Analysis of Game Result Data Set

Xinan Wang

MSc. Statistical Science, HT 2021, Week 4

1 Introduction

The data set consists of 134 lists, each corresponding to a competition among some the 24 players and making up of two sequences of integers. The first sequence records the players who were involved in the specific game ordered by their ranks, and the other gives the seniority rank of the corresponding player at the time that game was played. My goal is to construct prior and observational models for the provided data, to determine the relative skills of the players, and to see whether the seniority rank shall be included in the observational model as a covariate. To better understand the data, I conducted exploratory data analysis of the competition results and the seniority ranks of the players. Statistical methods I used include Metropolis–Hastings MCMC algorithm for sampling from posterior distribution, bridge estimator for estimating the marginal likelihoods and Bayes factors, and Monte Carlo integration to estimate predictive probability for the replay of game 68. In the conclusion, I made some comments on the results and drawbacks of the methods I used, as well as some possible extensions to my analysis.

2 Data Exploration

This section contains a selection of the exploratory plots of the data set that I produced. Together with them I add some comments on the information they provide with me and how they led to assumptions of my subsequent analysis.

The histograms in Fig.1 show the distribution of the number of games each individual played, and the number of participants for the games, which are both heavily right skewed. The first graph indicate that the majority of players (19 out of 24) participated in less than 20 games, with only 1 player competed in more than 70 games. The second graph tells that most games were played by only 2 players, and the largest number of players in a game was 9. Moreover, the occurrence of games of more than 4 participants was very rare.

In the data, there is also an ordinal variable of the seniority ranks of the players at which the games were played. To see whether the results of the games depend on the seniority, several diagnostics are shown next page, including the scatter plots of seniority ranks against game result ranks, grouped barplot of counts of different

ranks, and line charts of ranks of several players. Since the number of participants were mostly small, I mainly used the relative seniority ranks to get more sense, i.e. the seniority rank among all the participants, instead of the overall rank. Note that in my analysis I indeed used the original rank for consistency with the data.

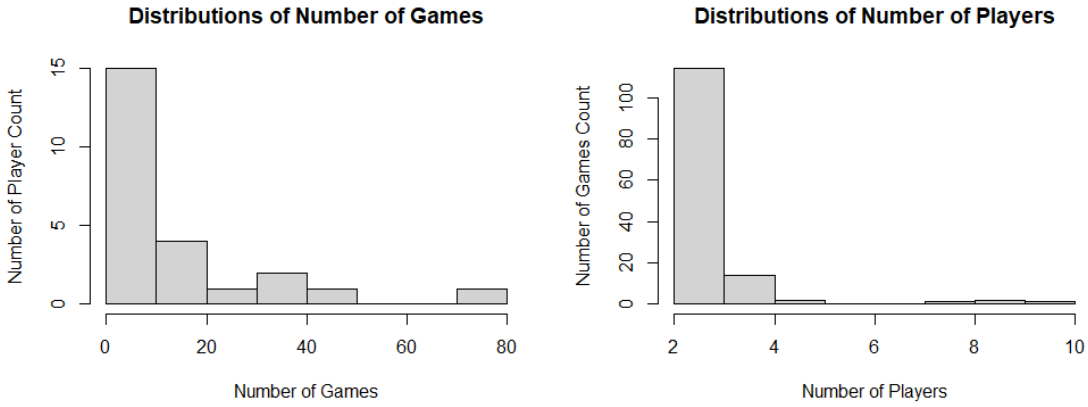


Fig. 1: Counts of
Left: Number of Games by Players
Right: Number of Players of Games

Fig.2 shown below summarise the distributions of competition results and seniority ranks. On the left is a scatter plot of the game result ranks against the relative seniority ranks of the players over the 134 games. On the right is a barplot of total number of players at different seniority ranks, grouped by their game results.

