2009-03-11

Measuring Skill Importance in Women's Soccer and Volleyball

Michelle L. Allan
Brigham Young University - Provo

MEASURING SKILL IMPORTANCE IN WOMEN'S SOCCER AND VOLLEYBALL

by

Michelle L. Allan

A project submitted to the faculty of

Brigham Young University

in partial fulfillment of the requirements for the degree of

Master of Science

Department of Statistics

Brigham Young University

April 2009

BRIGHAM YOUNG UNIVERSITY

GRADUATE COMMITTEE APPROVAL

of a project submitted by

Michelle L. Allan

This project has been read by each member of the following graduate committee and by majority vote has been found to be satisfactory.

Date	Gilbert W. Fellingham, Chair
Date	C. Shane Reese
Date	David A. Engler

BRIGHAM YOUNG UNIVERSITY

As chair of the candidate's graduate committee, I have read the project of Michelle L. Allan in its final form and have found that (1) its format, citations, and bibliographical style are consistent and acceptable and fulfill university and department style requirements; (2) its illustrative materials including figures, tables, and charts are in place; and (3) the final manuscript is satisfactory to the graduate committee and is ready for submission to the university library.

Date

Gilbert W. Fellingham
Chair, Graduate Committee

Accepted for the Department

Scott D. Grimshaw
Graduate Coordinator

Accepted for the College

Thomas W. Sederberg
Associate Dean, College of Physical and Mathematical Sciences

ABSTRACT

MEASURING SKILL IMPORTANCE IN WOMEN'S SOCCER AND VOLLEYBALL

Michelle L. Allan

Department of Statistics

Master of Science

The purpose of this study is to demonstrate how to measure skill importance for two sports: soccer and volleyball. A division I women's soccer team filmed each home game during a competitive season. Every defensive, dribbling, first touch, and passing skill was rated and recorded for each team. It was noted whether each sequence of plays led to a successful shot. A hierarchical Bayesian logistic regression model is implemented to determine how the performance of the skill affects the probability of a successful shot. A Division I women's volleyball team rated each skill (serve, pass, set, etc.) and recorded rally outcomes during home games in a competitive season. The skills were only rated when the ball was on the home team's side of the net. Events followed one of these three patterns: serve-outcome, pass-set-attack-outcome, or dig-set-attack-outcome. We analyze the volleyball data using two different techniques, Markov chains and Bayesian logistic regression. These sequences of events are assumed to be first-order Markov chains. This means the quality of the current skill only depends on the quality of the previous skill. The count matrix is assumed to follow a multinomial distribution, so a Dirichlet prior is used to estimate each row of

the count matrix. Bayesian simulation is used to produce the unconditional posterior probability (e.g., a perfect serve results in a point). The volleyball logistic regression model uses a Bayesian approach to determine how the performance of the skill affects the probability of a successful outcome. The posterior distributions produced from each of the models are used to calculate importance scores. The soccer data importance scores revealed that passing, first touch, and dribbling skills are the most important to the primary team. The Markov chain model for the volleyball data indicates setting 3–5 feet off the net increases the probability of a successful outcome. The logistic regression model for the volleyball data reveals that serves have a high importance score because of their steep slope. Importance scores can be used to assist coaches in allocating practice time, developing new strategies, and analyzing each player's skill performance.

ACKNOWLEDGEMENTS

There are many individuals who have helped me complete this project. I would like to thank Dr. Gilbert Fellingham for his insight and guidance through this master's project; the BYU Statistics Department for all their encouragement and support; my parents, for their support in believing that I can complete anything I set my mind to, their love, and their unfailing interest in what I am doing; and lastly my fiance, Ben, for his unfailing patience and support.

CONTENTS

TABLES

Table

1. INTRODUCTION

A sports team is ranked according to how well they perform during a game. The ultimate reflection of a team's performance is based on the number of points the team scores during a game or the number of games the team wins during a season. The number of points that a team scores during a game is ultimately based on how well the team performs key skills. During a game, the coach evaluates the performance of the players and determines how well the team performs key skills. Understanding how the performance of skills relates to the scoring of points would be useful for athletes and coaches in all sports. If a team can quantitatively understand how their performance of skills relates to the number of points scored, the coach could then adjust the team's practice time to focus on improving the performance of these key skills.

A division I university women's soccer team used a notational analysis system to analyze their skill performance and the skill performance of ten of their opponents during the 2005 competitive season. Each game was filmed and all defensive, dribbling, first touch, and passing skills were evaluated and recorded for each team. Each of the skills was evaluated using a grading rubric in order to evaluate how well the skill was performed. The grading rubric for each skill varied according to the number of distinct ways to classify the performance of a skill.

A division I university women's volleyball team used a similar notational analysis system to evaluate their skill performance during the 2006 competitive season. Each serve, pass, attack, and dig was evaluated and recorded as the skill was performed. Sets were evaluated and recorded after viewing the game film. Each skill was evaluated using a grading rubric in order to quantify how well the skill was performed.

To determine the impact of each skill in scoring a point, importance scores are used. Fellingham and Reese (2004) suggest that by analyzing posterior distributions, a

measure could be created that accounts for the impact and the amount of uncertainty in performing a skill. Importance scores could be useful to coaches since they would indicate how the impact of a particular skill relates to the probability of a successful outcome. Importance scores would also be useful because they would allow coaches to compare skill performance across teams.

In this paper we explore two different methods of measuring skill importance using data from two sports: soccer and volleyball. For the volleyball data, we use a Markov chain transition matrix and Gibbs sampling to arrive at posterior distributions. For the soccer data, we use Bayesian hierarchical methods in conjunction with logistic regression to compute the posterior distributions. The two methods of computing importance scores are used to quantify how the performance of skills relates to the scoring of points. A more precise way to quantify skill importance would allow coaches to allocate practice time more efficiently so the team could work on skills that increase the probability of scoring points.

2. LITERATURE REVIEW

The literature review is divided into eight parts. Section 2.1 discusses Bayesian models and two techniques used to draw samples from the posterior distribution. Section 2.2 reviews previous research in soccer. Section 2.3 discusses hierarchical Bayesian logistic regression. Section 2.4 reviews previous research in volleyball. Section 2.5 explains properties of Markov chains and methods used to estimate transition probabilities. Section 2.6 explores Bayesian estimates to Markov chains. Section 2.7 discusses how to use importance scores to calculate skill importance.

2.1 Bayesian Models

2.1.1 Theory of Bayesian Models

Bayes theorem was presented by Thomas Bayes in the *Essay Towards Solving a Problem in the Doctrine of Chances* in 1764. Bayesian models incorporate Bayes theorem, which states

$$\pi(\boldsymbol{\theta}|y) = \frac{f(y|\boldsymbol{\theta})\pi(\boldsymbol{\theta})}{\int f(y|\boldsymbol{\theta})\pi(\boldsymbol{\theta})d\boldsymbol{\theta}}, \tag{2.1}$$

where $\pi(\boldsymbol{\theta}|y)$ is the posterior distribution, $f(y|\boldsymbol{\theta})$ is the likelihood, $\pi(\boldsymbol{\theta})$ is the prior distribution, y is the data vector, and $\boldsymbol{\theta}$ is the vector of parameters (Gelman et al. 2004). The parameters of the prior distribution are based on an individual's *a priori* knowledge, or belief before the data are collected. The denominator of Equation 2.1 is referred to as the normalizing constant. Often the normalizing constant is difficult to find because the integral is so complicated.

Conjugacy is defined as follows. If $\mathcal{F}$ is a class of sampling distributions from $f(y|\boldsymbol{\theta})$ and $\mathcal{P}$ is a class of sampling distributions from $\pi(\boldsymbol{\theta})$, then $\mathcal{P}$ is conjugate for $\mathcal{F}$ if

$$\pi(\boldsymbol{\theta}|y) \in \mathcal{P} \text{ for all } f(\cdot|\boldsymbol{\theta}) \in \mathcal{F} \text{ and } f(\cdot) \in \mathcal{P}. \tag{2.2}$$

Of most interest are natural conjugate prior families. Natural conjugate families occur when the prior distribution has the same functional form as the likelihood. Conjugate priors are useful because draws can be obtained directly from the posterior distribution using Gibbs sampling, a simple case of Markov chain Monte Carlo (Gelman et al. 2004).

2.1.2 Methods to Sample from the Posterior Distribution

In some simple Bayesian models, it is possible to draw directly from the posterior distribution. This often happens when the prior distributions are conjugate. Usually, however, Bayesian models yield complex posteriors which cannot be sampled from directly. Thus, a different approach is needed to make draws from the posterior distribution. Markov chain simulation, also known as Markov chain Monte Carlo (MCMC), is the most common posterior simulation method. The goal of MCMC simulation is to draw values of $\boldsymbol{\theta}$ from approximate distributions and then correct those draws to model the target posterior distribution. The samples are drawn sequentially, so the distribution of the sampled draws depends on the previous value drawn. Thus, the sequence of draws forms a Markov chain. There are different ways to draw from the posterior distribution using Markov chains. Two posterior simulation techniques that are commonly used are Gibbs sampling and Metropolis-Hastings (Gelman et al. 2004).

