

**Final Project Report**

**Bayesian Hierarchical Model Analysis**

**of NBA Player Statistics in 2009-2010**

**STAT 430 Applied Bayesian Analysis**

**Team Member: Kefu Zhu, Dajun Xu, Bin Xu, Kexin Fei**

**I. Background Introduction**

Nowadays, basketball is an increasingly popular sport around the world. The National Basketball Association is considered to be the pinnacle of professional basketball leagues. In NBA, each player belongs to a certain team and teams are divided into six divisions, called Atlantic, Central, Northwest, Pacific, Southeast, Southwest respectively. Owing to the existence of trading system, the managers of each team would like to know how is a player’s performance so they can assess his trade value and further strengthen the team by trading players with the other teams. The coaches also want to evaluate the technical and tactical skills of players and teams from statistical analysis. Evaluating the performance of teams and players is therefore an important task for each team. The purpose of this paper is to help determine whether division and team would affect players’ performance.

**II. Data Preparation**

The data is obtained from ‘NBAPlayerStatistics0910’ dataset of ‘SportsAnalytics’ library in R[[1]](#footnote-0). The original dataset contains records of 25 variables for 441 different NBA players. We planned to use conference and teams as two stages at the beginning. But since conference has only two levels (East and West), placing a distribution on parameters with only two levels might not be proper. Therefore, we choose to manually divide teams into six divisions according to NBA division record in 2009-2010[[2]](#footnote-1). Finally, we will use Player, Team, Division (newly created) as three stages in our project. The team category is the dataset while division category is collected from external source. We also regenerate the response variable Points Per Game(PPG), which is calculated from dividing the value of the toal points (TotalPoints) by the number of games played(GamesPlayed) in the dataset. Considering that each team has a maximum number of thirteen players in a game but the number of players is varying from teams (for example, Boston Celtics(BOS) has 15 players, while team Denver Nuggets(DEN) only has 13 players), we randomly select 13 individual players from each team. Performance regarding the players are made using the posterior distributions of the division and team parameters.

**III. Model Construction**

Bayesian Hierarchical Model is used in examination of whether the Points Per Game (PPG) is significantly different between teams in different divisions. Two different Bayesian Hierarchical Models are performed in this study.

The first one is normal hierarchical model where is the PPG of each player calculated from the dataset. We place a normal distribution to with as the mean parameter. and as variance parameter. can be deemed as the average PPG of all players in team j of division i. It follows a normal distribution with the and as parameters. Using the same notation, is the average PPG of all players of division i. follows a normal distribution with the and as parameters. The prior distributions of variance parameters , , follow a very thin inverse gamma distribution, and the prior distribution of population mean follows flat uniform distribution. The Directed Acyclic Graph of the normal model one is demonstrated in Figure 1.

|, ~ *N* (,)

|, ~ *N* (,)

|, ~ *N* (,)

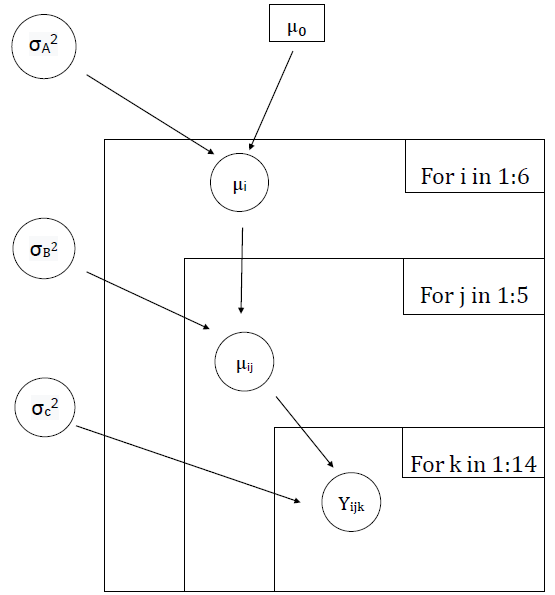
Prior:

~ 1 du

~ *IG* (0.001,0.001)

~ *IG* (0.001,0.001)

~ *IG* (0.001,0.001)