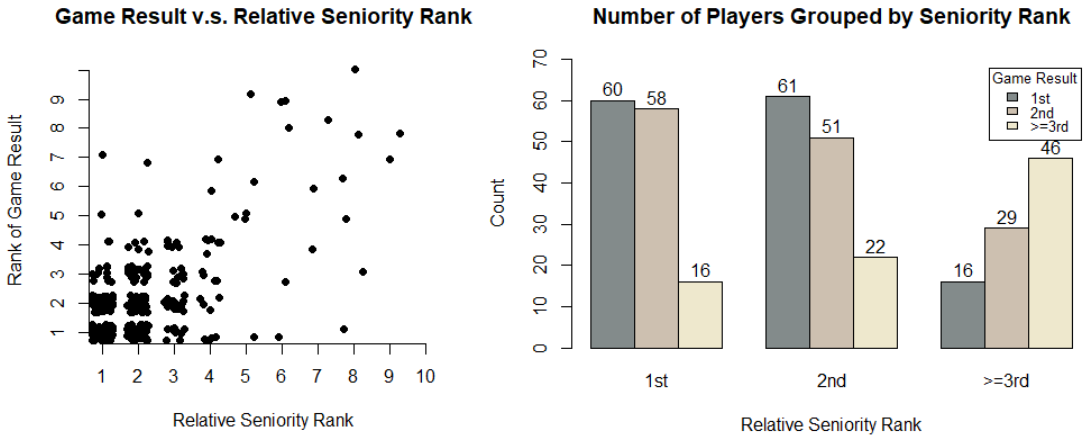
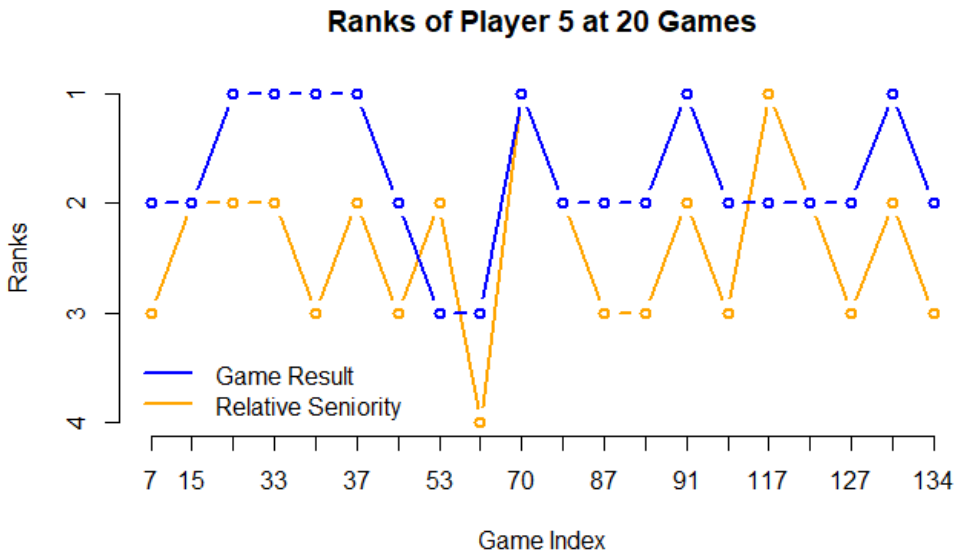
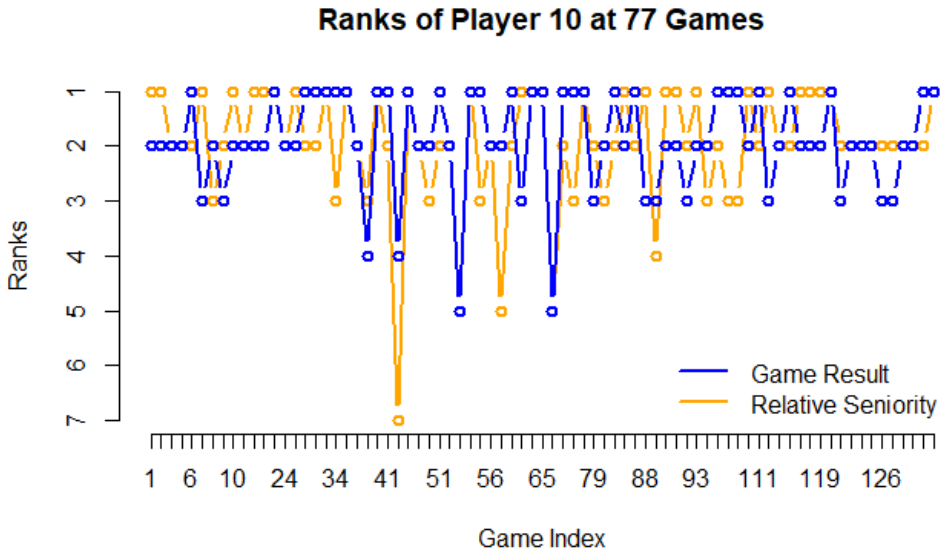


Fig. 2: Summary Plots of Game and Seniority Ranks

There are very few points lying far from the line $y = x$ in the scatter plot, and the barplot suggests that more experienced players tend to perform better. Therefore, there seems to be a relation between good performance or ranking with high experience level (seniority rank). To see this clearer, I also selected several players who played many games, and plotted their performance and seniority rank among all games they played shown in Fig.3.

The plot for player 15 (next page) to some extend reveals that the game and seniority ranks might be correlated, as they show approximately the same trend. On the other hand, the other two plots for players 5 & 10 provide little support for the relation between the, as the lines do not fluctuate in a similar manner.