Gibbs Sampling

Each iteration of the Gibbs sampler (also called alternating conditional sampling) is dependent on the value of the previous iterations. The Gibbs sampler updates each of the k parameters. The Gibbs sampler algorithm is as follows (Gelman et al. 2004):

(1) Start by assigning initial values $\boldsymbol{\theta}^{(0)} = (\theta_1^{(0)}, \theta_2^{(0)}, \ldots, \theta_k^{(0)})$, where $\boldsymbol{\theta}^{(0)}$ can be any value that is located inside the parameter space.

(2) Let $i = 1$.

(3) Generate a $\theta_1^{(i)}$ from the complete conditional distribution

$[\theta_1|\theta_2^{(i-1)}, \theta_3^{(i-1)}, \ldots, \theta_k^{(i-1)}, y]$, which is the conditional distribution of θ_1 given the most recent values of all the other parameters.

(4) Generate a $\theta_2^{(i)}$ from the complete conditional distribution,

$[\theta_2|\theta_1^{(i)}, \theta_3^{(i-1)}, \ldots, \theta_k^{(i-1)}, y]$.

(5) Continue this process until a value for $\theta_k^{(i)}$ has been generated from the complete conditional distribution, $[\theta_k|\theta_1^{(i)}, \theta_2^{(i)}, \ldots, \theta_{k-1}^{(i)}, y]$.

(6) These draws form the vector $\boldsymbol{\theta}^{(i)} = (\theta_1^{(i)}, \theta_2^{(i)}, \ldots, \theta_k^{(i)})$.

(7) Repeat steps three through six for $i = 2, \ldots, M$.

As the limit of M goes to infinity, $\boldsymbol{\theta}^{(M)}$ converges to the joint posterior distribution $\pi(\boldsymbol{\theta}|y)$, provided that the Markov chain is aperiodic and irreducible (Ross 1996).

Metropolis-Hastings

The Metropolis-Hastings algorithm uses an acceptance/rejection rule to draw samples from the appropriate distribution. The algorithm that implements Metropolis-Hastings is as follows (Gelman et al. 2004):

(1) Start by assigning an initial value to θ_0, where θ_0 can be any value that is located inside the parameter space.

(2) Let $i = 1$.

(3) Generate a candidate value θ^* from a candidate density, $p(\theta^*|\theta_{i-1})$. $p(\theta^*|\theta_{i-1})$ needs to be a distribution from which one can sample. $p(\theta^*|\theta_{i-1})$ is often chosen to be a symmetric distribution so some of the computations can be simplified.

(4) Let $\alpha(\theta_{i-1}, \theta^*) = \frac{g(\theta^*)p(\theta^*|\theta_{i-1})}{g(\theta_{i-1})p(\theta_{i-1}|\theta^*)}$, where $\alpha(\theta_{i-1}, \theta^*)$ is the probability of accepting a move from θ_{i-1} to θ^*, and $g(\cdot)$ is the unnormalized posterior distribution.

(5) Generate a value $u_i \sim \mathit{Uniform}(0, 1)$.

(6) If $u_i < \alpha(\theta_{i-1}, \theta^*)$ then $\theta_i = \theta^*$, else let $\theta_i = \theta_{i-1}$.

(7) Repeat steps three through six for $i = 2, \ldots, M$.

As the limit of M goes to infinity, $\theta_1, \theta_2, \ldots, \theta_M$ converges to the target distribution. The convergence occurs if the Markov chain is irreducible, aperiodic, and not transient (Gelman et al. 2004).

The Metropolis-Hastings algorithm is used most often in conjunction with the Gibbs sampler. If the complete conditionals cannot be directly sampled, the Metropolis-Hastings algorithm is then used.

2.2 Previous Research in Soccer

Pollard and Reep (1997) state that because of the importance of winning, anything that gives a soccer coach a slight advantage over other teams is worthy of investigation. Performance analysis allows coaches to quantify the performance of skills to see how they influence the outcome of the game. Various performance indicators have been developed to assess the impact of the performance of a skill in soccer (Reep and Benjamin 1968; Ali 1988; Bate 1988; Pollard and Reep 1997; Hughes and Bartlett 2002). Hughes and Bartlett (2002) report that analysts and coaches use

performance indicators to assess the performance of a team. Performance indicators are usually related to an outcome, and they can be used to compare two teams.

Thomas (2005) develops a notational analysis system for women's college soccer and uses a Bayesian regression approach to model the outcome as a function of the standardized performance score. Her technique calculates standardized performance scores for four skills: defense, passing, first touch, and dribbling. The study looks at a particular team's home games and analyzes their performance against their opponents' performance. Thus, she is interested in the difference between performance scores of the home team and the away teams. She looks at the overall importance scores for the four skills to determine which skills were most important to the outcome.

Hughes and Franks (2005) use a negative binomial approach to relate the length of a passing sequence to the number of goals scored by the team. This study uses the databases from the 1990 and 1994 FIFA world cup finals. They define a passing sequence as the number of passes that are performed before a shot is taken. In the 1990 world cup finals, they found that 84% of the goals came from passing sequences that are four passes or less. The 1994 world cup finals revealed that 80% of the goals came from passing sequences that were four passes or less. They note that this might be a facet of the data because more short passes are performed than long passes. This study also shows that if the teams have the skill to sustain possession for longer periods of time, the probability of scoring a goal increases, but the conversion ratio of shots to goals for longer possessions decreases.

McHale and Scarf (2007) use negative binomial and bivariate Poisson copulas to show the relationship between skills and shots. They analyze 1,048 soccer matches and record how many key skills are performed for the home and away teams. Some of the skills they recorded are the number of goals, shots, crosses, dribbles, passes made, and interceptions. Their study shows that passes and crosses characterize a successful team. In particular, they show that away teams are more likely to convert

crosses to shots than home teams.

Van Calster and Van Huffel (2008) analyze teams from fifty domestic leagues for three seasons from 2003 to 2006. They record the number of goals scored (offensive ability) and the number of goals allowed (defensive ability) per game . A team's overall ability is measured by the mean number of points scored per game. A team's entertainment ability is measured by the mean number of goals scored during a game; this includes the number of goals scored and the number of goals allowed. A team's overall, defensive, offensive, and entertainment ability are used to predict the percentage of scoreless draws for the team. Using two different methods, least squared support vector machines for regression and the hybrid Monte Carlo algorithm, this study indicates that a team's overall ability is related to the percentage of scoreless draws, with average teams having more scoreless draws. They also note that the defensive and offensive ability of a team affects the percentage of scoreless draws.

The project that we propose builds on the previous work in performance analysis of women's soccer and adds an extra dimension. Most research uses quantified offensive, defensive, and passing skills as an overall metric. Van Calster and Van Huffel (2008) incorporate offensive and defensive skills by counting the number of goals scored and the number of goals allowed. Hughes and Franks (2005) look at how the number of passes before a shot affects the probability of a goal.

Thomas (2005) actually looks at rating each individual skill as it is performed during the game to see how the performance of an individual skill can affect the probability of a shot or goal for a team or the opponent. This project adds an extra dimension to Thomas (2005) as it uses Bayesian logistic regression and hierarchical modeling to incorporate how the performance of a skill relates to the probability of taking a successful shot. A shot is considered successful if it is on target or a goal is made. A shot is on target if the shot is blocked by the goalie. The data set used for this analysis is provided by a division I women's soccer team and includes a rating

for every skill performed during ten home game soccer matches. The skill importance for the primary team and its ten opponents will be calculated to compare how each of the teams performed key skills relative to each other.

2.3 Hierarchical Logistic Regression

Gelman et al. (2004) state that hierarchical models can be used when an application involves many parameters that are thought to be related in some way by the nature of the problem. This means that the joint probability model for these parameters should reflect this dependence. A natural way to model this dependence is to allow the prior distribution for each of the dependent parameters to be a sample from a common population distribution. Each team has their own coefficient for recording how a particular skill affects the probability of a successful shot. Using hierarchical modeling, we will allow each parameter associated with the performance of a specific skill to come from a common population distribution. Each team's parameter is a draw from that distribution. Note that this still allows some teams to perform the skills better than others.

With soccer, we are concerned with whether or not a shot is successful. Whether or not a shot is successful depends on the performance of the skills leading up to a shot. Thus, logistic regression is used to relate the quality of the skills leading up to a shot to the probability of a successful shot. Logistic regression uses the log odds ratio to relate the quality of skills performed to whether or not those skills resulted in a successful shot.