**Figure 1.** Directed acyclic graph for Normal Bayesian Hierarchical Model

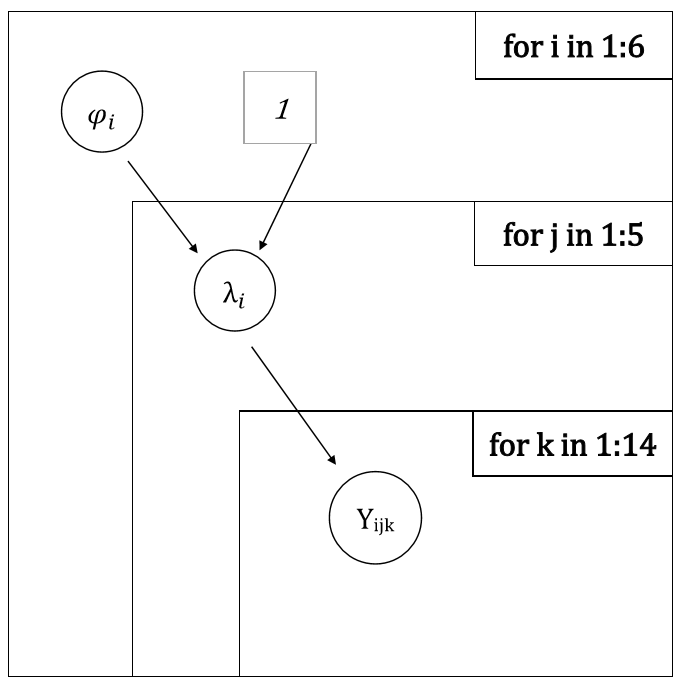
The second one is exponential hierarchical model where the PPG of each player follows exponential distribution with parameter , which follows gamma distribution with scale parameter 1 and shape parameter . The prior distributions of follows uniform distribution with range from 1 to 5. The DAG of exponential model is shown in Figure 2.

| ~ *Exp* ()

| ~ *Gamma* (, 1)

Prior:

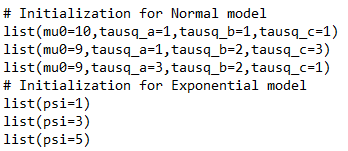
~ *Uniform* (1,5)



**Figure 2.** Directed acyclic graph for Gamma Bayesian Hierarchical Model

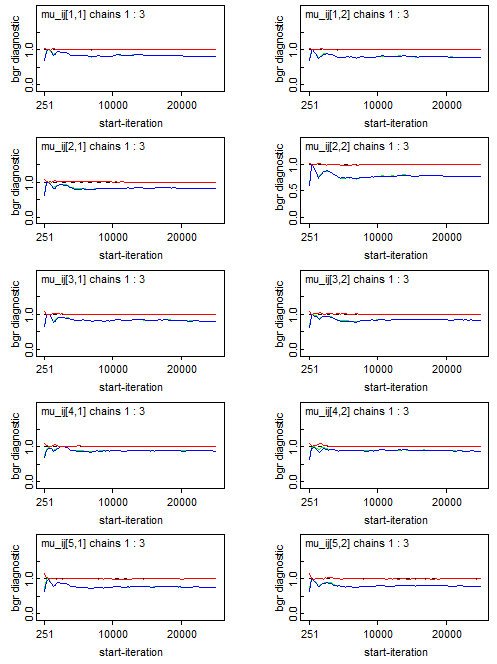
**IV. Experiment and Result**

Executions of both models are performed in OpenBUGS. Each model uses three chains in the simulation and the initialization of parameters in both models are shown in Figure 3.



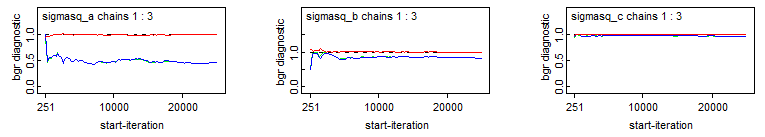
**Figure 3.** Initialization of parameters in OpenBUGS

In Normal Bayesian Hierarchical Model, we monitor the convergence using Brooks Gelman Rubin diagnostic (BGR). Figure 4 and Figure 5 are BGR plot with 50,000 iterations for four parameters (,,and ). After 10,000 iterations, the values of all parameters stabilize. Thus, this initial 10,000 iterations should be discarded as burn-in.