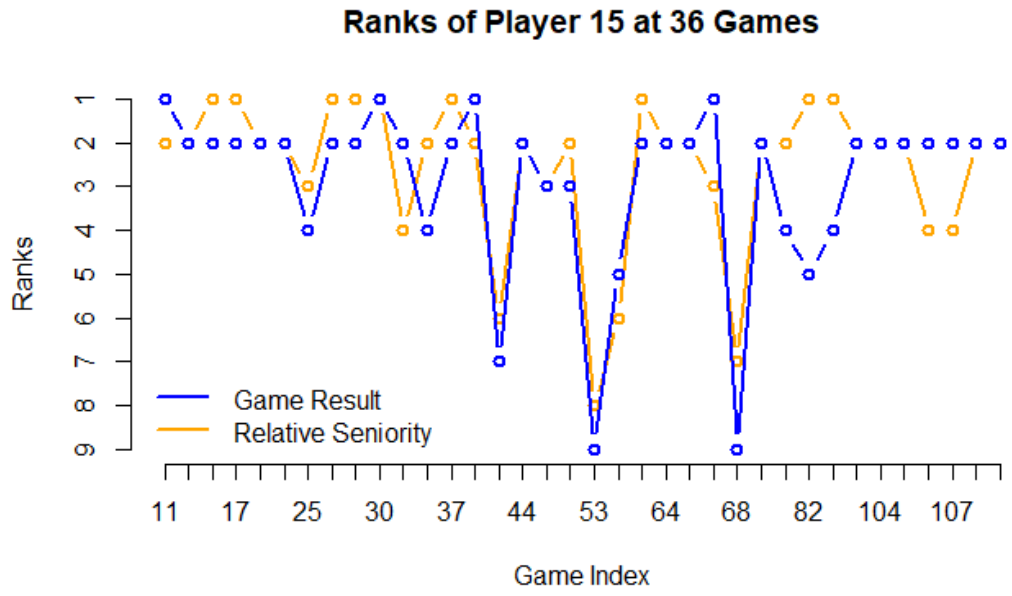


Fig. 3: Performance and Seniority Ranks of Players 5, 10 and 15

Therefore, qualitative EDA is insufficient and whether seniority should be included in the observational model needs to be determined via model selection.

3 Model Construction

3.1 Observational Model

Suppose each player $i \in \{1, 2, \cdots, 24\}$ has skill parameter λ_i. Due to the data structure, I decide to use Plackett-Luce ranking model which yields the form

$$Pr(O_l = o_l|\lambda) = \prod_{i=1}^{m_l} \frac{exp(\lambda_{o_{l_i}})}{\sum_{j=i}^{m_l} exp(\lambda_{o_{l_j}})}$$

for the Game l, $l \in \{1, 2, \cdots, 134\}$, where m_l is the number of players in Game l and $o_l = (o_{l_1}, \cdots, o_{lm_l})$ is the rank of the game. Further assume that the games are mutually independent, thus the joint probability can be written as

$$Pr(D|\lambda) = \prod_{l=1}^{134} Pr(O_l = o_l|\lambda)$$

An alternative observational model includes a term of seniority associated with each term in the exponent, with parameters $\beta = (\beta_1, \cdots, \beta_{24})$ and the seniority rank at Game l denoted by $e_l = (e_{l_1}, \cdots, e_{lm_l})$. That is,

$$Pr(O_l = o_l|\lambda, \beta) = \prod_{i=1}^{m_l} \frac{exp(\lambda_{o_{l_i}} + \beta_{e_{l_i}})}{\sum_{j=i}^{m_l} exp(\lambda_{o_{l_j}} + \beta_{e_{l_j}})} \text{ and}$$

$$Pr(D|\lambda,\beta) = \prod_{l=1}^{134} Pr(O_l = o_l|\lambda,\beta)$$

. With the above forms of observational models, I then move on to the step of choosing candidates of priors and performing model selection.

3.2 Choosing Prior Models

Since λ is involved in the model anyway, I would firstly choose priors for λ which represent the skills of the players, and then do selection between models with and without the seniority variable.

Before observing the data, I thought that the skills of players are independent, and most players do not differ much from each other, while the top players usually outweighs the others by a lot. In addition, the level of skills can be both positive or negative, since poor skill may lead to higher probability of failure. Hence, I choose independent normal and student-t distributions as my priors, differed by variance.

My life experience tells me that experienced players tend to perform better, so I would expect that players with higher seniority rank to have better performance and so they are probably going to be included in the model. The effect of experience should normally be positive, and restricted to a certain range. Hence, I want to choose independent Beta distribution as priors, The shape parameters depend on the ordinal seniority rank, and higher ranks should give higher expectations.

4 Model Selection

I used Metropolis Hasting MCMC algorithm to sample from posterior distributions given choices of prior and observational models, and bridge estimators to estimate the marginal likelihood to compare the models with different prior and covariates. Note that I shall just represent the parameters as λ, but the idea can be generalise to models with β's and other covariates.