A binary regression model can be created using logistic regression. One of the most common transformations used in logistic regression is the logit link. The logit link, which is in terms of the log odds ratio, is defined as

$$\log\left(\frac{p_i}{1-p_i}\right) = \alpha + \beta x_i, \tag{2.3}$$

where p_i is the probability of a successful outcome for observation i, α is the overall intercept, β is the effect associated with covariate x, and x_i is the i^{th} observation of covariate x (Gelman et al. 2004).

We will expand the simple logistic regression model, shown in Equation 2.3, to incorporate the hierarchical modeling associated with the four skills, and we will allow α to change with each team.

2.4 Previous Research in Volleyball

Zetou et al. (2007) explain how the statistical evaluation of a team's skill performance helps considerably with the development of the game of volleyball. Understanding how the performance of skills relates to the scoring of points is useful for athletes and coaches. Hughes and Daniel (2003) state that volleyball has not benefited from in-depth performance analyses. Considerable research has been done on developing notational analysis in volleyball. The notational analysis that is most commonly used was developed by Coleman (1975). Florence and Fellingham (2008) use a notational analysis that not only grades the performance of skills but bases the performance of skills on the outcome of the rally or the opponent's performance.

Hughes and Daniel (2003) focus on understanding the playing patterns of elite and non-elite volleyball teams by using a notational analysis system. They filmed twenty sets to analyze the technical, tactical, and court utilization of elite and non-elite volleyball teams. They use Chi-square tests to show that there is a significant difference between the playing patterns of elite and non-elite volleyball. This analysis shows that elite teams are significantly better at serving and receiving than the non-elite teams. This study also shows that the quality of the attack is dependent on the quality of the set, and that the quality of the set is dependent on the quality of the defense or the reception of the serve.

Yiannis and Panagiotis (2005) compare the effectiveness of five key skills in

men's volleyball between the Sydney 2000 and the Athens 2004 Olympic Games. This study uses Chi-square tests to compare the results for each of the elite teams across Olympic Games, and t-tests are used to compare the effectiveness of each team to the overall mean. They show that from the 2000 to 2004 Olympic Games there was a general trend of decreasing the proportion of mistakes performed on all the skills. In particular, this study shows that during the 2004 Olympic Games, Brazil, the gold medalist, performed their reception and attack skills far better than the teams they competed against.

Zetou et al. (2007) analyze the playing characteristics of teams to determine which characteristics led to a win and contributed to the final ranking of the teams. Discriminant analysis is used to determine which characteristics significantly contribute to a win or loss for Olympic volleyball teams. To determine which skills are most important for a win or a loss, stepwise selection is used. The jackknife (leave-one-out) method is used to determine the accuracy of the classifications. The study shows that for a serve-reception skill, the best predictor of a win is the receiver's ability to make the best reception possible so the setter can set for a first tempo attack or to set a high set to the outside hitter in zone four or two. For the attack from reception skill, the analysis shows that an "ace-point" is the most significant predictor in determining a winning team.

Florence and Fellingham (2008) use Bayesian models to estimate the probability of transitioning from one skill to another skill. Markov chain Monte Carlo methods are used to produce draws from the posterior distribution. This study calculates unconditional probabilities of moving from one skill to a specific outcome. The unconditional probabilities indicate ways to improve notational analysis and the performance of volleyball skills. They note that rating a passer according to passing average is not appropriate because the difference between a one-point pass and a two-point pass is not equivalent to the difference between a two-point pass and

a three-point pass. They also show that to increase the probability of a successful outcome, the passers should target their pass further off the net. The analysis also shows that if back sets that are high and inside are avoided, the attack has a higher probability of being successful.

This project builds on previous research but adds an extra dimension by classifying the importance of performing an individual skill on a specific outcome. We use the same approach discussed by Florence and Fellingham (2008) to calculate the unconditional probabilities of moving from one skill to a specific outcome, produced by Gibbs sampling. The data set used for this analysis was provided by a division I volleyball team, and includes a rating for every skill that was performed during the 2006 competitive season. We use the posterior distributions produced from the unconditional probabilities to calculate the importance scores for each skill. This enables us to see which skills are considered the most important when trying to perform skills with a successful outcome.

2.5 Markov Chains

2.5.1 Properties of Markov Chains

Volleyball skills are performed in a fairly rigid pattern (e.g., pass-set-attack). Thus, it makes sense to treat these volleyball patterns as Markov chains. Therefore, we approach the problem by treating the skills performed by the division I women's volleyball team as states and estimating the probability of transitioning from one state to another state. A Markov chain is a sequence of random variables in which the value of the current random variable depends only on the value of the previous random variable. More formally, a Markov chain is defined as

$$P[X_{n+1} = x_{n+1} | X_0 = x_0, X_1 = x_1, \ldots, X_n = x_n] = P[X_{n+1} = x_{n+1} | X_n = x_n], \quad (2.4)$$

where X is a random variable and x_n is the state in the Markov chain sequence that

occurred at time n (Stewart 1994). The transition probability matrix contains all the probabilities of transitioning from one state to another state.

Properties of Markov chains allow every state to be included in the transition probability matrix. A state is *recurrent* if the probability of ever returning to the state is one. This means that the Markov chain will return to this state in the future. A *transient* state means the probability of ever returning to that state is less than one. When state j is recurrent, the mean recurrence time, M_{jj}, of state j is defined as

$$M_{jj} = \sum_{n=1}^{\infty} n f_{jj}^{(n)}, \tag{2.5}$$

where n is the number of steps and $f_{jj}^{(n)}$ is the probability of transitioning from state j to state j in n steps. If state j is recurrent and M_{jj} is finite, then state j is called a *positive-recurrent* state (Stewart 1994, Ross 1996). If every state can be transitioned to from every other state, then the Markov chain is *irreducible* (Ross 1996). If it is only possible to return to a given state in a number of transitions that is a multiple of p, then the given state is said to be *periodic* with period p, where $p > 1$. When $p = 1$, the state is said to be *aperiodic* (Stewart 1994). This means the state can transition to itself in one step. The invariant (stationary) distribution exists if the Markov chain is irreducible, aperiodic, and the states are positive recurrent. The invariant distribution can also be defined in another way. A probability distribution $\mathbf{z}$, where $\mathbf{z}$ is the vector of elements containing the probabilities of transitioning from state i to any other state, is defined as the invariant distribution if and only if $\mathbf{z} = \mathbf{zP}$, where $\mathbf{P}$ is the transition probability matrix (Stewart 1994). It is important for Markov chains to converge to the invariant distribution, especially when using Markov chain Monte Carlo simulation (see section 2.5).

2.5.2 Estimating Transition Probabilities

Many different methods have been used to estimate transition probabilities. Miller (1952) and Tesler (1966) use the least squares method to estimate the transition probabilities when only sample proportions are available from time series data. A problem with this method is that sometimes the least squares method results in negative estimates of the transition probabilities. Thus, new techniques are developed in order to allow for restrictions to be placed on the estimates of the transition probabilities. Restricted least squares is based on a quadratic programming iteration method that is proposed by Lee et al. (1965) and Theil and Rey (1964). This estimation method does not account for heteroscedasticity and is not asymptotically efficient, as noted by Mandansky (1959). Mandansky (1959) suggests using weighted least squares to help correct the problem of heteroscedastic errors.

Lee et al. (1968) conduct a study to compare classical least squares, weighted least squares, maximum likelihood, and Bayesian estimates of the transition probabilities. The Bayesian estimator of the transition probability matrix is computed using a multinomial likelihood and a Dirichlet prior. A first-order stationary Markov chain with four states is used to generate 50 samples of size 25, 50, and 100. To evaluate the performance of each of the transition probability estimators, the root mean square error is calculated for each transition probability matrix. The study also uses an overall root mean square error, Wilcoxon matched-pair signed-rank test, Kendall's coefficient of concordance, and Kolmogorov-Smirnov's goodness-of-fit test to evaluate the performance of each of the estimators. The study finds that the Bayesian estimator performs better than the classical least squares, weighted least squares, and the maximum likelihood estimators using the above criterion.

2.6 Bayesian Estimation of Markov Processes

When estimating transition probabilities using a Bayesian approach, many researchers use a multinomial likelihood with a Dirichlet prior distribution (Lee et al. 1968; Meshkani and Billard 1992; Assoudou and Essebbar 2003; Zhao et al. 2005). Other Bayesian methods of estimating transition probabilities include Jeffrey's prior distribution or a normal prior distribution (Assoudou and Essebbar 2003 and Cargnoni et al. 1997).

Assoudou and Essebbar (2003) conducted a simulation study comparing the maximum likelihood estimator of the transition probabilities to the Bayesian estimators using Jeffrey's prior distribution and a Dirichlet prior distribution. This study simulated 20 two-state Markov chains with a sample size of 21. The study also simulated 10 three-state Markov chains with a sample size of 61. For the two-state and three-state Markov chains, the study showed that both the Jeffrey's and Dirichlet prior Bayesian estimators of the transition probabilities have a smaller mean square error than the maximum likelihood estimator. Thus, the Bayesian estimators perform better than the maximum likelihood estimator.