**Figure 4.** Partial BGR in OpenBUGS for

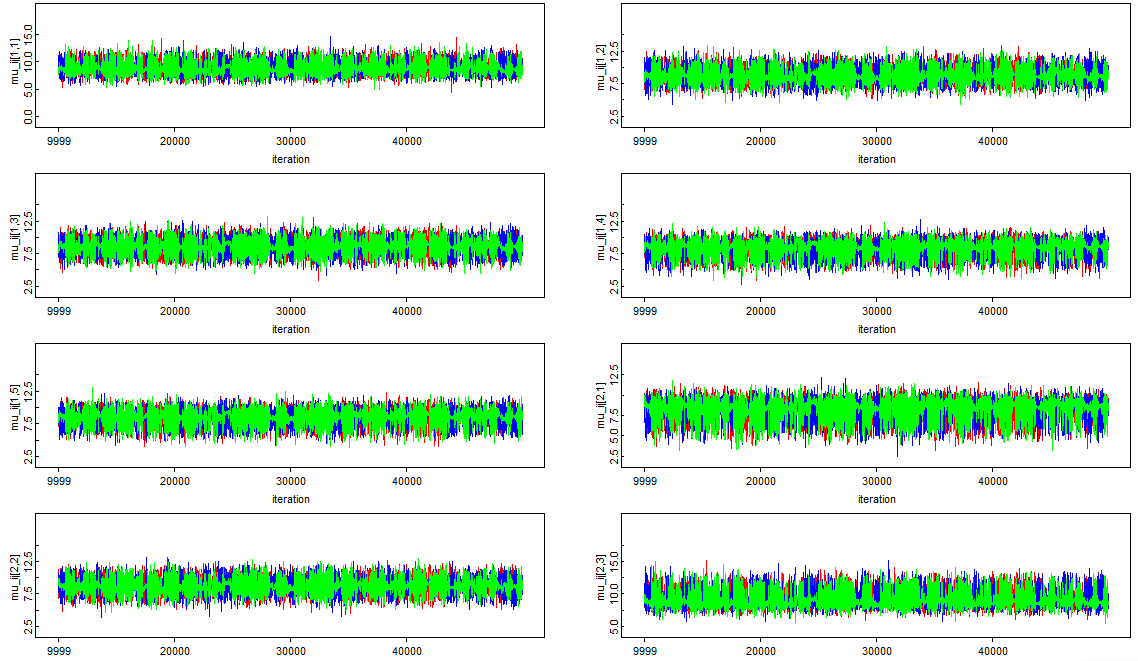
in Normal Bayesian Hierarchical Model



**Figure 5.** Output of Brooks Gelman Rubin Diagnostic in OpenBUGS for , and

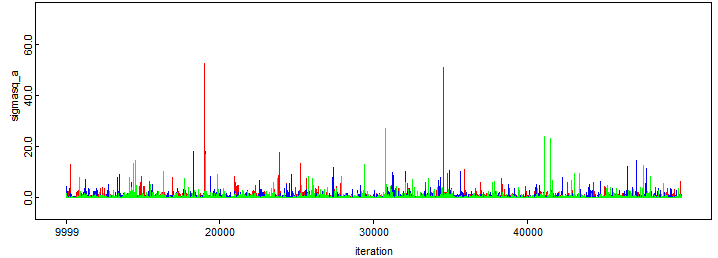
in Normal Bayesian Hierarchical Model

History plots give us preliminary run for four parameters (,,and ). To check if the parameters are stablizied after 10,000 iterations, we examine the plots for each parameter from 10,000 iteration to 50,000 iteration after burn-in phase. We can see that within 10,000 iterations, all three chains for each parameter came together and started drawing from the same range of values. See the following Figure 6, Figure 7, Figure 8, and Figure 9 for ,,and ) respectively.



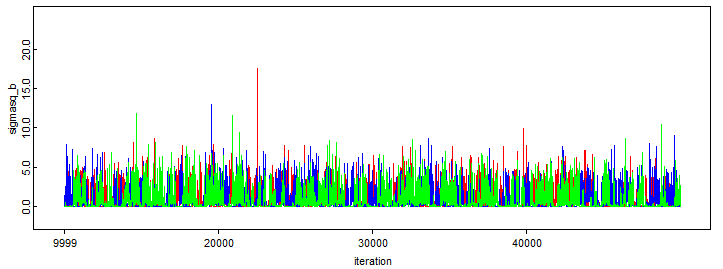
**Figure 6.** Sample History Diagnostic in OpenBUGS for

