Stochastic Methods for Fantasy Basketball

Xingchi Dai, Andy Shi, Hyungmok Son, Hidenori Tanaka
Applied Math 207
Harvard University

May 5, 2016

Abstract

Fantasy sports have become so popular in the United States, as many companies offer money to fantasy sports players for picking the best line-up of players to compete in the game. In this project, we used what we have learned from AM207 to find the best line-up constrained by factors such as salary, and number of players. Additionally, we constructed a model to investigate individual NBA players' contributions to their teams. Finally, we assess how players will perform on a game-to-game basis. Using techniques like simulated annealing, stochastic gradient descent, sampling, and time series methods, we can improve upon the greedy method for selecting players to the lineup, begin to understand how individual players contribute to their team's overall outcome, and predict how players will perform in their upcoming games given their past performance.

1 Problem Statement

Given the explosive growth of fantasy sports in the United States, our group was interested in seeing how techniques from AM 207 can be used to play fantasy sports. We focused on three areas:

1. Lineup selection;
2. Evaluating individual players' contributions;
3. Predicting players' condition over time;

Our goal in the lineup selection part of the model is to select an optimal lineup of players. However, we are constrained by a salary cap, and by the number of players we can choose.

We are also interested in how individual players contribute to their teams. To accomplish this, we constructed a model for player performance and use a variety of techniques, including L-BFGS optimization, stochastic gradient descent, and sampling methods, to estimate players' contributions.

Finally, we evaluate how players' condition and performance change over time. So far, we have discussed how to chose players to make the best team under constraints such as salary cap and used data from past year. The modeling is done under assumption that result of last year is consistent with result of next year. However, we haven't considered any information from time series analysis in the modeling. In this section, we try to capture trend in time series of player's score to predict player's condition in both short time scale (next game) and long time scale (next half year).

2 Data Description

We downloaded all of our data from `basketball-reference.com`. We primarily focused on the 2015-2016 NBA regular season for lineup selection, and downloaded salary information and player statistics (both on a per-game and per-100 possessions level) for the 2015-2016 season. Overall, for lineup selection and player contributions, we evaluated around 468 players, based on data from 1230 games over the course of an entire NBA season. We also downloaded data for individual players across their careers that was used to predict players' condition.

3 Methods

3.1 Lineup Optimization

For the lineup optimization, we used Simulated Annealing (SA). In order to evaluate this method, we first tried the naive method. The naive method ranks all basketball players by their points per game and selects players in descending order, from highest points per game, until the salary cap is reached or there are too many players.

For simulated annealing, which is similar to normal Metropolis-Hastings MCMC, however, we mimic the cooling and reheating system process. Suppose we have a current state with energy E_i and a new state with energy E_j, the new state is accepted with probability equal to $A = \exp(-\Delta E/kT)$. With this small change, the method will go to the optimum faster. However, it is still possible that our method will get stuck in the local optimum, in this case, we "re-heated" our system to try to escape from it.

3.2 Player Contribution Model

We build a probabilistic model to estimate each player's contribution to their team. The observed data in our model is the outcome of each NBA game played during the regular season. Additionally, for player j on team i, we will have two features $x_{i,j}$ and $y_{i,j}$, corresponding to some metric of that player's offensive and defensive skill set, respectively. For example, $x_{i,j}$ could be the points per 100 possessions that a player scores, and $y_{i,j}$ could be blocks per 100 possessions, steals per 100 possessions, or the defensive rating. For each matchup between team i and i', we will compute the *team offensive rating O* and *team defensive rating D* (the formula is given for team i but proceeds similarly for team i'):

$$\begin{aligned} O_i &= \beta_{i,0} + \beta_{i,1}x_{i,1} + \beta_{i,2}x_{i,2} + \ldots + \beta_{i,J}x_{i,J} \\ D_i &= \gamma_{i,0} + \gamma_{i,1}y_{i,1} + \gamma_{i,2}y_{i,2} + \ldots + \gamma_{i,J}y_{i,J} \end{aligned} \quad (1)$$

Here, the coefficient $\beta_{i,j}$ represents the offensive contribution of player j on team i, and $\gamma_{i,j}$ represents his defensive contribution.

The team offensive ratings and the team defensive ratings are combined to give a noisy estimate of the actual score differential observed in the game. The score differential $S_{i,j}$ between teams i and j will be modelled as

$$S_{i,j} \sim \mathcal{N}((O_i - D_j) - (O_j - D_i), \sigma^2). \quad (2)$$

As such, our score difference Gaussian is a mixture model consisting of many Gaussian models of different team pairs, The likelihood of this model can be written as:

$$\prod_{n=1}^{N} N(\mu_{nk}, \sigma)^{z_n k}$$

where $z_{n,k}$ is an indicator showing which team pair the nth data belongs to.