To calculate Bayesian estimates of the transition probabilities, the posterior mean or mode is often used (Lee et al. 1968; McKeigue et al. 2000; Ozekici and Soyer 2003). Degroot (1970) illustrated that the expected value of the posterior distribution is the best performing Bayesian estimator when compared to the quadratic loss function.

2.7 Importance Scores

Fellingham and Reese (2004) suggest that in order to measure the impact and uncertainty in performing a specific skill, importance scores should be calculated. Thus, for a specific team, they define the coefficient of skill importance as the ratio

$$I_i = \frac{E(\beta_i|Y)}{\sqrt{V(\beta_i|Y)}},$$

where i corresponds to the skill and β_i indicates the coefficient associated with skill i. This provides the ability to obtain an importance score for each skill for every team. The importance scores within a team are ordered from largest to smallest. The larger importance scores indicate that these skills are more important in performing a successful outcome. An advantage of importance scores is the ability to incorporate not only the impact in performing specific skills, but also the uncertainty. Since uncertainty is associated with sample size, the skills that are performed the most often receive larger importance scores. Thus, when comparing importance scores across teams, the magnitude of the importance scores should not be compared. Only the ordering of importance scores between teams should be compared.

Fellingham and Reese (2004) were interested in determining skill importance for the United States Men's National Volleyball Team. To determine which of the skills the United States Men's National Volleyball Team needed to improve, they analyzed the skill performance of the following teams: the United States, Brazil, France, Yugoslavia, Italy, Argentina, Cuba, Russia, Greece, and the Netherlands. They created two models. One model separated attack by position and attack by front or back row. Another model combined all the attacks into one category and created a new category called defend. Using importance scores, this study found that attacking is the most important skill. In particular, for the model that separated attacks by position and front and back row, five out of the seven top skills are related to kills. Using the attack/defend model, the most important skill was the attack kill.

3. PAPER FOR THE *JOURNAL OF QUANTITATIVE ANALYSIS IN SPORTS*

3.1 Introduction

A sports team's success is determined by the number of contests won during a season. Winning a contest generally means scoring more points than the opponent. The number of points that a team scores during a game is primarily based on how well the team performs key skills. Understanding how the performance of skills relates to the scoring of points is useful for athletes and coaches in all sports. If a coach can quantitatively understand how the performance of various skills relates to the number of points scored, the coach could then adjust the team's practice schedule to focus on improving the performance of key skills that are more closely tied to point scoring.

To determine the impact of each skill in scoring a point, Fellingham and Reese (2004) suggest the use of importance scores. An importance score would be a measure that could account for both the impact and the amount of uncertainty associated with the performance of a skill relative to the probability of scoring a point. Importance scores would be useful to coaches since they indicate how the impact of a particular skill relates to the probability of a successful outcome. Importance scores would also allow coaches to compare skill performance across teams.

A division I women's volleyball team used a notational analysis system to evaluate their skill performance during the 2006 competitive season. Each serve, pass, attack, and dig was evaluated and recorded as the skill was performed. Sets were evaluated and recorded after viewing the game film. Each skill was evaluated using a grading rubric in order to quantify how well the skill was performed.

We explore two different methods of measuring skill importance using this data from women's volleyball. One method uses a Markov chain transition matrix and Gibbs sampling to arrive at posterior distributions for parameters associated with

skill performance. The second method uses Bayesian logistic regression to compute the posterior distributions. These posterior distributions are then used to compute importance scores. The two methods of computing importance scores are then compared relative to the importance associated with various skills.

Section 3.2 examines previous research in skill importance and volleyball. Section 3.3 discusses the data. Section 3.4 explains the methodology used to calculate posterior distributions using logistic regression and Markov chains. Section 3.5 presents the importance scores. Lastly, section 3.6 compares the results from the two methods.

3.2 Literature Review

3.2.1 Measuring Skill Importance

In order to measure skill importance, we use the importance score, a metric suggested by Fellingham and Reese (2004). For a specific team, they define the coefficient of skill importance as the ratio

$$I_i = \frac{E(\beta_i|Y)}{\sqrt{V(\beta_i|Y)}},$$

where i corresponds to the skill and β_i indicates the posterior distribution associated with skill i given the data, Y. This provides the ability to obtain an importance score for each skill. The importance score incorporates not only the impact of a specific skill ($\mathrm{E}(\beta_i|Y)$), but also the uncertainty associated with the performance ($V(\beta_i|Y)$). Thus, a skill whose association with scoring a point is less certain will be penalized when using this metric when compared to a skill where performance at a given level is more closely associated with a positive outcome.

3.2.2 Previous Research

Considerable research has been done on developing notational analysis in volleyball (Zetou et al. 2007). The notational analysis that is most commonly used was developed by Coleman (1975). Florence and Fellingham (2008) used a notational system to evaluate skills in women's volleyball. They concluded the target area for a pass should be further from the net.

Hughes and Daniel (2003) focused on understanding the playing patterns of elite and non-elite volleyball teams. Their analysis showed that elite teams were significantly better at serving and receiving than non-elite teams. The study also showed that the quality of the attack was dependent on the quality of the set, and that the quality of the set was dependent on the quality of the defense or the reception of the serve.

Yiannis and Panagiotis (2005) compared the effectiveness of five key skills in 10 men's volleyball matches that occurred between the Sydney 2000 and the Athens 2004 Olympic Games. They showed that from the 2000 to 2004 Olympic Games there was a general trend of decreasing the proportion of mistakes performed on all the skills. In particular, the study showed that during the 2004 Olympic Games, Brazil, the gold medalist, performed their reception and attack skills far better than the teams they competed against.

Zetou et al. (2007) explained how the statistical evaluation of a team's skill performance helped considerably with the development of the game of volleyball. Their study showed that for a serve-reception skill, the best predictor of a win was the receiver's ability to make the best reception possible so the setter could set for a first tempo attack or set a high set to the outside hitter in zone four or two. For the attack from reception skill, the analysis showed that an "ace-point" is the most significant predictor in determining a winning team.

3.3 The Data

The data were collected during the 2006 competitive season of a women's division I volleyball team. During each home game, the game was recorded, and the skills were analyzed when the ball was on the home team's side of the net. Thus, these data come from a single team. Using the performance scoring system developed by Coleman (1975) as a guide and consulting the expertise of volleyball coaches and researchers, each technique was rated. The women's volleyball data were recorded into a system called Data Volley (Data Project, Salerno, Italy, release 2.1.9). The grading system of the skills was created based on the number of codes Data Volley could handle.

Serves were rated on a five-point (0–4) scale, passes on a six-point (0–5) scale, and digs on a six-point (0–5) scale. Attacks were noted by position on the court (back row, left side, right side, middle, etc.). Sets were rated according to distance from the net (0–3 feet, 3–5 feet, etc.). A detailed breakdown of all the skill ratings are shown in Table 3.1. There were three possible outcomes: (1) point for the home team, (2) rally continuation, or (3) point for the visiting team.

During a game, a trained member of the volleyball team's coaching staff rated and recorded every serve, pass, dig, and attack performed by the team during the 13 home games in the 2006 season. When a set occurred, a default code was inserted into the system so that the set could be rated later while looking at the film of the game. Only serves and attacks for the opposing team were recorded. This was done to determine when the ball crossed the net. The score of the game was determined at the end of the rally by noting which team was the next to serve. Since the data set only consisted of touches on the primary team's side of the net, a continuation of a rally was determined by seeing if the ball came back to the primary team's side during a rally.

Table 3.1: Performance ratings for all the skills.

Skill	Performance Rating
Serves	
Ace Serve	4
3-point Serve	3
2-point Serve	2
1-point Serve	1
Service error	0
Passes	
4-point Pass	5
3-point Pass	4
2-point Pass	3
1-point Pass	2
Overpass	1
Passing error	0
Digs	
5-point Dig	5
4-point Dig	4
3-point Dig	3
2-point Dig	2
1-point Dig	1
Digging error	0
Set Distances	
3–5 feet	3
0–3 feet	2
5–8 feet	1
8–10 feet	0

3.4 Methods for the Volleyball Analyses

Since importance scores were defined in terms of Bayesian techniques, Bayesian methods were used for the analyses. The framework presented by Lindley (1964) and Leonard (1972) were used to construct the Bayesian logistic regression models. The work of Lee et al. (1968) and Assoudou and Essebbar (2003) was used as a guide to construct the first-order Markov chain model.

3.4.1 Markov Chain Approach

Whenever the ball was on the primary team's side of the net, one of the following sequences of events occurred: serve-outcome, pass-set-attack-outcome, or dig-set-attack-outcome. There were three possible outcomes: a point for the primary team, a point for the opponent, or a continuation of the rally.