in Normal Bayesian Hierarchical Model

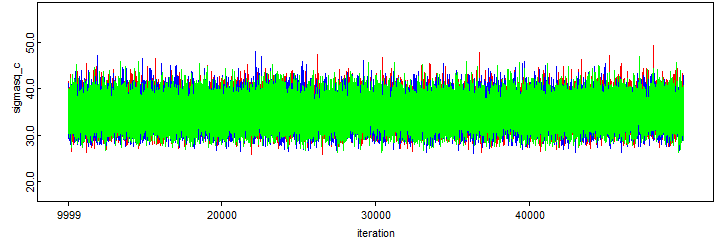


**Figure 7.** History Diagnostic in OpenBUGS for

in Normal Bayesian Hierarchical Model

**Figure 8.** Output of History Diagnostic in OpenBUGS for

in Normal Bayesian Hierarchical Model

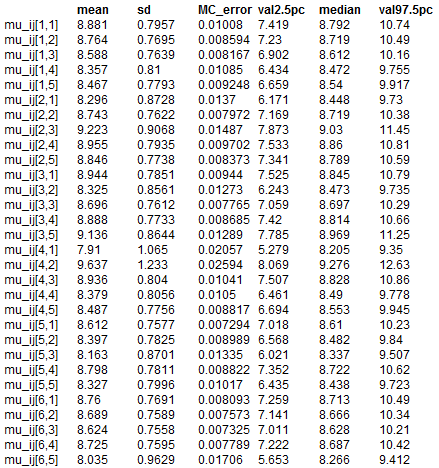


**Figure 9.** History Diagnostic in OpenBUGS for

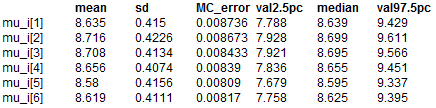
in Normal Bayesian Hierarchical Model

From the summary of estimators of the posterior distributions after performing 50,000 iterations we also excluded first 10,000 iterations. The posterior are within 7.9 and 9.3 and standard deviations are around 0.8. The MCMC error are very small and the posterior 95% credible intervals are given in Table 1 and Table 2. The differences between and standard deviation for for all i are even smaller.

Based on the simulation result, there is not any significant difference between the average Points Per Game of players in each team. We do not see any difference between the average Points Per Game of players in different divisions as well.



**Table 1.** Summary Results of Normal Model Estimates of Points Per Game on Team Level

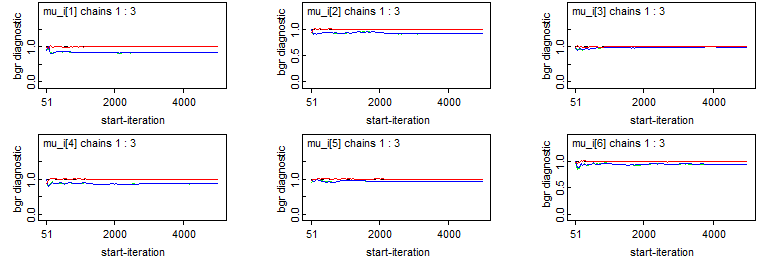


**Table 2.** Summary Results of Normal Model Estimates of Points Per Game on Division Level



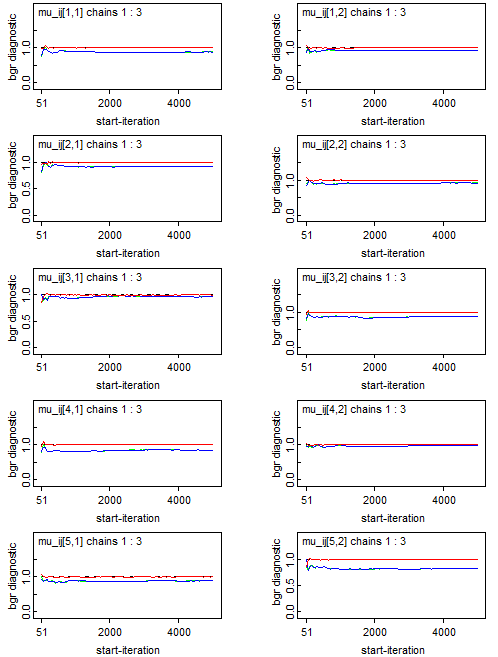
**Table 3.** Summary Results of Normal Model Estimates of , and

In Exponential Bayesian Hierarchical Model, because it seems like after 2,000 iterations all and converge, the initial 2,000 iterations should be discarded as burn-in (Figure 10, 11).