Looking at a histogram of the actual score differential across all games in the 2015-2016 NBA season, we see that a normal distribution is justified for $S_{i,j}$ (Figure 1).

Through our modeling approach, we hope to learn the values of $\gamma_{i,j}$ and $\gamma_{i,j}$, the offensive and defensive contributions for player j on team i.

We will have some kind of a prior over the β and γ. One simple idea is to model the β independently, with a Normal prior. Otherwise, we can impose a block-correlated multivariate normal model, where players on each team share some covariance. This would model the influence of a coach or team organization on player performance. Additionally, the model can be extended by considering different measures of $x_{i,j}$ and $y_{i,j}$, the player offensive or defensive ratings. We could model each of them as some combination of other factors, and the weightings for each of these factors could have priors as well.

After experimenting with different measures for each player's $x_{i,j}$ and $y_{i,j}$, we settled on using

$$
\begin{aligned}
y_{i,j} &= \text{blocks} + \text{steals} + \text{defensive rebounds} \\
x_{i,j} &= \text{points per game} + \text{assists per game}
\end{aligned} \tag{3}
$$

Histograms of $x_{i,j}$ and $y_{i,j}$ are shown in Figures 2–3.

We also used independent normal priors for the β and γ, and experimented with a multivariate normal prior with non-identity covariance matrix.

3.3 Model Estimation

3.3.1 L-BFGS

We first used L-BFGS [1] to estimate our coefficients. We applied L-BFGS to maximize the likelihood and obtain MLE of the β and γ coefficients. L-BFGS requires a gradient: to compute this, we used Autograd, a Python automatic differentiation library (`https://github.com/HIPS/autograd`). We used Scipy's built-in implementation of L-BFGS. The likelihood we optimized was ℓ_2 penalized: this regularized our coefficient estimates. Otherwise, optimizing an unpenalized likelihood led to high variance in the coefficient estimates, because we had around 1000 parameters and only 1230 games to learn them.

3.3.2 Stochastic Gradient Descent

We also used stochastic gradient descent to obtain MLEs for the β and γ coefficients. At each iteration, we evaluate the gradient of the likelihood of one datum (one matchup between two teams), and step in the direction of that gradient, scaled by some step size. Stochastic gradient descent is used to train models with lots of data. In these models, it's expensive to evaluate the gradient for all the parameters, so we approximate the gradient by evaluating it at one data point. Unfortunately, the stochastic gradient descent algorithm did not find the optimum and was sensitive to the initialization of the parameters. We hypothesize this is because we did not have enough data to be able to use a noisy approximation of the gradient with one datum at a time.

3.3.3 Metropolis Hastings

We used Metropolis Hastings to estimate our model. Each β and γ is drawn from a standard normal distribution. We used PyMC2 [2] to optimize our parameters. In this setting, we need to use over 2000 data to optimize nearly 500 parameters. This method is hard to get converged data, as the over-fitting situation might occur. A better choice would be to use clustered data to do MH algorithm.

3.3.4 Clustering and Metropolis Hastings

Intuitively, people who have relatively same abilities, either offensive or defensive, shall have similar contribution to the group. It is unlikely a very good player did a super bad job in one game if all other conditions keep the same. As such, we clusters all of our over four hundred players into ten groups by their offensive ratings and defensive ratings using K-Means. We then use their cluster mean value to approximate the group's. We use those optimal values and rank our players again in this section by the following metrics:

$$\text{Offensive Score} = \beta_j * \text{Offensive Rating}$$

$$\text{Defensive Score} = \gamma_j * \text{Defensive Rating}$$

Here, j is the cluster this player is in. By using this method, we could potentially avoid the over-fitting situation and get converged data.

3.3.5 Elliptical Slice Sampling (ESS)

In many probabilistic models, the dependency among latent variables is assumed to follow the multivariate Gaussian distribution. In the model where such strong dependency among the variables exists, like Gaussian priors, the performance of usual MCMC samplers such as Gibbs sampler might not be great. In this case, we can use ESS [3] instead, which is also known to be faster than Gibbs sampling and Metropolis-Hastings (MH) in these cases. As one of our goals in this project we try to obtain the information about our parameters, β's and γ's for each player and each team, by fitting our model into the observed data with the ESS method. In this method, we sample from a posterior distribution for the parameters that is proportional to the product of the multivariate Gaussian prior and a likelihood that is presumed to be a Gaussian based on the distribution of the differential scores calculated from the actual data.

3.3.6 Other Sampling Methods

We considered using other sampling methods, such as Hamiltonian Monte Carlo or No-UTurn Sampling (NUTS) [4]. We tried using NUTS, as implemented in PyStan [5], but the model was unable to run before the deadline.