We assumed that these sequences of events (serve-outcome, pass-set-attack-outcome, and dig-set-attack-outcome) were first-order Markov chains. We represented these sequences in a transition matrix where the elements of the matrix comprise the probability of moving from one state to another state (e.g., a four-point pass to a set 3–5 feet off the net).

The 36×36 transition matrix contained the transitions for float serves, jump serves, passes, set distances off the net, attacks, digs, and possible outcomes. There were five different ratings for both float and jump serves, six ratings for passes, four ratings for sets, seven places for attacks (left, right, middle, back row, set dump, out of system, and over pass attack), six ratings for digs, and three outcomes of a rally (continuation, point for home team, point for visiting team). Thus, the transition matrix had $5 + 5 + 6 + 4 + 7 + 6 + 3 = 36$ rows and columns.

Sequences that were impossible (e.g., a four-point pass to an ace serve) were constrained to have zero probability. Sequences that always occurred (e.g., an ace serve to a point) were constrained to have a probability of one. Data were organized

in a count matrix. Thus, y_{ij} is the $(i,\ j)^{th}$ element of the count matrix and is the number of times play moved from state i to state j during the season.

We used Bayesian methods to estimate the transition probabilities. We assumed a multinomial likelihood

$$f(y_{i1}, \ldots, y_{ik}|\pi_{i1}, \ldots, \pi_{ik}) \propto \pi_{i1}^{y_{i1}} \pi_{i2}^{y_{i2}} \ldots \pi_{ik}^{y_{ik}}, \tag{3.1}$$

for each row, $i = 1, \ldots, m$, of the count matrix, where k is the number of possible states that could occur in the next sequence of touches, and m is the number of states. π_{ij} represents the probability of moving from state i to state j in the transition probability matrix, and $\sum_{j=1}^{k} \pi_{ij} = 1$, for each i.

We assumed the prior probability densities of each row were distributed as Dirichlet random variables

$$f(\pi_{i1}, \ldots, \pi_{ik}|\alpha_{i1}, \ldots, \alpha_{ik}) \propto \pi_{i1}^{\alpha_{i1}-1} \pi_{i2}^{\alpha_{i2}-1} \ldots \pi_{ik}^{\alpha_{ik}-1}, \tag{3.2}$$

where α_{ij} represents the expectation of how often we think the women's team moves from state i to state j relative to moving to a different state in the transition probability matrix. Since we are interested in what the data indicates about the association between the different states, we assumed weak prior information. Thus, the prior counts, α_{ij}, were all assumed to be equal to one (except those that were constrained to be zero).

Because of the conjugacy that exists between the multinomial distribution and the Dirichlet prior, Gibbs sampling can be used to produce draws from the posterior distribution

$$f(\pi_{i1}, \ldots, \pi_{ik}|y_{i1}, \ldots, y_{ik}, \alpha_{i1}, \ldots, \alpha_{ik}) \propto \pi_{i1}^{y_{i1}+\alpha_{i1}-1} \pi_{i2}^{y_{i2}+\alpha_{i2}-1} \ldots \pi_{ik}^{y_{ik}+\alpha_{ik}-1}, \tag{3.3}$$

for each row of the transition probability matrix. To make draws from the posterior distribution slightly more efficient, we drew $x_1, \ldots, x_k$ from independent gamma distributions with a shape parameter of $y_{i1} + \alpha_{i1}, \ldots, y_{ik} + \alpha_{ik}$ and a common scale

parameter of one. We used the independent draws from the gamma distributions to calculate a $\pi_{ij} = x_j / \sum_{j=1}^{k} x_j$ (Gelman et al. 2004). The posterior distributions of each π_{ij} were based on 100,000 draws.

In order to compute importance scores in this setting, we also calculated the unconditional probabilities of moving from one state (e.g., a four-point pass) to an outcome (e.g., a point for the primary team) and called this β. In order to obtain an estimate for the unconditional probability for every skill rating (i.e., β_i, $i = 1, \ldots, 25$), we used all possible sequences of touches that could occur between the state and the outcome in the transition probability matrix. For each sequence, we used the associated probabilities in the transition probability matrix. To obtain the overall unconditional probability, we summed the probabilities associated with each sequence.

Thus, at each step, we computed a draw of the unconditional probability using the current state of the transition probability matrix. Therefore, posterior distributions of the β_i are also based on 100,000 draws. Using the 100,000 draws associated with the posterior distribution for each β_i, the mean of the draws was used as an estimate of $E(\beta_i|Y)$ and the standard deviation of the draws was used as an estimate of $\sqrt{V(\beta_i|Y)}$. $E(\beta_i|Y)$ and $\sqrt{V(\beta_i|Y)}$ were used to calculate the importance score for the associated skill.

The appropriateness of the model was determined by constructing a Bayesian χ^2 goodness-of-fit test. The Bayesian χ^2 test was computed for each row and column combination of the transition matrix. To summarize all the Bayesian χ^2 tests, we calculated the average number of times the Bayesian χ^2 test resulted in a significant p-value. The Bayesian χ^2 test resulted in a significant p-value only 5.2% of the time. Thus, the first-order Markov chain does a reasonable job of modeling the association between the skill-rating combinations and a point for the primary team.

3.4.2 Bayesian Logistic Regression Approach

A Bayesian logistic regression model was also implemented to determine how the performance of individual skills affects the probability of scoring a point. The following skills were analyzed in this setting: jump serve, float serve, pass, dig, and set. Implementation of this model required that skills be scored in some fashion. Attacks were not analyzed this way because the grading of the attack was dependent on the outcome of the attack. Thus, the data set does not have an independent mechanism to grade the performance of the attack. Since sets were only noted in the data by distance off the net, the scoring of sets was determined by one of the investigators, a former volleyball coach. The ratings used for sets are shown in Table 3.2.

Table 3.2: Performance Ratings for Set Distance off the Net

Set Distance	Performance Rating
3–5 feet	3
0–3 feet	2
5–8 feet	1
8–10 feet	0

We modeled the response variable as a Bernoulli random variable. The response was given a zero if a point was not scored (opponent score or rally continued), and a one if a point was scored. Four separate models were constructed, one for each of the four skills. For each skill, we constructed a logistic regression model, where the log odds ratio was defined as

$$\log\left(\frac{Pr[Y_{ik}=1|\text{skill}=i]}{Pr[Y_{ik}=0|\text{skill}=i]}\right)=\beta_0+\beta_i\mathrm{R}_{ik}, \tag{3.4}$$

where i is associated with a specific skill, $k=1,\ldots,n_i$ corresponds to the number of times skill i is performed, β_0 is the primary team's overall ability, β_i is the effect of the i^{th} skill, and R_{ik} is the rating associated with the k^{th} time the i^{th} skill is performed.

Thus, $Y_{ik} \sim Bernoulli\left(\frac{exp(\beta_0+\beta_i \mathrm{R}_{ik})}{1+exp(\beta_0+\beta_i \mathrm{R}_{ik})}\right)$. The logistic regression model assumed that each skill was linearly related to the outcome. The linearity condition assumed that skill progression was linearly related to a positive outcome. A priori, we believed this assumption to be reasonable because we believed a skill performed with a higher rating would imply a higher probability of scoring a point.

The following prior distribution was chosen for each β_i parameter and the overall team effect (β_0)

$$f(\beta_i|m_i, s_i^2) \sim Normal(m_i, s_i^2).$$

The normal distribution was chosen because we expected the effect of a skill on the probability of scoring a point could be either positive or negative. Similar to the Markov chain model, weak prior information was assumed. We let each $m_i = 0$ and each $s_i^2 = 1000$.

The resulting posterior distributions for the β_i are not available in closed form. Thus, the Metropolis-Hastings algorithm was used to obtain draws from the posterior distributions. Mixing plots were used to ensure that sampling occurred from all parts of the distribution. Posterior distributions were estimated with 100,000 draws after a burn-in period of 500. Using the 100,000 draws associated with each skill, the mean of the draws for β_i was used as an estimate for $E(\beta_i|Y)$ and the standard deviation of the draws for β_i was used as an estimate for $\sqrt{V(\beta_i|Y)}$. $E(\beta_i|Y)$ and $\sqrt{V(\beta_i|Y)}$ were used to calculate the importance score for the associated skill.

To determine the appropriateness of each model, a Bayesian χ^2 goodness-of-fit test was calculated for each of the four models. For each model, to summarize all the Bayesian χ^2 tests, we calculated the average number of times the Bayesian χ^2 test resulted in a significant p-value. The Bayesian χ^2 test resulted in a significant p-value only 5.2% of the time. For serves, the Bayesian χ^2 goodness-of-fit test resulted in a significant p-value only 1.6% of the time. For passing, the Bayesian χ^2 goodness-of-fit test resulted in a significant p-value only 2.4% of the time. For setting, the Bayesian

χ^2 goodness-of-fit test resulted in a significant p-value only 4.6% of the time. For digging, the Bayesian χ^2 goodness-of-fit test resulted in a significant p-value only 2.0% of the time. These goodness-of-fit tests indicate that the proposed Bayesian logistic regression model does reasonably well modeling the probability of scoring a point.