**Figure 10.** Partial Output of Brooks Gelman Rubin Diagnostic in OpenBUGS for

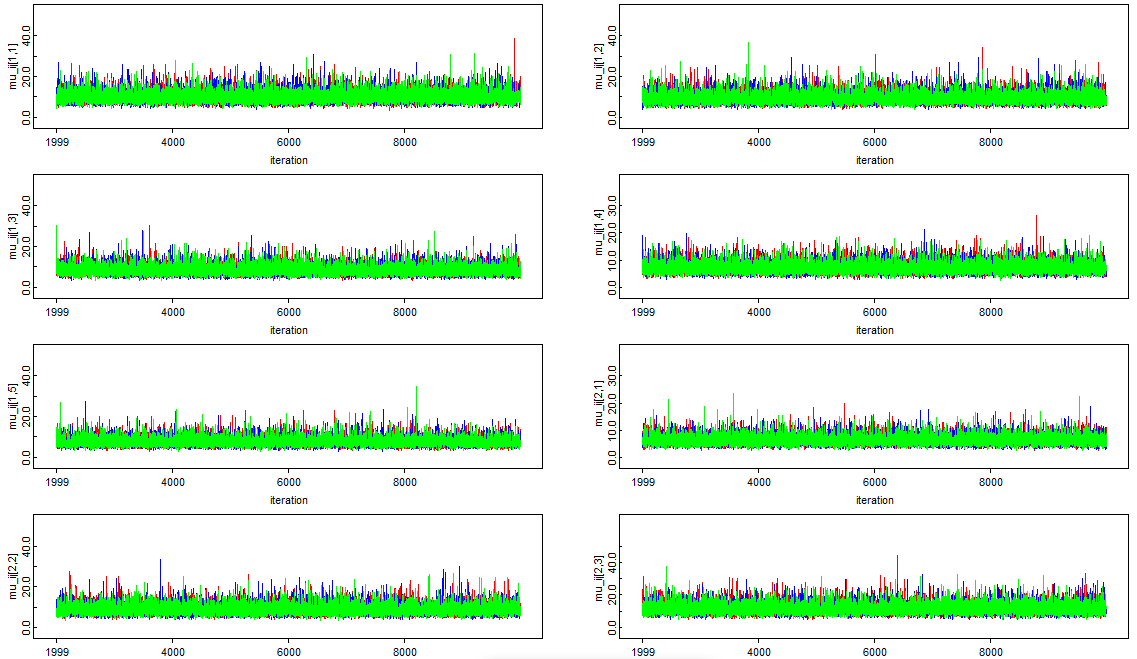
in Exponential Bayesian Hierarchical Model



**Figure 11.** Partial Output of Brooks Gelman Rubin Diagnostic in OpenBUGS for

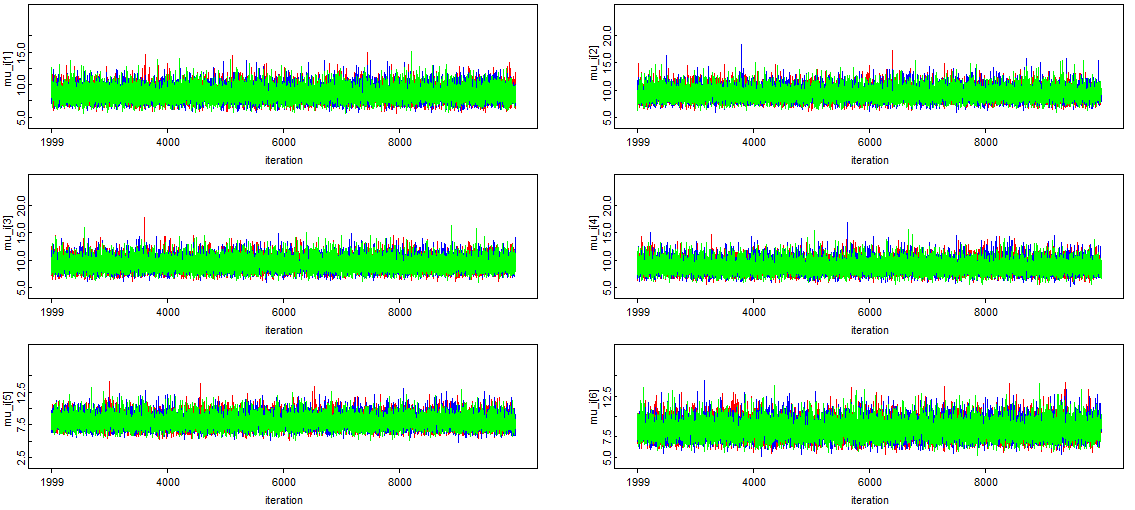
in Exponential Bayesian Hierarchical Model

Looking at History plot of sampler, History plots give us preliminary run for and . To check if the parameters are stablizied after 2,000 iterations, we examine the plots for each parameter from 2,000 iteration to 10,000 iteration after burn-in phase. We can see that within 8,000 iterations, all three chains for each parameter came together and started drawing from the same range of values. See the following Figure 12 and Figure 13 for and respectively.