3.4 Prediction of player's condition

3.4.1 Short term prediction: Predict points in the next game with Markov Model

Can there be any correlation between points in the last game and the next game? Assuming each player has condition which lasts for certain period, we can naively expect to capture each player's scoring pattern. Even if two players have the same average points per game, one player may score with small fluctuation, while another player do pretty well for three games and then do poorlly for three games after. To capture these trends, we employed Markov Model to construct transition matrix with training data and tested our prediction with testing data. For consistency we demonstrate the method using scoring data from legendary NBA player Kareem Abdul-Jabbar who played from 1969 to 1989. The raw data of points per game in Kareem's entire career is shown in Figure 12.

We define 4 states of player's condition of ith game(c_i) by analyzing training data which is time series of "Score(S) per game". From the training data we first calculate mean(μ) and standard deviation(σ). We then defined 4 conditions as "Best" ($c_i = 4$ when $\mu + \frac{\sigma}{2} < S_i$), "Better" ($c_i = 3$ when $\mu < S_i < \mu + \frac{\sigma}{2}$), "Worse" ($c_i = 2$ when $\mu - \frac{\sigma}{2} < S_i < \mu$), and "Worst" ($c_i = 1$ when $S_i < \mu - \frac{\sigma}{2}$).

Based on this classification, we construct transition matrix in both 1st and 2nd order using results from Kareem's first 900 games as training data. This transition matrix tells us how much score can we expect in the next game given results from past several games. This kind of prediction can be particularly useful when selecting players who should be on today's game.

3.4.2 Long term prediction: Predict trend of points in the next half year with ARMA model

Since it requires physical and mental strength to be competitive in NBA, there are many factors that can affect player's condition for long period. For example, aging or injury can negatively affect their performance in longer time scale. Also some younger players who is still developing their skills may continuously improve their performance. To characterize these longer time scale trends, we used a rolling mean and an Autoregressive-moving-average model (ARMA model) for prediction.

4 Results

4.1 Lineup Selection

The naive method gives us an expected result: the method chose the most famous players first. When the salary approaches to the cap, it skipped a lot of good players and chose many very cheap players to fill up the space. The total value we got is 176.4.

With SA method, we got a much better result, 238.6, with convergence in a few thousand iterations (Figure 4). By looking at the list given by this method, we see that it did choose many famous NBA players, but also it chose many new, underpaid players who actually have very good statistics (Table 1).

However, with different initial setting, we ended up different line-up. Simulated Annealing does have a high chance of getting stuck in one local optimum point. We used re-heating method, however, when the temperature cools down, it still got the chance to go back to the original point. The entire process is like I restart the problem, but with a closer location to the local optimum.

With multiple tries, we were able to find a relatively good result.

4.2 Player Importance Estimation

Using the coefficients obtained from L-BFGS optimization, we ranked the players in the NBA by their offensive contribution β (Table 2). Additionally, using the fitted β and γ coefficients, it's possible to rank players in each team by their contribution. We see that the top players on each team are the ones predicted by the β and γ coefficients—see example for the Cleveland Cavaliers in Table 3.

Furthermore, we validated the model used to estimate player importance. We used β and γ coefficients fitted on the first 80% of games from the 2015-2016 season to predict game outcomes and score differentials for the remaining 20% of the games in the 2015-2016 season. We achieved a mean squared error at predicting the score differential of 12.6, but in many cases our score difference prediction was very different from the actual score of the game (see Figure 5). However, we found that we were able to predict the winning team with around 70% accuracy using our model.

4.3 Sampling

4.4 Metropolis Hastings

The PyMC didn't converge for the non-clustering data, even after 1 million samples (Figure 6). We have too many parameters to optimize, but only limited data at hand. It was really hard to get converged values. We tried to use PyMC3, however, with very limited documentations, we weren't able to get it done on time. Too many parameter results in the over-fitting situation: a little bit noise will cause the unstable of the entire model. With a number of times failures, we decided to reduce our features before we run PyMC. After we clustered our data first, we ended up with a good result. Our result shows that most of β and γ parameters converged (Figure 7), judged by their Geweke Tests. We also ranked players used scores we got. The ranking gives a relatively reasonable result, as most of best NBA players are on the top of lists.