3.5 Results

We developed the importance scores using two different methods. Thus, the importance scores themselves are not comparable across models. However, the relative rankings are comparable. Table 3.3 shows the importance scores using the Markov chain model. The importance scores for the logistic regression model are shown in Table 3.4.

3.6 Discussion

First, we need to again emphasize that it is possible for skills to have higher expected outcomes but lower importance scores if the variance associated with the parameter estimate is high. For example, a four-point pass has a higher probability of leading to a point than a three-point pass, but with a higher variance, the importance score is lower (Table 3.3). This is also true of left attack relative to both middle and right attack for this team. Left attack has a lower expected outcome, but a much higher importance score because it occurs more often than either a middle or right attack for this team. Nonetheless, the data would lead us to encourage the team to try to set to the middle and right sides more often.

The results seem, in general, to be fairly consistent. It is important to remember, however, that the methodologies lead to different interpretations. Using the Markov chain analysis, each skill-rating combination has an importance score attached. Using the logistic regression analysis, the importance score is for the entire

Table 3.3: Importance scores for the volleyball Markov chain analysis.

Skill	$E(\beta\|Y)$	$V(\beta\|Y)$	Importance Score
3 point Pass	0.50551	0.00017	38.32173
Set 3–5 feet off the net	0.51304	0.00018	37.88245
4 point Pass	0.51001	0.00020	36.51091
2 point Pass	0.48935	0.00019	35.78412
4 point Dig	0.43787	0.00016	34.67090
Set 5–8 feet off the net	0.49893	0.00025	31.60894
5 point Dig	0.50061	0.00025	31.58385
Left Attack	0.49665	0.00033	27.46854
Set 0–3 feet off the net	0.50669	0.00044	24.27541
Middle Attack	0.53806	0.00070	20.30614
Right Attack	0.55130	0.00101	17.35607
Set 8–10 feet off the net	0.42340	0.00066	16.50323
1 point Pass	0.36451	0.00054	15.67747
Overpass Attack	0.65062	0.00270	12.52568
3 point Float Serve	0.26774	0.00054	11.56483
3 point Jump Serve	0.18633	0.00040	9.35556
Back Attack	0.38659	0.00197	8.71921
2 point Dig	0.38268	0.00211	8.33366
Set Dump Attack	0.54814	0.00776	6.22122
3 point Dig	0.48367	0.00665	5.93223
2 point Float Serve	0.24707	0.00216	5.31146
1 point Float Serve	0.21983	0.00188	5.07389
2 point Jump Serve	0.16202	0.00122	4.64685
1 point Jump Serve	0.16242	0.00168	3.96645
Out of System Attack	0.26291	0.00974	2.66430

Table 3.4: Importance scores for the volleyball logistic regression analysis.

Skill	$E(\beta\|Y)$	$V(\beta\|Y)$	Importance Score
Pass	0.51946	0.00375	8.48683
Float Serve	0.81906	0.00992	8.22162
Jump Serve	0.74160	0.00949	7.61225
Set Distance	0.33156	0.00271	6.36639
Digs	0.51379	0.00951	5.26835

skill, and thus is related to the slope of the line being used to compute the importance score. Thus, serving importance is higher in the logistic regression analysis than it seems generally to be in the Markov chain analysis. Since a four-point serve is an ace, it cannot be included in the Markov analysis as the rating is exactly the same as the outcome. However, the four-point serve can be included in the logistic analysis, and this raises the importance score for serving. This points to the necessity of an appropriate rating system if importance scores are going to be compared across skills.

For this team, float serves are more important than jump serves using both methods. Float serves have better expected outcomes for all ratings, and higher importance despite occurring less often.

It is interesting that digging tends to mirror passing but at a lower importance level. This would indicate that it is more difficult to convert after a dig than after a pass. This outcome seems reasonable when one considers that after a pass, the offense is set up, while after a dig, the offense is usually scrambling to set up a play.

When this approach was used on data from the Men's National Volleyball Team, serving and attacking were the most important skills (Fellingham and Reese 2004). Setting was not measured in that data set, while digging had virtually no importance. The results from this division I women's team indicates that passing, setting, and digging all have relatively high importance. We believe this is a fundamental difference in the men's and women's games. The men hit the ball harder, meaning that rallies terminate sooner, and serving and attacking have greater importance. In the women's game, rallies are longer, so passing and digging increase in importance.

Based on these analyses, we would give the following recommendations to this team.

(1) Only serve a jump serve if the individual's float serve is clearly inferior.

(2) Keep sets and passes away from the net.

(3) Force the attack to the middle and right side if at all possible.

(4) Work on setting up a more diverse attack scheme off a dig.

We have shown two different methodologies to develop skill importance scores. These importance scores can be used by coaches to change team tactics, change skill performance goals, and focus practice time to increase the probability of scoring points.

BIBLIOGRAPHY FOR ARTICLE

Assoudou, S. and Essebbar, B. (2003), "A Bayesian Model for Markov Chains via Jeffrey's Prior," *Communications in Statistics*, 32, 2163–2184.

Coleman, J. (1975), "A statistical evaluation of selected volleyball techniques at the 1974 world's volleyball championships," Ph.D. thesis, Brigham Young University.

Fellingham, G. and Reese, C. (2004), "Rating Skills in International Men's Volleyball," Unpublished report to the USA National Men's Volleyball Team.

Florence, L. W. (2008), *Skill Evaluation in Women's Volleyball*, master's project, Brigham Young University.

Florence, L. W. and Fellingham, G. W. (2008), "Skill Evaluation in Women's Volleyball," *Journal of Quantitative Analysis in Sports*, 4, article 14.

Gelman, A., Carlin, J., Stern, H., and Rubin, D. (2004), *Bayesian Data Analysis*, New York: Chapman & Hall, 2nd ed.

Hughes, M. and Daniel, R. (2003), "Playing patterns of elite and non-elite volleyball," *International Journal of Performance Analysis in Sport*, 3, 50–56.

Lee, T. C., Judge, G. G., and Zellner, A. (1968), "Maximum Likelihood and Bayesian Estimation of Transition Probabilities," *Journal of the American Statistical Association*, 63, 1162–1179.

Leonard, T. (1972), "Bayesian Methods for Binomial Data," *Biometrika*, 59, 581–589.

Lindley, D. V. (1964), "The Bayesian Analysis of Contingency Tables," *The Annals of Mathematical Statistics*, 35, 1622–1643.

Yiannis, L. and Panagiotis, K. (2005), "Evolution in men's volleyball skills and tactics as evidenced in the Athens 2004 Olympic Games," *International Journal of Performance Analysis in Sport*, 5, 1–8.

Zetou, E., Moustakidis, A., Tsigilis, N., and Komninakidou, A. (2007), "Does Effectiveness of Skill in Complex I Predict Win in Men's Olympic Volleyball Games?" *Journal of Quantitative Analysis in Sports*, 3, article 3.

BIBLIOGRAPHY

Ali, A. H. (1988), "A statistical analysis of tactical movement patterns in soccer," in *Science and football*, eds. Reilly, T., Lees, A., Davids, K., and Murphy, W., London: Spon E & F N, pp. 302–308.

Assoudou, S. and Essebbar, B. (2003), "A Bayesian Model for Markov Chains via Jeffrey's Prior," *Communications in Statistics*, 32, 2163–2184.

Bate, R. (1988), "Football chance: Tactics and strategy," in *Science and football*, eds. Reilly, T., Lees, A., Davids, K., and Murphy, W., London: Spon E & F N, pp. 293–301.

Bayes, T. (1764), "Essay Towards Solving a Problem in the Doctrine of Chances," *Philosophical Transactions of the Royal Society of London*.

Brillinger, D. R. (2007), "Potential Function Approach to the Flow of Play in Soccer," *Journal of Quantitative Analysis in Sports*, 3, article 3.

Cargnoni, C., Muller, P., and West, M. (1997), "Bayesian Forecasting of Multinomial Time Series Through Conditionally Gaussian Dynamic Models," *Journal of the American Statistical Association*, 92, 640–647.

Chervenjakov, M. and Dimitrov, G. (1988), "Assessment of the playing effectiveness of soccer players," in *Science and football*, eds. Reilly, T., Lees, A., Davids, K., and Murphy, W., London: Spon E & F N, pp. 299–292.

Coleman, J. (1975), "A statistical evaluation of selected volleyball techniques at the 1974 world's volleyball championships," Ph.D. thesis, Brigham Young University.

Congdon, P. (2006), *Bayesian Statistical Modelling*, England: John Wiley & Sons, Inc, 2nd ed.