**Figure 12.** Output of History Diagnostic in OpenBUGS for

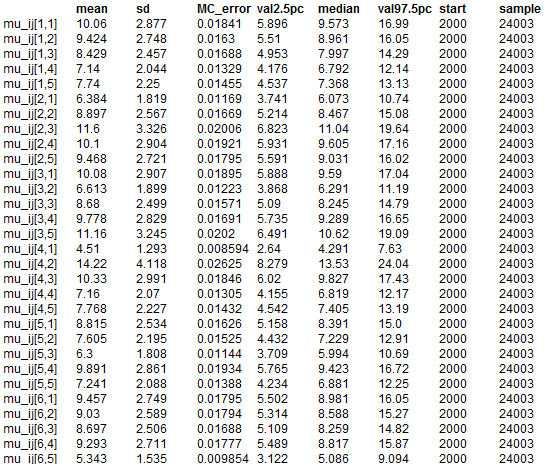
in Exponential Bayesian Hierarchical Model



**Figure 13.** Output of History Diagnostic in OpenBUGS for

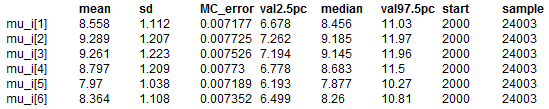
in Exponential Bayesian Hierarchical Model

The posterior varies a lot with the minimum of 5.343 and the maximum of 14.22. Most of the standard deviations are around 2 but with few exceptions that have standard deviations of one and four (Table 3, 4). The MCMC errors are very small and also the posterior 95% credible intervals are given in Table 3 and Table 4. Based on the simulation result, although there is not any significant difference between the average Points Per Game of players on division level, we do see noticeable differences between the average Points Per Game of players in different teams.



**Table 4.** Summary Results of Exponential Model Estimates

of Points Per Game of players on Team Level



**Table 5.** Summary Results of Exponential Model Estimates

of Points Per Game of players on Division Level

**V. Model Comparison**

Based on Table 6 and Table 7, the deviance information criterion (DIC) value of Normal Bayesian Hierarchical Model is 2498 while DIC value of Exponential Bayesian Hierarchical Model is 2500. We should choose the model with minimal DIC. That indicates that these two Bayesian models are pretty similar and Normal Bayesian Hierarchical Model is slightly better compared with Exponential Bayesian Hierarchical Model. Also, from Table 3, mean and variance estimators of are not stable, so we conclude that Normal Bayesian Hierarchical Model is a better Bayesian model choice for analyzing NBA Player Statistics data.



**Table 6.** DIC result of Normal Bayesian Hierarchical Model

**Table 7.** DIC result of Exponential Bayesian Hierarchical Model

**VI. Further Exploration on Frequentist Inference**

For normal Bayesian hierarchical model, we perform frequentist method on it. The maximum likelihood estimators (MLE) and () confidence intervals are listed below:

≈ Xi

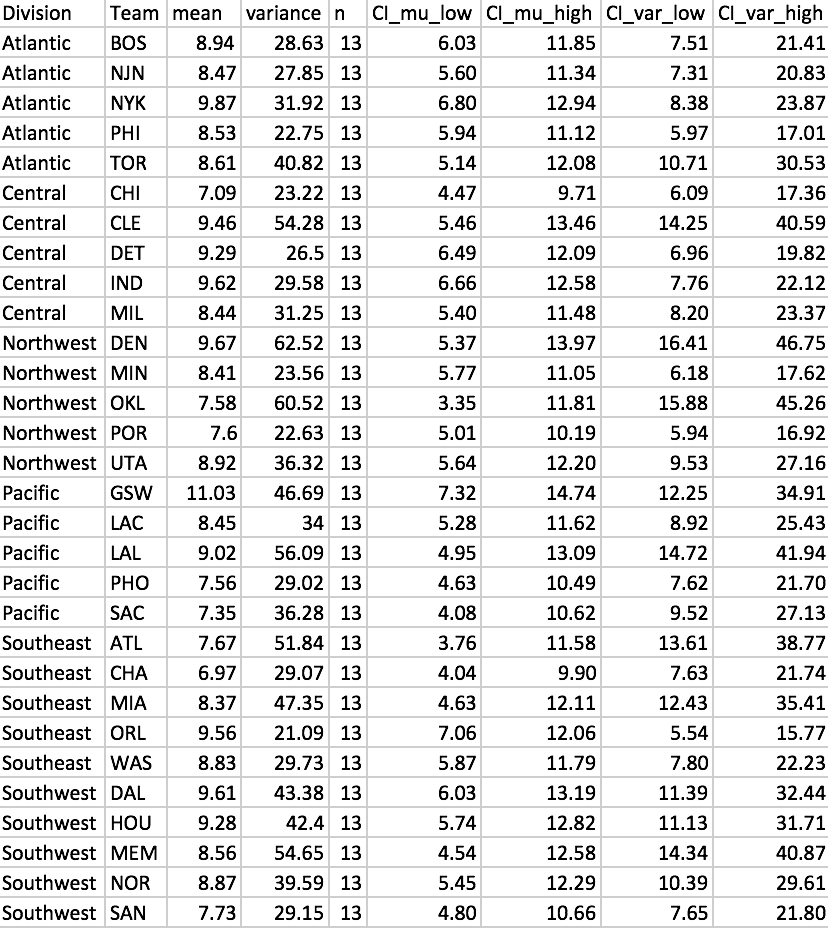
≈ Xij

σ2i ≈ S2i

σ2ij ≈ S2ij

Confidence interval of : (, +)