4.4.1 Elliptical Slice Sampling (ESS)

For the ESS, we constructed and used a covariance matrix in which the players in the same team have covariance among themselves. The covariance value was randomly sampled from a gamma distribution so that it was ensured to be non-negative. The reason we used such simple covariance matrix is that it was very difficult to deduce true correlation among players given data. We could apply our intuition in NBA - for instance, if the play style of a certain player is in harmony or conflict with those of other players, etc - to come up with a more realistic covariance matrix, but our such knowledge and relevant information was very limited. By using the covariance matrix, we tried ESS for 40,000 times of iterations, which took slightly longer than 1 hour. We obtained the states of each parameter—β's and γ's of each player and each team—at each round of iteration. The trace plot of one of the β's is shown Figure 9, and the result seems non-convergent. Figure 8 shows the histogram of the beta values which is plotted for the purpose of MAP analysis, but since the result is not convergent, the MAP estimation seems not useful in this case. Also, we ranked the players based on the offense beta values in Table 5, but the result was different from our expectation and also from the L-BFGS optimization outcome. Since the offense beta is the contribution factor of each player into the score, we expected that the players with high PTS (points per game) would contribute more into the team scores so that they would have high offense beta values. However, the result in Table 5 does not capture this intuition (as a reference, the highest PTS is 30.1) and we think this is because the ESS result is not convergent. Possible reasons for the non-convergence result will be discussed in the next section.

We also applied the ESS for each game separately where only two teams were involved (the number of iterations = 40,000). In this case, the β's and γ's nicely converge and such convergence is well captured (Figure 10 and 11). Moreover, using the mean of each parameter, we estimated the differential score and it agrees pretty well with the actual differential score. Having the data from 1,230 games, we applied ESS to each of them and as a result, the mean difference between the estimated and the actual differential scores is 0.3 % with 6.8 % of the standard deviation. Each ESS was iterated for 1,500 times for this result. A very small mean with a pretty large standard deviation means that the performance of ESS for some games are relatively poor, and it might be because complex correlations among players that might exist in those games are not reflected well enough with our covariance matrix.

4.5 Player Condition Estimation

4.5.1 Short Term Prediction

We tested prediction ability of the 1st and 2nd order Markov model with results from 900th to 1100th games of Kareem's career. We got accuracy of 48.2% for 1st order Markov Model and 51.0% for 2nd order Markov Model. This result shows high prediction ability of this modeling, since the accuracy should be around 25% if we randomly predict player's condition.

4.5.2 Long Term Prediction

First, the rolling mean clearly captures that Kareem's performance decreased every year in latter part of his carrier. Second, the ARMA model successfully captured fluctuation pattern of scores per game and gives reasonable prediction as shown in Figure 13.

5 Discussion

In the sampling part, one of the reasons that the ESS we applied for the entire games did not converge would be our covariance matrix. Since new proposals in the ESS are determined by the random draws from a multivariate Gaussian, the Markov chain might not be properly mixed depending on the covariance matrix of the multivariate Gaussian. For better sampling, a covariance matrix which reflects real correlation among players would be essential. Furthermore, we have data for 30 teams and 497 players and each team played 82 games. This means that we are trying to obtain the information about 1,000 parameters (i.e. betas) while having at most roughly 80 data points with which we can sample for each parameter, because each team (and each player) participates in 80 games. Every time we tweak a parameter, it changes the likelihood in 80 places: thus, games are very correlated, which can pose a problem for many types of samplers. It seems that the number of data we have is insufficient. The fact that the ESS applied for a single game works pretty well tells us we presumably have too many parameters to sample given the number of data for each when applying the ESS for the entire games. As applied in conjunction with the Pymc method, clustering players to effectively decrease the number of parameters is a nice way to solve this issue.

6 Conclusion

Using simulated annealing, we beat the naive method when it came to selecting the best lineup, with a restriction on salary and number of players. Even though many of our sampling methods did not converge, we used L-BFGS and sampling with a reduced parameter space to estimate player importances, given a player's aggregate performance over a season and score differentials from each game they played. Finally, using Markov models and ARMA models, we predicted player performance in future games given past performance.

Code and Data

Our code and data are available online at `https://github.com/AM207Fantasy/am207`.

References

[1] Ciyou Zhu, Richard H Byrd, Peihuang Lu, and Jorge Nocedal. “Algorithm 778: L-BFGS-B: Fortran subroutines for large-scale bound-constrained optimization”. In: *ACM Transactions on Mathematical Software (TOMS)* 23.4 (1997), pp. 550–560.

[2] Anand Patil, David Huard, and Christopher J Fonnesbeck. “PyMC: Bayesian stochastic modelling in Python”. In: *Journal of statistical software* 35.4 (2010), p. 1.

[3] Iain Murray, Ryan Prescott Adams, and David J.C. MacKay. “Elliptical slice sampling”. In: *JMLR: W&CP* 9 (2010), pp. 541–548.

[4] Matthew D Homan and Andrew Gelman. “The no-U-turn sampler: Adaptively setting path lengths in Hamiltonian Monte Carlo”. In: *The Journal of Machine Learning Research* 15.1 (2014), pp. 1593–1623.

[5] Bob Carpenter et al. “Stan: a probabilistic programming language”. In: *Journal of Statistical Software* (2015).