4.1 Metropolis Hasting Algorithm

To simulate generations from posterior distributions, I used MH-MCMC with the posterior $\pi(\lambda|D)$ as the target, sub-sampled per 100 samples and burn-in period removed. The acceptance probability $\alpha(\lambda,\lambda')$ for the scheme can be simplified as $min\{1, \frac{p(D|\lambda')}{p(D|\lambda)}\}$ by choosing the proposal to be the prior $\pi(\lambda)$,

$$\alpha(\lambda,\lambda') = min\{1, \frac{\pi(\lambda'|D)q(\lambda|\lambda')}{\pi(\lambda|D)q(\lambda'|\lambda)}\} = min\{1, \frac{p(D|\lambda')\pi(\lambda')q(\lambda|\lambda')}{p(D|\lambda)\pi(\lambda)q(\lambda'|\lambda)}\} = min\{1, \frac{p(D|\lambda')}{p(D|\lambda)}\}$$

and then the process becomes more convenient as we only need to simulate from the priors and compute joint likelihood.

4.2 Bridge Estimator

With the samples from the posterior, we can then form robust estimators for the marginal likelihoods, namely the bridge estimators. It is known that for proper choice of function $h : \Lambda \to \mathbf{R}$, where Λ is the parameter space of λ, we have the identity

$$p(D) = \frac{E_\lambda(\pi(\lambda)p(D|\lambda)h(\lambda))}{E_{\lambda|D}(\pi(\lambda)h(\lambda))}$$

and by choosing $h(\lambda) = (\pi(\lambda))^{-1}p(D|\lambda)^{-\frac{1}{2}}$, the resulting estimator $\hat{p}(D)$ has the least relative mean square error, so the estimator is robust. The closed form for $\hat{p}$ is

$$\hat{p}(D) = \frac{\sum_{t=1}^{T} p(D|\lambda_{1,t})}{\sum_{t=1}^{T} p(D|\lambda_{2,t})}$$

where $\lambda_{1,t}$, $\lambda_{2,t}$ are samples from the prior and posterior distributions. We can simulate realisations from prior directly, and from posterior from MCMC.

5 Results and Model Selection

5.1 λ-only Model

I considered t_2, t_5 and $N(0, 1)$ distributions as priors with decreasing variance. The bridge estimators gave estimates for marginal probabilities of $1.01 \times 10^{-92}, 2.86 \times 10^{-84}$ and 1.35×10^{-82}, so the model with the normal prior is strongly favored with Bayes factors 47 and 1.34×10^{10} against the other two models. Below are some diagnostics of the sub-sampled results of the MCMC, treated as samples from posterior distribution $\pi(\lambda|D)$.

