum((bb$FGM - bb$FGA*probs[s,])^2 /
                   (bb$FGA*probs[s,]*(1-probs[s,])))
  Tchirep[s] <- sum((FGMrep[s,] - bb$FGA*probs[s,])^2 /
                      (bb$FGA*probs[s,]*(1-probs[s,])))
}
mean(Tchirep >= Tchi)
```

```
## [1] 0.04565
```

(g) (i) JAGS model:

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

```
    logit(prob[i]) <- beta_P[Pos[i]] + beta_H * Htscaled[i] + epsilon[i]
    epsilon[i] ~ dnorm(0, 1/sigmaepsilon^2)

    FGMrep[i] ~ dbin(prob[i], FGA[i])
  }

  for (j in 1:max(Pos)) {
    beta_P[j] ~ dt(0, 0.01, 1)
  }
  beta_H ~ dt(0, 0.16, 1)
  sigmaepsilon ~ dunif(0, 10)
}
```

R code:

```
d1n <- list(FGM = bb$FGM,
            FGA = bb$FGA,
            Pos = unclass(bb$Pos),
            Htscaled = as.vector(scale(bb$Ht, scale=2*sd(bb$Ht))))
inits1n <- list(list(beta_H=10,  beta_P=c(10,10,10), sigmaepsilon=0.01,
                     .RNG.name="base::Wichmann-Hill", .RNG.seed=12),
                list(beta_H=-10, beta_P=c(-10,10,10), sigmaepsilon=9,
                     .RNG.name="base::Wichmann-Hill", .RNG.seed=34),
                list(beta_H=10,  beta_P=c(10,-10,10), sigmaepsilon=0.01,
                     .RNG.name="base::Wichmann-Hill", .RNG.seed=56),
                list(beta_H=-10, beta_P=c(10,10,-10), sigmaepsilon=9,
                     .RNG.name="base::Wichmann-Hill", .RNG.seed=78))
m1n <- jags.model("~/UIUC/STAT578_20Spring/HW6/m1n.bug",
                  d1n, inits1n,
                  n.chains=4, n.adapt=1000)
```

```
## Compiling model graph
##    Resolving undeclared variables
##    Allocating nodes
## Graph information:
##    Observed stochastic nodes: 15
##    Unobserved stochastic nodes: 35
##    Total graph size: 142
##
## Initializing model
```

```
update(m1n, 2000)  # burn-in
x1n <- coda.samples(m1n, c("beta_H","beta_P","sigmaepsilon","prob","FGMrep"),n.iter=80000)
gelman.diag(x1n[,c("beta_H","beta_P[1]","beta_P[2]","beta_P[3]","sigmaepsilon")],
            autoburnin=FALSE)
```

```
## Potential scale reduction factors:
##
##              Point est. Upper C.I.
## beta_H                1          1
## beta_P[1]             1          1
## beta_P[2]             1          1
## beta_P[3]             1          1
## sigmaepsilon          1          1
##
```

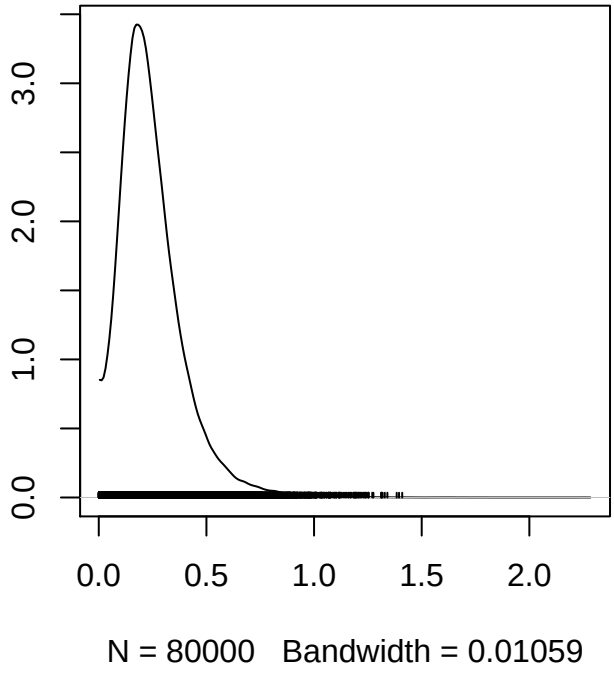
```
## Multivariate psrf
##
## 1
```

```
effectiveSize(x1n[,c("beta_H","beta_P[1]","beta_P[2]","beta_P[3]","sigmaepsilon")])
```

```
##       beta_H    beta_P[1]    beta_P[2]    beta_P[3] sigmaepsilon
##     5891.485     9422.249     7302.353     8285.246     5436.583
```

(ii)

```
densplot(x1n[,"sigmaepsilon"])
```



(iii) $P(\beta_F > \beta_G \mid Data) = 0.79$.

Bayes factor favoring $\beta_F > \beta_G$ versus $\beta_F < \beta_G$: $\frac{P(\beta_F > \beta_G | Data)}{P(\beta_F < \beta_G | Data)} = 3.67$.

The data still supports the hypothesis that $\beta_F > \beta_G$, but not as strongly as in part (e).