Degroot, M. (1970), *Optimal Statistical Decisions*, New York: McGraw-Hill Inc.

Fellingham, G. and Reese, C. (2004), "Rating Skills in International Men's Volleyball," Unpublished report to the USA National Men's Volleyball Team.

Florence, L. W. (2008), *Skill Evaluation in Women's Volleyball*, master's project, Brigham Young University.

Florence, L. W. and Fellingham, G. W. (2008), "Skill Evaluation in Women's Volleyball," *Journal of Quantitative Analysis in Sports*, 4, article 14.

Gelman, A., Carlin, J., Stern, H., and Rubin, D. (2004), *Bayesian Data Analysis*, New York: Chapman & Hall, 2nd ed.

Hughes, M. and Bartlett, R. (2002), "The use of performance indicators in performance analysis." *Journal of Sports Sciences*, 20, 739–754.

Hughes, M. and Daniel, R. (2003), "Playing patterns of elite and non-elite volleyball," *International Journal of Performance Analysis in Sport*, 3, 50–56.

Hughes, M. and Franks, I. (2005), "Analysis of passing sequences, shots and goals in soccer," *Journal of Sports Sciences*, 23, 509–514.

Lee, T. C., Judge, G. G., and Cain, R. L. (1969), "A Sampling Study of the Properties of Estimators of Transition Probabilities," *Management Science*, 15, 374–398.

Lee, T. C., Judge, G. G., and Takayama, A. (1965), "On Estimating the Transition Probabilities of a Markov Process," *Journal of Farm Economics*, 47, 742–762.

Lee, T. C., Judge, G. G., and Zellner, A. (1968), "Maximum Likelihood and Bayesian Estimation of Transition Probabilities," *Journal of the American Statistical Association*, 63, 1162–1179.

Leonard, T. (1972), "Bayesian Methods for Binomial Data," *Biometrika*, 59, 581–589.

Lindley, D. V. (1964), "The Bayesian Analysis of Contingency Tables," *The Annals of Mathematical Statistics*, 35, 1622–1643.

Mandansky, A. (1959), "Least Squares Estimation in Finite Markov Processes," *Psychometrika*, 24, 137–144.

McHale, I. and Scarf, P. (2007), "Modelling soccer matches using bivariate discrete distributions with general dependence structure," *Statistica Neerlandica*, 61, 432–445.

McKeigue, P. M., Carpenter, J. R., Parra, E. J., and Shriver, M. D. (2000), "Estimation of admixture and detection of linkage in admixed populations by a Bayesian approach: application to African-American populations," *Annals of Human Genetics*, 64, 171–186.

Meshkani, M. R. and Billard, L. (1992), "Empirical Bayes Estimators for a Finite Markov Chain," *Biometrika*, 79, 185–193.

Miller, G. A. (1952), "Finite Markov Processes in Psychology," *Psychometrika*, 17, 149–167.

Ozekici, S. and Soyer, R. (2003), "Network Reliability Assessment in a Random Environment," *Naval Research Logistics*, 50, 574–591.

Pollard, R. and Reep, C. (1997), "Measuring the effectiveness of playing strategies at soccer," *The Statistician*, 46, 541–550.

Pollard, R., Reep, C., and Hartley, S. (1988), "The quantitative comparison of playing styles in soccer," in *Science and football*, eds. Reilly, T., Lees, A., Davids, K., and Murphy, W., London: Spon E & F N, pp. 309–312.

Reep, C. and Benjamin, B. (1968), "Skill and chance in association football," *Journal of Royal Statistical Society*, 131, 581–585.

Ross, S. M. (1996), *Stochastic Processes*, New York: John Wiley & Sons, Inc, 2nd ed.

Stewart, W. J. (1994), *Introduction to the Numerical Solution of Markov Chains*, New Jersey: Princeton University Press.

Tesler, L. G. (1966), "Least Squares Estimates of Transition Probabilities," in *Measurement in Economics*, California: Stanford University Press, pp. 270–293.

Theil, H. and Rey, G. (1964), "A Quadratic Programming Approach to the Estimation of the Transition Probabilities," *International Center for Management Science*, report 6426.

Thomas, C. (2005), "Development of a Notational Analysis System for Selected Soccer Skills of a Women's College Team," Ph.D. thesis, Brigham Young University.

Van Calster, Ben; Smits, T. and Van Huffel, S. (2008), "The Curse of Scoreless Draws in Soccer: The Relationship with a Team's Offensive, Defensive, and Overall Performance," *Journal of Quantitative Analysis in Sports*, 4, article 4.

Yiannis, L. and Panagiotis, K. (2005), "Evolution in men's volleyball skills and tactics as evidenced in the Athens 2004 Olympic Games," *International Journal of Performance Analysis in Sport*, 5, 1–8.

Zetou, E., Moustakidis, A., Tsigilis, N., and Komninakidou, A. (2007), "Does Effectiveness of Skill in Complex I Predict Win in Men's Olympic Volleyball Games?" *Journal of Quantitative Analysis in Sports*, 3, article 3.

Zhao, J. X., Foulkes, A. S., and Georged, E. I. (2005), "Exploratory Bayesian Model Selection for Serial Genetics Data," *Biometrics*, 61, 591–599.

A. METHODS AND RESULTS FOR THE SOCCER ANALYSIS

A.1 Introduction

This portion of the project explores skill importance in women's soccer. For a particular season, four skills (passing, dribbling, defensive, and first touch) are individually analyzed. The skills are evaluated in order to determine which skills have the greatest effect on producing a successful outcome. The methodology used by Fellingham and Reese (2004) is used to calculate skill importance. The data set and methods used to calculate the posterior distributions are described below.

A.2 The Data

The data were collected during the 2005 competitive season of a women's division I soccer team. During each home game, the game was recorded, and the four skills (passing, dribbling, first touch, and defensive) were analyzed for the home and the opposing team. Using the performance scoring system developed by Coleman (1975) as a guide and the expertise of soccer coaches and researchers, each technique and tactic was rated. The panel of soccer coaches and researchers also noted if the tactic was part of a sequence of plays that led to a shot on goal or a goal. Defensive skills are rated on a nine-point (0–8) scale, passes on an eight-point (0–7) scale, dribbling skills on a five-point (0–4) scale, and first touch skills are rated on a six-point (0–5) scale.

Using the data collected from the women's soccer team during the competitive season, we determine how the performance ratings of each of the four techniques (passing, dribbling, first touch, and defensive) affect the probability of a successful shot for 11 different division I women's soccer teams. The 11 teams consist of the primary team plus the 10 opponents they played at home during the 2005 competitive

season. We refer to the 10 teams that the primary team played during the 2005 season as Opponent 1 through Opponent 10.

A.2.1 Bayesian Modeling

A Bayesian logistic regression model is implemented to determine how the performance of defensive, first touch, passing, and dribbling skills affects the probability of a successful shot. For skills related to a specific team, we model the response variable as

$$Y_i = \begin{cases} 0 & \text{if a shot was off target.} \\ 1 & \text{if a shot was on target or a goal was made.} \end{cases}$$

A shot is considered successful if it is on target—meaning the shot is attempted and blocked by the goalie—or if the shot results in a goal. Logistic regression is used to relate the quality of the skills leading up to a shot to the probability of a successful shot. Logistic regression uses the log odds ratio to relate the quality of skills being performed to the probability of a successful shot. For the j^{th} team, the log odds ratio is defined as

$$\log\left(\frac{Pr[Y_k = 1|\text{skill}{=}i]}{Pr[Y_k = 0|\text{skill}{=}i]}\right) = \beta_{0j} + \beta_{ij}\mathrm{R}_{ijk}, \tag{A.1}$$

where $i = 1, \ldots, 4$ is associated with a specific skill, $j = 1, \ldots, 11$ corresponds to a specific team, $k = 1, \ldots, n_j$ corresponds to the k^{th} touch for the j^{th} team, β_{0j} is team j's overall ability, β_{ij} is the effect of the i^{th} skill for the j^{th} team, and R_{ijk} is the rating associated with the k^{th} touch that corresponds to the i^{th} skill for the j^{th} team. Therefore, $Y_{ijk} \sim \textit{Bernoulli}\left(\frac{exp(\beta_{0j}+\beta_{ij}\mathrm{R}_{ijk})}{1+exp(\beta_{0j}+\beta_{ij}\mathrm{R}_{ijk})}\right)$.

A.2.2 Prior Specification

β_{ij} is the effect of performing skill i on the log odds ratio of taking a successful shot. For team j, β_{1j} is the effect of performing a defensive skill, β_{2j} is the effect of

performing a passing skill, β_{3j} is the effect of performing a first touch skill, and β_{4j} is the effect of performing a dribbling skill.