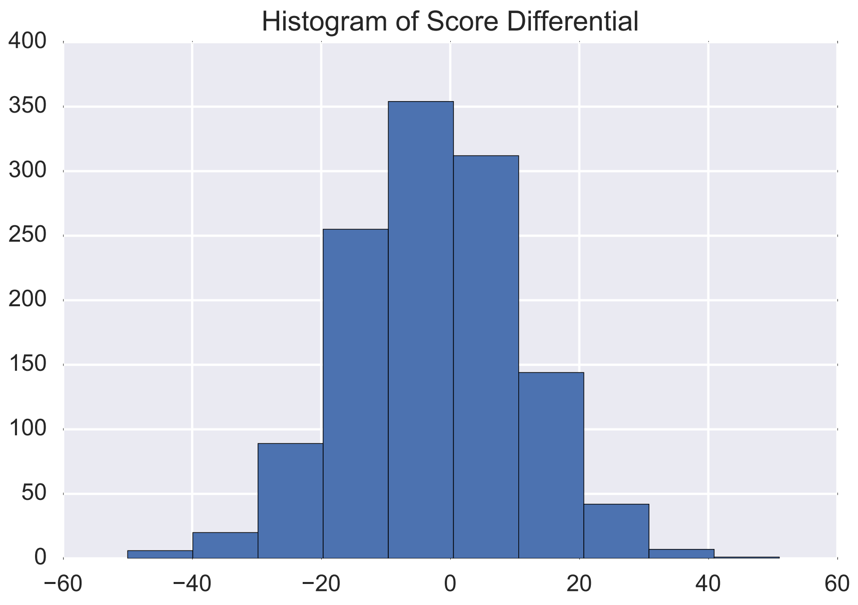

Figure 1: Distribution of Score Differences across the 2015-2016 NBA season.

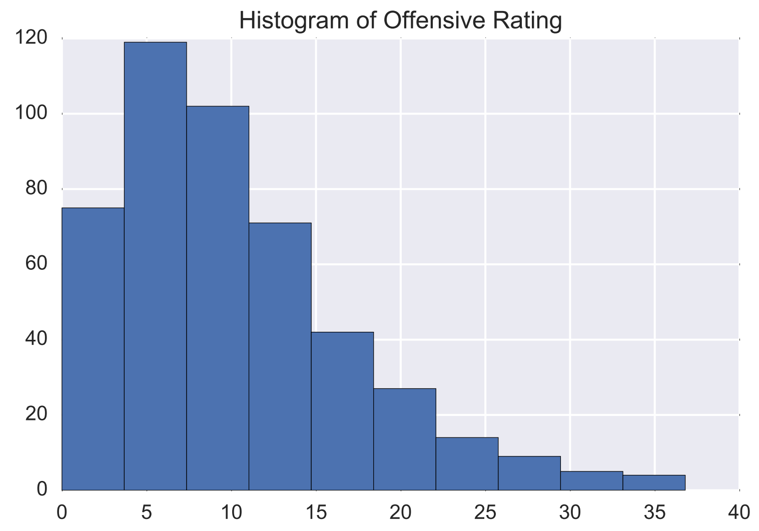

Figure 2: Histogram of offensive rating

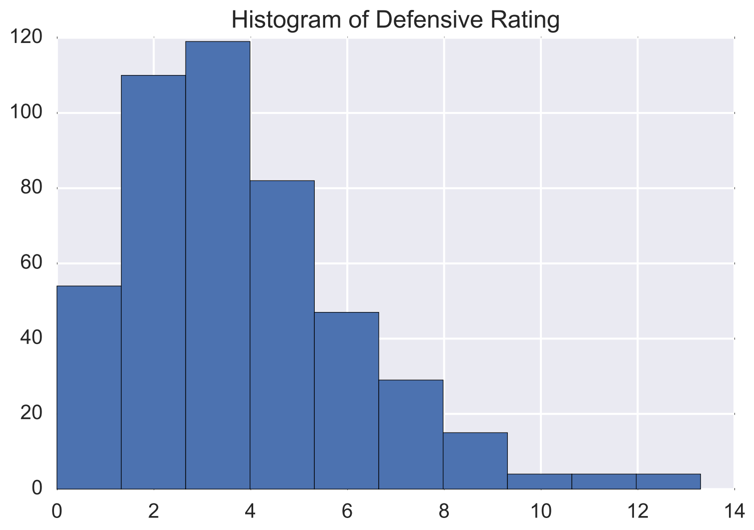

Figure 3: Histogram of defensive rating.