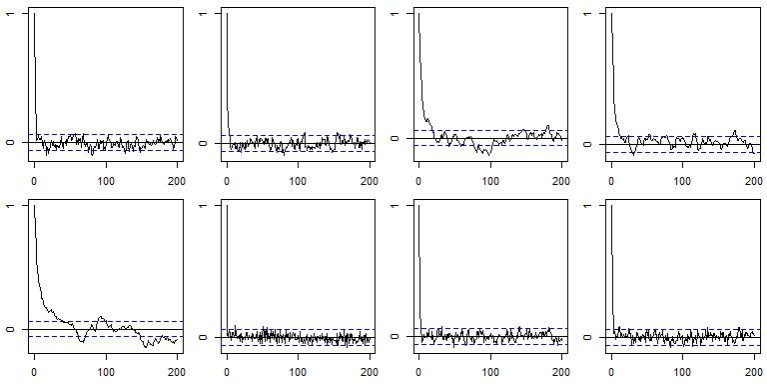


Fig. 4: Autocorrelation Plots of Some $\lambda_i|D$

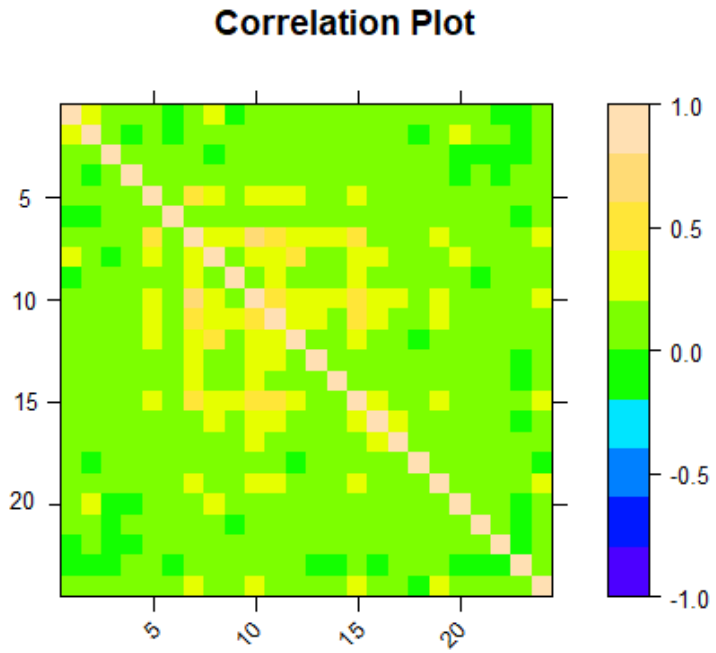


Fig. 5: Correlation of Realisations of λ's

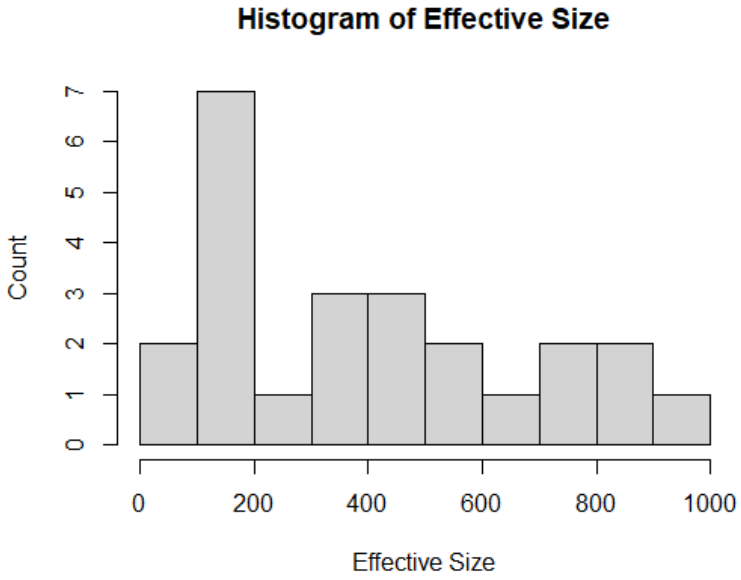


Fig. 6: Effective Sample Sizes of λ's

The ACF and correlation plots show satisfactory level of independence among simulations. In the histogram of effective sample sizes of λ, over two thirds of the λ_i's have effective sample sizes larger than 200. The rest components are questioned, as the correlation among the samples might be too large. This problem can be solved by increasing the sub-sampling step size, which I shall omit in my work.

To conclude, my choice of prior model for λ without seniority rank as a covariate is jointly independent standard normal, with estimated marginal likelihood for the data approximately 1.34×10^{-82}.

5.2 Models Involving Seniority Rank

Now fix the prior of λ to be i.i.d. standard normal, I want to test whether seniority should be included in my model as a covariate. I chose two types of Beta distributions with shape parameters related with ranks as prior for β_i, $i \in \{1, 2, \cdots, 24\}$.
The first type is

$$\beta_i \sim Beta(a(1 + \frac{1}{i}), a(2 - \frac{1}{i}))$$

and the second is

$$\beta_i \sim Beta(a(5 - \frac{i}{6}), a(1 + \frac{i}{6}))$$

The two families both has a rate parameter a to control the variance. They differ by how the ranks are involved, and as i increases, prior mean in the first type decreases as $\frac{1}{i}$, while in the second type as i, up to proportionality. This difference reflect how to the absolute seniority/experience level are related with the original categorical variable in different ways.

After fitting each of these prior models with different choices of a, the prior for β giving the largest marginal likelihood for the data was the second type with the rate parameter $a = 1$, giving marginal likelihood estimates $\hat{p}(D) = 5.94 \times 10^{-84}$. Surprisingly, this estimated likelihood is much smaller than the model with λ of standard normal prior only, which was my final choice of model.

5.3 Discussion

Before moving on to the next section, I want to briefly comment on why the model involving the seniority rank as covariate have failed. Admittedly, there are far more priors that might be tested, and it might be possible to find some distribution under which prior the estimated marginal likelihood could achieve a relatively higher value. They could have different scales, shapes or even from different families of distribution. More importantly, when doing EDA, it was found that most players' seniority rank do not differ too much from one game to another, so the effect of seniority rank largely are usually associated with specific players throughout, acting

as the players' characteristics as λ does. This problem is similar to having two correlated explanatory variables in a linear model, resulting in insignificance of both of them, which is not desirable especially for making predictions.

6 Estimating Predictive Distribution

In this section, I will demonstrate how I estimated the probability that Player 7 wins a replay of Game 68.