Confidence interval of σ2ij: , )



**Table 8.** Frequentist approach results of μij in each division and team

Compared with the results of normal Bayesian hierarchical model in Table 1, frequentist method presents much greater estimators than these of Bayesian method. And all the confidence intervals of frequentist method are wider than these of normal Bayesian hierarchical model. Large variance shows that the scoring ability of players in each team are varying greatly.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Methods | Atlantic | Central | Northwest | Pacific | Southeast | Southwest |
| Frequentist | 8.88 | 8.78 | 8.43 | 8.68 | 8.28 | 8.81 |
| Bayesian | 8.635 | 8.716 | 8.708 | 8.656 | 8.58 | 8.619 |

**Table 9.** Comparison between Frequentist and Bayesian approaches for μi

Both the frequentist and Bayesian methods results for μi are approximately 8.5 and the pairwise differences are very small. That indicates that results of Bayesian model seem good and the differences of Players’ Points Per Game (PPG) between different divisions are small. Players in each division have close average levels.

|  |  |  |  |
| --- | --- | --- | --- |
| Methods | σA2(Between Division  Standard Deviation) | σB2 (Between Team Standard Deviation) | σC2(Between Players Standard Deviation) |
| Frequentist | 0.2472 | 0.9319 | 37.09 |
| Bayesian | 0.1318 | 0.7632 | 34.7 |

**Table 10.** Comparison between Frequentist and Bayesian approaches for σ2

Based on Table 5, we can clearly see that normal Bayesian hierarchical model has slightly greater standard deviation between different teams and inside each team than frequentist model, but the differences are very small. But for between different divisions, result of frequentist method is almost twice as much as that of Bayesian method. That indicates that normal Bayesian hierarchical model shows representative results compared to frequentist model. Additionally, between different divisions and between teams, the results in the set are close to the mean and each other. But in each team, Players’ Points Per Game (PPG) in each set are far from the mean and each other. So the differences of players in each team are obvious.

**Appendix**

# Normal Model

model

{

for(i in 1:6){

for(j in 1:5){

for(k in 1:13){

y[k,j,i] ~ dnorm(mu\_ij[j],tausq\_c)

}

mu\_ij[j] ~ dnorm(mu\_i[i],tausq\_b)

}

mu\_i[i] ~ dnorm(mu0,tausq\_a)

}

mu0 ~ dflat()

tausq\_a ~ dgamma(0.001,0.001)

tausq\_b ~ dgamma(0.001,0.001)

tausq\_c ~ dgamma(0.001,0.001)

sigmasq\_a <- 1/tausq\_a

sigmasq\_b <- 1/tausq\_b

sigmasq\_c <- 1/tausq\_c

}

list(mu0=10,tausq\_a=1,tausq\_b=1,tausq\_c=1)

list(mu0=9,tausq\_a=1,tausq\_b=2,tausq\_c=3)

list(mu0=9,tausq\_a=3,tausq\_b=2,tausq\_c=1)

# Exponential Model

model

{

for(i in 1:6){

for(j in 1:5){

for(k in 1:13){

    y[k,j,i] ~ dexp(lambda[i,j])

}

lambda[i,j] ~ dgamma(psi,1)

mu\_ij[i,j] <- 1/lambda[i,j]

}

mu\_i[i] <- mean(mu\_ij[i,])

}

psi~dunif(1,5)

}

# Initialization for Gamma model

list(psi=1)

list(psi=3)

list(psi=5)