```
mean(unlist(x1n[,"beta_P[2]"]) > unlist(x1n[,"beta_P[3]"]))
```

```
## [1] 0.7859844
```

```
mean(unlist(x1n[,"beta_P[2]"]) > unlist(x1n[,"beta_P[3]"])) /
  (1 - mean(unlist(x1n[,"beta_P[2]"]) > unlist(x1n[,"beta_P[3]"])))
```

```
## [1] 3.672556
```

3. (a) JAGS model:

```
model {
  for (i in 1:length(BLK)) {
    BLK[i] ~ dpois(lambda[i])
    log(lambda[i]) <- logMIN[i] + beta_P[Pos[i]] + beta_H * Htscaled[i]

    BLKrep[i] ~ dpois(lambda[i])
  }
```

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

R code:

```
d2 <- list(BLK = bb$BLK,
           logMIN = log(bb$MIN),
           Pos = unclass(bb$Pos),
           Htscaled = as.vector(scale(bb$Ht)))

inits2 <- list(list(beta_H=100,  beta_P=c(100,100,100),
                    .RNG.name="base::Wichmann-Hill", .RNG.seed=12),
               list(beta_H=-100, beta_P=c(-100,100,100),
                    .RNG.name="base::Wichmann-Hill", .RNG.seed=34),
               list(beta_H=100,  beta_P=c(100,-100,100),
                    .RNG.name="base::Wichmann-Hill", .RNG.seed=56),
               list(beta_H=-100, beta_P=c(100,100,-100),
                    .RNG.name="base::Wichmann-Hill", .RNG.seed=78))

m2 <- jags.model("~/UIUC/STAT578_20Spring/HW6/m2.bug",
                 d2, inits2,
                 n.chains=4, n.adapt=1000)
```

```
## Compiling model graph
##    Resolving undeclared variables
##    Allocating nodes
## Graph information:
##    Observed stochastic nodes: 15
##    Unobserved stochastic nodes: 19
##    Total graph size: 120
##
## Initializing model
```

```
update(m2, 2000)  # burn-in
x2 <- coda.samples(m2, c("beta_H","beta_P","lambda","BLKrep"),n.iter=20000)
gelman.diag(x2[,c("beta_H","beta_P[1]","beta_P[2]","beta_P[3]")],
            autoburnin=FALSE)
```

```
## Potential scale reduction factors:
##
##           Point est. Upper C.I.
## beta_H             1          1
## beta_P[1]          1          1
## beta_P[2]          1          1
## beta_P[3]          1          1
##
## Multivariate psrf
##
## 1
```

```
effectiveSize(x2[,c("beta_H","beta_P[1]","beta_P[2]","beta_P[3]")])
```

```
##    beta_H beta_P[1] beta_P[2] beta_P[3]
```

```
##  4758.557  5503.947  5991.088 17194.479
```

(b)

```
summary(x2[,c("beta_H","beta_P[1]","beta_P[2]","beta_P[3]")])
```

```
##
## Iterations = 3001:23000
## Thinning interval = 1
## Number of chains = 4
## Sample size per chain = 20000
##
## 1. Empirical mean and standard deviation for each variable,
##    plus standard error of the mean:
##
##              Mean     SD  Naive SE Time-series SE
## beta_H     0.9964 0.2741 0.0009691       0.003988
## beta_P[1] -5.2739 0.6012 0.0021256       0.008146
## beta_P[2] -4.5008 0.2862 0.0010119       0.003724
## beta_P[3] -4.4532 0.1792 0.0006335       0.001374
##
## 2. Quantiles for each variable:
##
##              2.5%     25%     50%    75%  97.5%
## beta_H     0.4624  0.8128  0.9938  1.178  1.540
## beta_P[1] -6.4726 -5.6752 -5.2709 -4.863 -4.114
## beta_P[2] -5.0839 -4.6905 -4.4945 -4.305 -3.957
## beta_P[3] -4.8195 -4.5706 -4.4483 -4.331 -4.115
```

(c) Approximate 95% central posterior credible interval for $e^{\beta_{H_t}}$: $(1.59, 4.66)$ is above 1, so it seems that greater height is associated with a higher rate of blocking shots.

```
exp(summary(x2[,"beta_H"])$quantiles[c(1,5)])
```

```
##     2.5%    97.5%
## 1.587916 4.664208
```

(d) Chi-square discrepancy posterior predictive p-value is less than 0.05, indicating evidence of overdispersion.

```
lambdas <- as.matrix(x2)[, paste("lambda[",1:nrow(bb),"]", sep="")]
BLKrep <- as.matrix(x2)[, paste("BLKrep[",1:nrow(bb),"]", sep="")]
Tchi <- numeric(nrow(BLKrep))
Tchirep <- numeric(nrow(BLKrep))
for(s in 1:nrow(BLKrep)){
  Tchi[s] <- sum((bb$BLK - lambdas[s,])^2 / lambdas[s,])
  Tchirep[s] <- sum((BLKrep[s,] - lambdas[s,])^2 / lambdas[s,])
}
mean(Tchirep >= Tchi)
```

```
## [1] 0.0070875
```

(e) (i)

```
p_value <- c()
for(i in 1:length(bb$BLK)){
  p_value[i] <- mean(unlist(x2[, paste0("BLKrep[",i,"]")]) >= bb$BLK[i])
}
names(p_value) <- bb$Player
p_value
```

```
## Bezhanishvili, Giorgi           Cayce, Drew    De La Rosa, Adonis
##             0.5907625             1.0000000             0.9974375
##          Dosunmu, Ayo         Feliz, Andres        Frazier, Trent
##             0.7992375             0.9558875             0.9532750
##         Griffin, Alan       Griffith, Zach         Jones, Tevian
##             0.0216375             1.0000000             0.9770625
##         Jordan, Aaron          Kane, Samba       Nichols, Kipper
##             0.1938000             0.0048250             0.3286625
##     Oladimeji, Samson      Underwood, Tyler    Williams, Da'Monte
##             1.0000000             1.0000000             0.0901875
```

(ii)

```
p_value[p_value < 0.05]
```

```
## Griffin, Alan   Kane, Samba
##     0.0216375     0.0048250
```

(iii) The four players all had 0 blocks, and the Poisson model cannot predict any lower than 0, so this is not a surprising result. It is also not surprising that these players had 0 blocks, since they had very few minutes played.