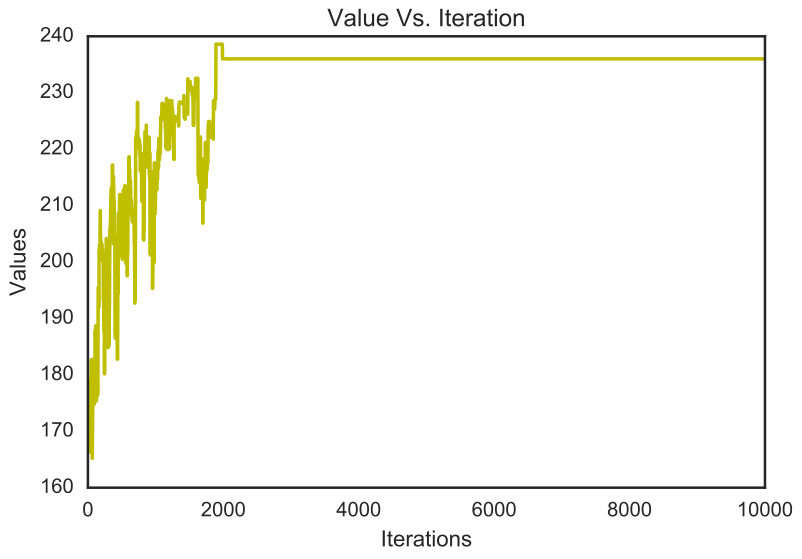

Figure 4: Plot of the value of our lineup (total points per game scored) vs. the iteration number for simulated annealing. We see that simulated annealing method converged quickly to an optimum.

Naive Method	**Simulated Annealing**
Stephen Curry	Stephen Curry
James Harden	Jordan Clarkson
Jordan Hamilton	C.J. McCollum
Michael Carter-Williams	Michael Beasley
Harrison Barnes	DeMarcus Cousins
C.J. Miles	Hassan Whiteside
Omri Casspi	Damian Lillard
Ish Smith	Dahntay Jones
J.R. Smith	Giannis Antetokounmpo
Joe Johnson	Evan Fournier
Isaiah Thomas	Isaiah Thomas
J.J. O'Brien	Anthony Davis

Table 1: Lineups generated with the naive method (greedy algorithm) and simulated annealing.

Player	**Team**	**Salary**	**PPG**	**beta**
Stephen Curry	GSW	11.4	30.1	0.112917
Kawhi Leonard	SAS	16.5	21.2	0.10513
LaMarcus Aldridge	SAS	19.5	18	0.086141
Tony Parker	SAS	13.4	11.9	0.075976
Klay Thompson	GSW	15.5	22.1	0.074252
Russell Westbrook	OKC	16.7	23.5	0.073085
Kevin Durant	OKC	20.2	28.2	0.071576
Draymond Green	GSW	14.3	14	0.065665
LeBron James	CLE	23.0	25.3	0.059836
Manu Ginobili	SAS	2.8	9.6	0.056099
Patrick Mills	SAS	3.6	8.5	0.049918
Tim Duncan	SAS	5.0	8.6	0.049911
Kyrie Irving	CLE	14.8	19.6	0.045292

Table 2: Player importances, as estimated using L-BFGS optimization.

Player	Assists	Points	beta
LeBron James	6.8	25.3	0.059836
Kyrie Irving	4.7	19.6	0.045292
Kevin Love	2.4	16	0.034291
J.R. Smith	1.7	12.4	0.026281
Matthew Dellavedova	4.4	7.5	0.022183
Mo Williams	2.4	8.2	0.01976
Tristan Thompson	0.8	7.8	0.016028
Iman Shumpert	1.7	5.8	0.013981
Channing Frye	1	6.1	0.013231
Timofey Mozgov	0.4	6.3	0.012486
Richard Jefferson	0.8	5.5	0.011741
Jordan McRae	1.1	4.5	0.010434
James Jones	0.3	3.7	0.00745
Sasha Kaun	0.1	0.9	0.001865

Table 3: Player importances for offense, sorted by β value, for the Cleveland Cavaliers. Superstars like LeBron James, Kyrie Irving, and Kevin Love are the most important to their teams.

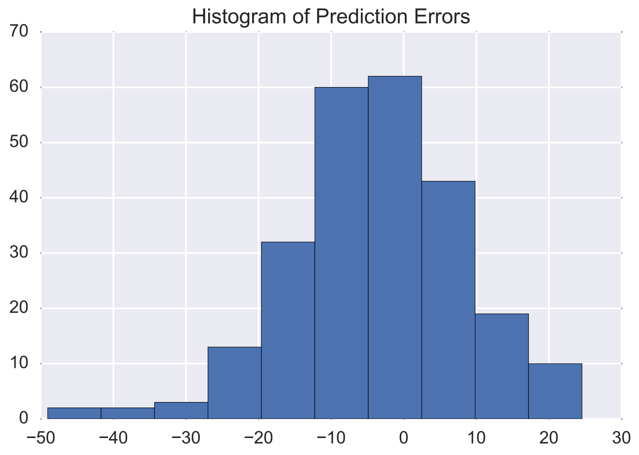

Figure 5: Histogram of prediction errors for the last 20% of the games in the 2015-2016 season

.