In short, the problem can be rewritten as finding

$$Pr(O'_{68_1} = 7|D) = \sum_{o'_{68} \in P_{68}} Pr(O'_{68} = o'_{68}|D)$$

where P_{68} denote all the permutations p of the player indices of Game 68, with $p_1 = 7$. Furthermore, the probabilities in the sum can be denoted in integral form

$$Pr(O'_{68} = o'_{68}|D) = \int_{\Lambda} Pr(O'_{68} = o'_{68}|\lambda)\pi(\lambda|D)d\lambda$$

The above integral can be estimated by

$$\hat{P}(o'_{68}) = \frac{1}{N}\sum_{i=1}^{N} Pr(O'_{68} = o'_{68}|\hat{\lambda}))$$

where $\hat{\lambda}$ are realisations from the posterior distribution obtained from the MH MCMC scheme in section 4.1, and N is the number of posterior realisations that we want to use. Hence, an estimate of the predictive probability is given by

$$Pr(O'_{68_1} = 7|D) = \frac{1}{N}\sum_{o'_{68} \in P_{68}} \sum_{i=1}^{N} Pr(O'_{68} = o'_{68}|\hat{\lambda}) = \cdots \approx 0.1783$$

7 Conclusion

In this report, I explored the Game Result data set, conducted some exploratory analysis in Section 2 to gain an overall understanding of the data, and questioned whether seniority should be included when modelling the data. In Section 3, I constructed the observational model and discussed the assumptions I want to made on the prior. I used MH MCMC algorithm to sample from the posterior and bridge estimator to estimate marginal likelihood in Section 4, which are then used to do model comparison and selection in Section 5. I showed some results of the model I want to choose, which does not contain a seniority rank component. Although this violates the common knowledge, I discussed the reasons behind this conclusion in Section 5.3 and pointed out that it is due to the correlation between game and seniority ranks, as well as the limited number of candidates of priors used to fit the model. I also checked that the dependence in the simulations is mild and acceptable, and compared the estimated likelihood of the data under the favored model with the other candidates. Lastly, I calculated the predictive probability of Player 7 winning the replay of Game 68 with the same competitors, which is approximately 0.1783.

There are several points I want to make regarding the strengths and weaknesses of the statistical methods I used. Firstly the MH MCMC algorithm is capable of sampling the complicated posterior distribution, but to reduce the effect of correlation between subsequent realisations, I sub-sampled the results with step size 100, which caused a waste of computational power and efficiency. Nonetheless, the samples are representative of the posterior which makes the effort valuable. Secondly, when estimating the marginal likelihood, I implemented the bridge estimator which yields a robust estimate than the traditional methods such as Monte Carlo estimator or Harmonic mean estimator.

I also have some final remarks that might be considered for future work. On one hand, the seniority rank as an ordinal categorical variable is not very informative, as it does not reflect the absolute experience level of each player, which is not desirable for parametric modeling. More data on the experience level can be collected in the future to improve the performance of models. On the other hand, more priors on both λ and β may be tried with if I had time, since the joint density has a complicated form. However, based on the available time and data, the prior model with independent standard normal λ is favored by my analysis.