data

list(y=structure(

.Data = c(16.300000,6.111111,5.647059,6.296296,4.391304,14.347826,7.000000,2.444444,10.141026,18.253521,10.125000,13.703704,1.500000,2.400000,4.015873,6.867925,9.805970,16.875000,4.461538,7.777778,7.057971,11.980769,12.464789,18.804878,2.240000,5.304348,11.714286,4.720000,15.276923,3.714286,8.589286,7.373134,15.074074,2.156250,17.722222,6.955882,20.246914,8.166667,6.575758,13.105263,4.691176,8.134146,1.000000,8.739726,8.041096,17.085366,13.750000,5.684211,4.683333,3.375000,8.580645,14.031250,4.954545,17.200000,7.106061,23.971429,10.279412,8.597403,3.392857,11.353659,6.231707,1.950000,3.880952,1.727273,11.283784,0.500000,7.980000,17.5714286,8.963415,10.878378,1.000000,3.938462,4.3461538,8.768293,9.933333,10.734375,5.460317,2.055556,6.339286,2.000000,8.5061728,7.406250,29.710526,18.712121,4.852459,12.0000000,7.345679,3.950000,4.837209,8.565789,8.800000,4.025000,3.270833,9.9682540,5.057971,13.790323,18.065217,9.262500,6.7500000,13.489796,16.643836,3.045455,11.935897,5.521739,9.940299,10.297872,3.0625000,24.145161,8.482759,7.595745,11.679012,10.2105263,3.980769,4.261905,14.583333,7.321429,9.439024,6.464789,15.869565,0.9047619,11.000000,2.750000,10.370370,2.113636,0.8214286,15.463415,6.205479,11.944444,10.390244,15.370370,8.829268,2.058824,5.894737,28.159420,1.076923,19.547945,3.333333,4.290323,13.841463,8.338462,11.500000,3.388889,15.413333,13.000000,1.655172,6.644737,13.506173,10.947368,6.123288,3.179487,17.118421,14.033333,6.718750,3.704225,4.477273,8.219512,2.000000,4.000000,5.853333,30.146341,15.109756,9.907895,6.315068,8.407895,4.716049,1.076923,1.760000,5.963415,3.260870,17.858974,10.135135,8.513514,7.513514,3.857143,8.112903,5.986301,13.975610,2.600000,11.095238,2.666667,4.100000,2.333333,19.474359,2.612245,3.312500,2.642857,11.862069,7.192308,1.472222,9.378049,9.873016,11.634146,13.534247,4.250000,18.671053,13.888889,11.833333,5.030303,17.487500,25.484375,5.377778,4.500000,4.000000,19.814286,5.00000,13.028986,6.243902,11.636364,7.325000,4.777778,11.939024,2.581395,15.266667,4.388889,10.900000,16.854839,4.800000,18.52632,2.092593,9.088235,1.285714,10.974026,8.134146,26.986301,15.030769,7.207317,7.500000,18.307692,2.142857,2.387097,10.75610,2.682540,2.761194,2.379310,4.696203,9.500000,2.745098,1.000000,7.937500,8.219512,11.160494,1.250000,11.259259,0.00000,8.431373,16.456790,15.683544,2.151515,2.826923,10.285714,1.533333,20.138889,8.080000,8.513158,10.041667,16.787500,2.60000,3.324324,0.745098,8.466667,9.125000,0.6666667,18.037975,5.734177,14.172840,1.5714286,21.302632,2.214286,4.269231,3.031250,15.666667,3.211268,0.7692308,1.666667,6.3500000,3.263158,6.529412,11.292683,1.1851852,12.100000,4.289855,2.627907,9.133333,20.580247,7.913793,3.6326531,2.725000,6.0972222,14.820513,7.136986,4.955556,0.8333333,9.884615,4.055556,2.138889,13.571429,8.921053,26.558442,7.0833333,8.211538,7.7301587,8.839506,5.800000,16.586667,3.6172840,18.329268,4.193548,14.069444,12.584615,8.680000,9.609756,6.0121951,22.562500,14.1111111,6.567164,2.500000,10.057143,7.3636364,2.230769,12.742857,4.500000,6.944444,7.833333,6.542373,10.8518518,7.641026,7.089286,16.297297,1.8400000,5.963636,9.142857,10.300000,11.973333,3.413043,25.024691,2.125000,16.623377,7.500000,5.825397,14.888889,7.970149,19.5609756,8.878378,1.200000,1.500000,4.365854,5.191489,5.314286,9.117647,20.565217,16.207317,4.468750,8.810345,2.943662,11.9875000,14.608696,19.587500,1.000000,1.722222,2.320000,2.523810,17.463415,20.753086,3.058824,12.447368,3.250000,0.400000,0.7142857,10.365854,18.688889,7.108696,5.181818,7.240000,12.645161,14.479452,19.024691,3.808824,7.780488,4.392405,7.030769,17.8846154,16.493333,2.106383,2.500000,12.358974,1.666667,12.271605,3.884615,6.341772,5.792208),

.Dim=c(13,5,6)

)

)

1. Retrevied from: <https://www.rdocumentation.org/packages/SportsAnalytics/versions/0.2/topics/NBAPlayerStatistics0910> [↑](#footnote-ref-0)
2. Based on NBA official website: <http://www.nba.com/standings/2009/team_record_comparison/conferenceNew_Std_Div.html> [↑](#footnote-ref-1)