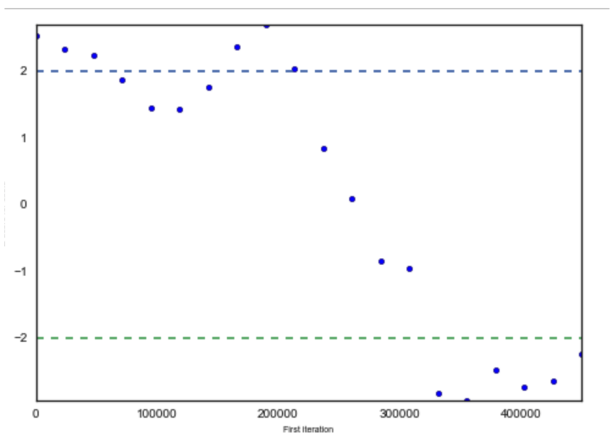

Figure 6: Geweke test plot for the 50th β for the first version of MH(No clustering). Most of parameters didn't converged

.

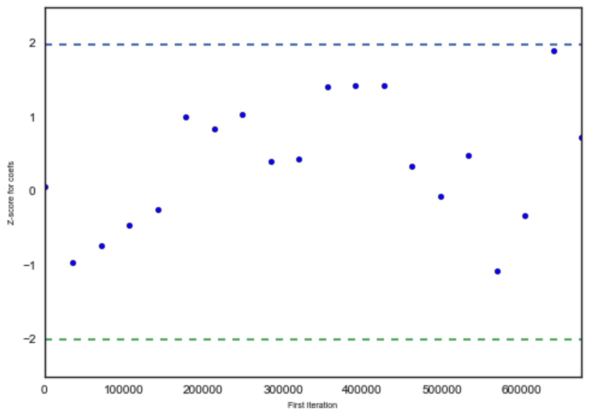

Figure 7: Geweke test plot for the 16th β for the version of MH(clustering). Most of parameters converged after we reduced the dimension

.

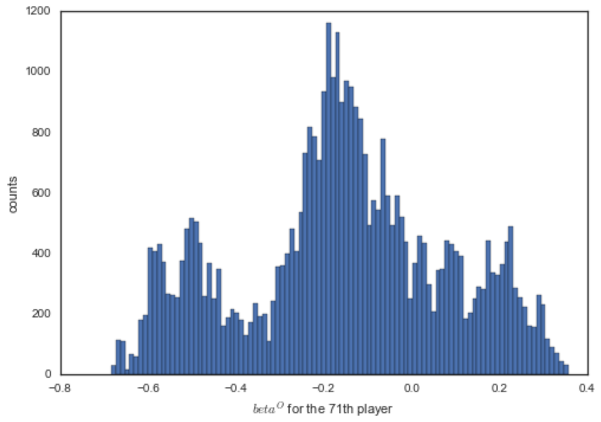

Figure 8: The histogram of the offensive rating, beta, for the 71-th player

Player	Offensive_score	Defensive_Score
Stephen Curry	111.892686	12.297513
James Harden	110.980517	13.545087
Russell Westbrook	103.075056	14.792661
Kevin Durant	100.946662	17.466034
LeBron James	97.602044	14.257987
John Wall	91.52092	12.653963
Paul George	82.70329	14.792661
Blake Griffin	79.966784	14.614436
Carmelo Anthony	79.054615	13.901537
Damian Lillard	69.031052	2.337387
Chris Paul	63.837493	2.983898
Isaiah Thomas	61.457112	1.790339
Kyle Lowry	59.725926	3.232556
DeMar DeRozan	59.509527	2.486581
Eric Bledsoe	57.345545	3.033629
Kemba Walker	56.479951	2.884434

Table 4: PyMC Results by clustered data

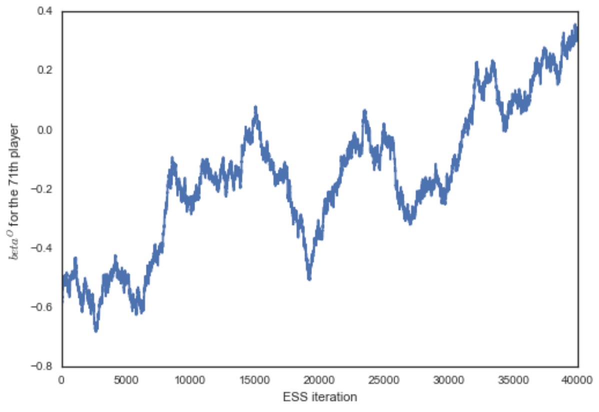

Figure 9: The trace plot of the offensive rating, beta, for the 71-th player

Player	Team	AST	PTS	beta
B. Jennings	ORL	3.5	6.9	2.312
Taj Gibson	CHI	1.5	8.6	2.244
Salah Mejri	DAL	0.3	3.7	2.049
Kyle Anderson	SA	1.6	4.5	2.026
C.J. Watson	ORL	2.7	4.3	2.014
B. Biyombo	TOR	0.4	5.5	1.963
Boris Diaw	SA	2.3	6.4	1.863
P.J. Tucker	PHX	2.2	8.0	1.862
Marcus Smart	BOS	3.0	9.1	1.813
Solomon Hill	IND	1.0	4.2	1.765