R Code

```
library(MCMCpack)
library(lattice)

obs.dat=dget("CompRank21.txt")
n <- length(obs.dat) #134
num_player <- 24

# plist contains the data for each player
plist <- vector(mode = "list", length = num_player)
for (i in 1:num_player){
  plist[[i]] <-  list('G'=c(),'S'=c(), 'RS'=c(), 'R'=c())
}
par(mfrow=c(1,1))
num_per_game <- c()
for (i in 1:134){
  game_i <- obs.dat[[i]]
  num_player_i <- length(game_i[[1]])
  for (j in 1:num_player_i){
    # the jth player in game i
    player_j <- game_i[[1]][j]
    # Record game index, seniority, relative seniority and result rank
    plist[[player_j]][['G']] <- c(plist[[player_j]][['G']],i)
    plist[[player_j]][['S']] <- c(plist[[player_j]][['S']],game_i[[2]][j])
    plist[[player_j]][['RS']] <- c(plist[[player_j]][['RS']],
                                   rank(game_i[[2]],ties.method = "min")[j])
    plist[[player_j]][['R']] <- c(plist[[player_j]][['R']],j)
  }
  num_per_game <- c(num_per_game,num_player_i)
}

# Obtain number of games each player participated
num_per_player <- c()
for (i in 1:num_player){
  num_per_player[i] = length(plist[[i]][[1]])
}

# Diagnostics
barplot(num_per_player,names.arg = 1:24, xlab = 'Player',
        main = 'Number of Games Played by Each Competitor')
hist(num_per_game, main = 'Distributions of Number of Players',
     xlab='Number of Players', ylab='Number of Games Count')
hist(num_per_player,breaks=seq(0,80,10),
main = 'Distributions of Number of Games',
     xlab='Number of Games', ylab='Number of Player Count')


## Line chart of games player by player i
plot_player <- function(i){
  num_game <- length(plist[[i]][[1]])
  rank_range <- range(c(plist[[i]][['R']],plist[[i]][['RS']]))
  plot(plist[[i]][['R']], ylim = rev(c(rank_range[1], rank_range[2])),
       col='orange',lwd=2, type="b" , xlab ='Game Index',
       ylab="Ranks", axes=FALSE,
```

```
       main=paste('Ranks of Player',i, 'at', num_game, 'Games'))
  lines(plist[[i]][['RS']], col='blue',lwd=2, type="b")
  axis(1, at=1:num_game, labels=plist[[i]][['G']])
  axis(2, at=rank_range[1]:rank_range[2])
  legend('bottomleft', c("Game Result", "Relative Seniority"), box.lwd = 0.2,
         col = c("blue", "orange"),bty = 'n', lwd=c(2,2))

}

plot_player(5)
plot_player(7)
plot_player(10)
plot_player(15)


## Scatter plot
a = 0.3

plot(jitter(plist[[1]][['RS']], amount=a),jitter(plist[[1]][['R']], amount=a),
     pch=19, ylab = 'Rank of Game Result', xlab = 'Relative Seniority Rank',
     ylim = c(1,10), xlim = c(1,10), axes = FALSE,
     main = 'Game Result v.s. Relative Seniority Rank')
for (k in 2:24){
  points(jitter(plist[[k]][['RS']],amount=a),
         jitter(plist[[k]][['R']],amount=a), pch=19)
}
axis(1,at=1:10)
axis(2,at=1:10)

plot(jitter(plist[[1]][['S']], amount=a),jitter(plist[[1]][['R']], amount=a),
     pch=19, ylab = 'Rank of Game Result', xlab = 'Absolute Seniority Rank',
     xlim = c(1,16), ylim = c(1,10), axes = FALSE,
     main = 'Game Result v.s. Absolute Seniority Rank')
for (k in 2:24){
  points(jitter(plist[[k]][['S']],amount=a),
         jitter(plist[[k]][['R']],amount=a), pch=19)
}
axis(1,at=1:16)
axis(2,at=1:10)

## Grouped barplot
rank_table <- matrix(rep(0,9),nrow=3,ncol=3)
for (i in 1:length(plist)){
  playeri <- plist[[i]]
  for (j in 1:2){
    for (k in 1:2){
      rank_table[j,k] = rank_table[j,k] +
        sum((playeri$RS==j)&(playeri$R==k))
    }
  }

  for (l in 1:2){
    rank_table[l,3] = rank_table[l,3] +
      sum((playeri$RS==l)&(playeri$R>=3))
    rank_table[3,l] = rank_table[3,l] +
      sum((playeri$RS>=3)&(playeri$R==l))
```

```
  }
  rank_table[3,3] = rank_table[3,3] +
    sum((playeri$RS>=3)&(playeri$R>=3))
}
rank_table

x<-barplot(rank_table, beside=T, ylim = c(0,70), xlab="Relative Seniority Rank",
        ylab = 'Count', names.arg = c('1st','2nd','>=3rd'),
        legend=c('1st','2nd','>=3rd'),
        col = c('azure4','antiquewhite3','cornsilk2'),
        args.legend = list(title="Game Result", cex=0.8),
        main = 'Number of Players Grouped by Seniority Rank')
text(x,rank_table+3,labels=as.character(rank_table))


## Observational model
obs_prob <- function(d, lambda, beta = rep(0,24)) {
 out <- 1
 for (i in 1:length(d)) {
   players <- d[[i]]$o
   srank <- d[[i]]$e
   mi <- length(players)