Table 5: Player ranking based on the offensive rating

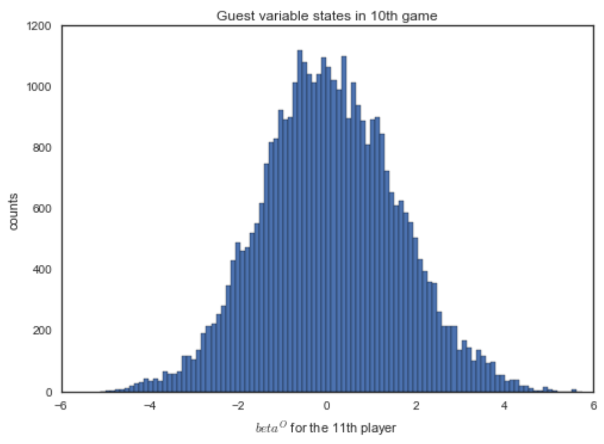

Figure 10: The histogram of the offensive rating, beta, for the 11-th player of the guest team in the 10th game. MAP = -0.6093.

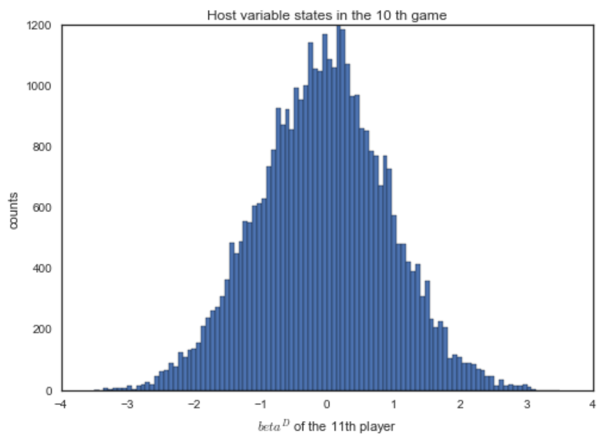

Figure 11: The histogram of the offensive rating, beta, for the 11-th player of the host team in the 10th game. MAP = 0.1711.

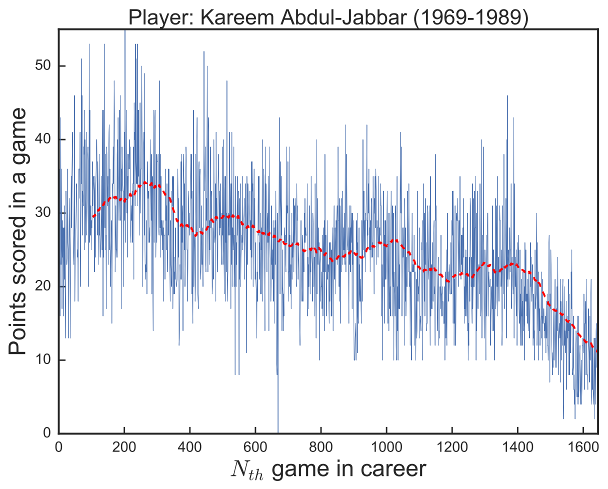

Figure 12: Raw data of points per game in Kareem's entire career.

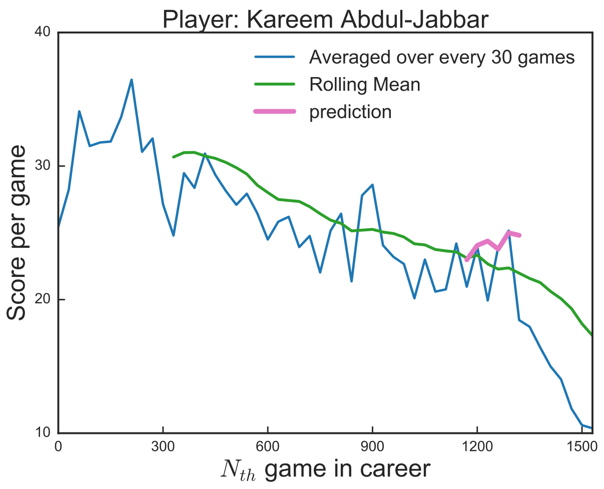

Figure 13: Rolling mean and result of prediction based on ARMA model.