   for (j in 1:mi){
     # Multiply by likelihood
     out <- out * exp(lambda[players[j]]+beta[srank[j]])/
       sum(exp(lambda[players[j:mi]]+beta[srank[j:mi]]))
   }
 }
 return(out)
}

## Construct bridge estimator
b_est <- function(d, l1, l2, b1 = matrix(rep(0,24*1000),ncol=24),
                  b2 = matrix(rep(0,24*1000),ncol=24)){
  n1 = 0; n2 = 0
  for (i in 1:1000){
    n1 = n1 + sqrt(obs_prob(d, l1[i,], b1[i,])) # numerator
    n2 = n2 + (sqrt(obs_prob(d, l2[i,], b2[i,])))^(-1) # Denominator
  }
  out <- n1/n2
  return(out)
}


#initialize state for MCMC with lambda and beta
lambda <- rnorm(24)
beta <- rbeta(24, shape1 = 1 +1/(1:24), shape2 = 2 - 1/(1:24))
oldp=obs_prob(obs.dat, lambda, beta)

K=1e5; SS=100;
out.mcmc=matrix(NA,K/SS,24*2)
for (k in 1:K) {
  # Choose state to update
  state = sample(1:48,1)
  if (state<=24){
    lambdap = lambda
```

```
    lambdap[state] = rnorm(1)
    betap = beta
  }
  else {
    betap = beta
    betap[state-24] = rbeta(1, shape1 = 1 +1/(1:24), shape2 = 2 - 1/(1:24))
    lambdap = lambda
  }

  newp=obs_prob(obs.dat, lambdap, betap)

  # Update probability
  alpha = newp/oldp
  if (runif(1)<alpha) {
    lambda=lambdap; beta=betap;
    oldp=newp
  }
  # Subsampling
  if (k%%SS==0) {
    out.mcmc[k/SS,]=c(lambda,beta)
  }
}

# sample prior, obtain posterior
lambda1 = matrix(rnorm(24*1000),ncol=24)
lambda2 = out.mcmc[,1:24]
beta1 = matrix(nrow=1000,ncol=24)
for (j in 1:1000){
  beta1[j,] = rbeta(24, shape1 = 1 +1/(1:24), shape2 = 2 - 1/(1:24))
}
beta2 = out.mcmc[,25:48]

#Compute bridge estimates
p1 <- b_est(obs.dat, lambda1, lambda2, beta1, beta2)
p1
# lambda  normal
# beta  Beta(5-rank/6, 1+rank/6)  Beta((5-rank/6)/2, (1+rank/6)/2)
#         1.672759e-84              8.034321e-87
#        Beta(1+1/rank, 2-1/rank)   Beta(2+2/rank, 4-2/rank)
#        5.941653e-84               8.116373e-89


out.mcmc <- as.mcmc(out.mcmc)
par(mfrow=c(1,1))
#correlation can be a problem, only mild correlation here
levelplot(out.mcmc,scales = list(x = list(rot = 45)), main='Correlation Plot')
effectiveSize(out.mcmc)  #these could be bigger closer to 1000
hist(effectiveSize(out.mcmc), breaks=seq(0,1000,100),
     xlab='Effective Size',ylab='Count',main='Histogram of Effective Size')
par(mfrow=c(2,4),oma=c(1,1,1,1)); #acf plots
for (i in 1:dim(out.mcmc)[2]) {
  par(mai=0.2*c(1,1,1,1));
  plot(acf(out.mcmc[,i],lag.max=200,plot=F),
       type='l',ann=F,xaxp=c(0,200,2),yaxp=c(0,1,1));
  text(50,0.8,colnames(out.mcmc)[i])
}
```

```
#initialize state for MCMC with lambda only
lambda0 <- rnorm(24)
oldp0=obs_prob(obs.dat, lambda0)

K=1e5; SS=100;
out.mcmc0=matrix(NA,K/SS,24)
for (k in 1:K) {
  state = sample(1:24,1)
  lambdap0 = lambda0
  lambdap0[state] = rnorm(1)

  newp0=obs_prob(obs.dat, lambdap0)

  alpha = newp0/oldp0
  if (runif(1)<alpha) {
    lambda0=lambdap0
    oldp0=newp0
  }
  if (k%%SS==0) {
    out.mcmc0[k/SS,]=lambda0
  }

  }
}


lambda01 = matrix(rnorm(24*1000),ncol=24)
lambda02 = out.mcmc0
p0 <- b_est(obs.dat, lambda01, lambda02)
p0
# lambda normal         t(2)                  t(5)
#        1.347753e-82   1.006573e-92          2.860277e-84

out.mcmc0 <- as.mcmc(out.mcmc0)
par(mfrow=c(1,1))
levelplot(out.mcmc0,scales = list(x = list(rot = 45)), main='Correlation Plot')
effectiveSize(out.mcmc0)
hist(effectiveSize(out.mcmc0), breaks=seq(0,1000,100),
     xlab='Effective Size',ylab='Count',main='Histogram of Effective Size')
par(mfrow=c(2,4),oma=c(1,1,1,1));   #acf plots
for (i in 1:dim(out.mcmc0)[2]) {
  par(mai=0.2*c(1,1,1,1));
  plot(acf(out.mcmc0[,i],lag.max=200,plot=F),
       type='l',ann=F,xaxp=c(0,200,2),yaxp=c(0,1,1));
  text(50,0.8,colnames(out.mcmc0)[i])
}

## Predictive Distribution

library(combinat)

game <- obs.dat[[68]]
players <- game[[1]] # get player list
n_player <- length(players)
players <- players[-2]
```

```
# remove player 7 first
comb <-  permn(players)
prob = 0
for (i in 1:length(comb)){
  # Add player 7 as first
  ranki = c(7, comb[[i]])
  for (j in 1:1000){
    out = 1e-3
    # Posterior sample of lambda
    lambdaj = lambda02[j,]
    for (k in 1:n_player){
      # Multiply by the kth component
      out <- out * exp(lambdaj[ranki[k]])/
        sum(exp(lambdaj[ranki[k:n_player]]))
    }
    prob = prob + out
  }
}

prob
# 0.1783173
```