We expect that the β parameters associated with a specific skill are related to each other. For any team, we expect that the effect of performing a specific skill should come from the same overall distribution. Hierarchical modeling is used so each team's parameter associated with skill i comes from the same common population distribution. The following prior distribution is chosen for all 55 β_{ij} parameters:

$$f(\beta_{ij}|\mu_{\beta_i}, \sigma^2_{\beta_i}) \sim Normal(\mu_{\beta_i}, \sigma^2_{\beta_i}) \quad i = 0, \ldots, 4 \quad j = 1, \ldots, 11,$$

where i corresponds to the overall team ability ($i = 0$) or the skill being performed ($i = 1, \ldots, 4$), and j corresponds to the team.

The normal distribution is chosen as the prior for each of the β parameters because the effect of performing specific skills can be positive or negative and a team's overall ability can have a positive or negative effect on the probability of a successful shot. The priors of the hyperparameters are chosen as

$$f(\mu_{\beta_i}) \sim Normal(m_{\mu_{\beta_i}}, s^2_{\mu_{\beta_i}}) \qquad f(\sigma^2_{\beta_i}) \sim Inverse\ Gamma(a_{\beta_i}, b_{\beta_i}).$$

These prior distributions are chosen because they match the parameter space of μ_{β_i} and $\sigma^2_{\beta_i}$. The normal distribution is often used to model the mean of a distribution, and it is common to use the inverse gamma distribution to model the variance of a distribution. The normal and inverse gamma distributions are convenient priors when the parent distribution follows a normal distribution. This choice of prior distributions allows conjugacy to occur in some of the complete conditionals.

In order to determine the prior values for each of the hyperparameters, elicitation was conducted. Gilbert Fellingham, a statistics professor at BYU who has previously worked with the women's soccer team data, was consulted to determine the appropriate values for the hyperparameters. He recommended putting diffuse priors on all the parameters. The values for each of the means are centered about

zero because, for any team, performing a specific skill might not affect the probability of a successful shot. The hyperparameter values are shown in Table A.1, where $i = 0, \ldots, 4$ corresponds to each of the appropriate β parameters.

Table A.1: Hyperparameter value specification for the prior distributions.

Parameter	Hyperparameters	
μ_{β_i}	$m_{\mu_{\beta_i}} = 0$	$s^2_{\mu_{\beta_i}} = 1000$
$\sigma^2_{\beta_i}$	$a_{\beta_i} = 2.01$	$b_{\beta_i} = 0.001$

Because the posterior distribution cannot be written in closed form, Gibbs sampling, in conjunction with Metropolis-Hastings, is used to obtain draws from the posterior distribution. The convenient priors chosen previously are useful because they allow conjugacy to occur in some of the complete conditionals. When Metropolis-Hastings is used, mixing plots are used to ensure that sampling has occurred from all parts of the distribution. Using Gibbs sampling in conjunction with Metropolis-Hastings, 100,000 observations are drawn from the posterior distribution. Mixing plots are used to determine the appropriate burn-in period. The mixing plots indicate that a burn-in period of 200 observations is sufficient.

To determine how well the model fits the data, a Bayesian χ^2 goodness-of-fit test is constructed. To summarize all the Bayesian χ^2 tests, we calculated the average number of times the Bayesian χ^2 test resulted in a significant p-value. The Bayesian χ^2 test resulted in a significant p-value only 4.2% of the time. This implies that only 4.2% of the time is the model inappropriate. Since this value is relatively small, we will continue exploring the resulting posterior distribution.

A.3 Results

The draws from the posterior distribution are used to calculate importance scores for each team for all of the skills. Thus, for a specific team, the coefficient of

skill importance is defined as the ratio

$$I_i = \frac{E(\beta_i|Y)}{\sqrt{V(\beta_i|Y)}},$$

where i corresponds to the skill and β_i indicates the posterior distribution associated with skill i when modeling a successful shot, as specified by Fellingham and Reese (2004). For a given team, the importance scores for the soccer data are calculated by using the posterior distributions associated with the skills. The importance scores for the primary team and two of their opponents are shown in Table A.2.

Table A.2: Importance scores for the soccer data for the primary team and two of their opponents.

	Primary Team	Opponent 1	Opponent 7
Defensive	2.24	0.42	0.94
Dribbling	6.13	1.99	0.72
First Touch	6.45	1.40	0.73
Passing	7.10	0.90	1.36

A.4 Discussion

Table A.2 shows which skills are the most important to different teams. The skills that are the most important to the primary team in performing a successful shot are ranked in the following order: passing, first touch, dribbling, and defense. Passing and first touch skills for the primary team appear to have near equal importance, with dribbling skills following closely behind. Opponent 1's importance scores suggest that dribbling is the most important skill followed by first touch, passing, and lastly defensive skills. Opponent 7's important scores indicate that passing is slightly more important than the other skills, but the remainder of the skills have equal importance.

Importance scores can help coaches to allocate practice time more efficiently in order to maximize team performance. Coaches can then practice the skills that contribute the most to a successful outcome. The primary team should focus on

practicing passing, first touch, and then dribbling skills whereas Opponent 7's coach should focus more on passing and then equally on all the other skills.

Importance scores can also be used to compare a team's performance to another team. During the 2005 season, the primary team only lost two games. It would make sense to compare other opponents to the primary team to see how their importance scores rank in comparison to the primary team. Thus, if Opponent 7 wanted to mimic the primary team's strategy, Opponent 7 should focus on improving their passing and first touch skills during their next practice.

This project shows how importance scores can be used to help a coach allocate practice time. It shows that skill importance in soccer can be determined through the use of importance scores. Importance scores enable a coach to see how the performance of skills affects the probability of a successful shot. Importance scores also allow a coach to directly compare the performance of their team to another team in order to identify which skills their team needs to practice.

B. COMPUTER CODE

The following is the computer code written for this project. Section B.1 displays the R code to compute the posterior distributions for the volleyball Markov chain model. Section B.2 gives the C Code used to compute the posterior distributions for the volleyball logistic regression model. Lastly, section B.3 shows the C code used to compute the posterior distributions for the soccer hierarchical logistic regression model.

B.1 Volleyball Markov Chain

```
#################################################################
##  Clean Data for BYU Women's Volleyball Team Analysis       ##
#################################################################

# Read in the current file with all 13 games combined into one file
vb <- read.table("Data/Combined New Data.txt",sep=";",
comment.char="@")

#Gives names to the first three columns in the data frame
names(vb) <- c("play", "opponent", "rotation")

#Disregard computer code
vb <-vb[substr(as.character(vb$play),3,3)!="&",]
#Disregard home setters and home scores
vb <-vb[substr(as.character(vb$play),2,2)!="P",]
#Disregard opponent setters and opponent scores
vb <-vb[substr(as.character(vb$play),2,2)!="z",]

#Separates out the player #'s (Will have NAs for scores)
vb$players <- as.numeric(substr(as.character(vb$play),1,2))
vb$skill <- substr(as.character(vb$play),3,4) #Separate out skill
vb$score <- substr(as.character(vb$play),5,5) #Separate out score

skillscore <- substr(as.character(vb$play),3,5)

# Loop through the data and look for when each game is over
#      (**1set, **2set,**3set,**4set)
team <- rep(NA,length(vb$players))
```

```
outcome <- rep(NA,length(vb$players))
for(i in 1:length(vb$play)){
    if(substr(as.character(vb$play[i]),1,2)=="**"){
        outcome[i] <- "GAMEOVER"
        if(substr(as.character(vb$play[i-1]),1,2)=="*p"){
            j<-2
            while(1){
                if(vb$players[i-j]<50 ||
                  substr(as.character(vb$skill[i-j]),1,1)=="S")
                  {break}
                j <- j+1}
            outcome[i-j] <- "Good"
        }
        else if(substr(as.character(vb$play[i-1]),1,2)=="ap"){
            j<-2
            while(1){
                if(vb$players[i-j]<50 ||
                   substr(as.character(vb$skill[i-j]),1,1)=="S")
                {break}
                j <- j+1}
            outcome[i-j] <- "Bad"
        }
    }
}

vb$outcome <- outcome

#Disregard opponent scores
vb <-vb[substr(as.character(vb$play),1,2)!="ap",]
#Disregard home scores
vb <-vb[substr(as.character(vb$play),1,2)!="*p",]

### Goes through a loop and indicates when there is a new serve
#   and which hits are by the BYU/opp team
for(i in 1:length(vb$players)){
    if(substr(as.character(vb$play[i]),1,2)=="**"){
         vb$team[i] <- "GAMEOVER"}
    else if(vb$players[i] > 50){
        #Signifies when opponent serves
        if(substr(vb$skill[i],1,1)=="S") vb$team[i] <- "OPPSERVE"
        else vb$team[i] <- "OPP"   #Signifies when opponent hits
    }
    else if(vb$players[i] < 50 && substr(vb$skill[i],1,1)=="S"){
        vb$team[i] <- "HOMESERVE"}    #Signifies when home serves
    else {vb$team[i] <- "HOME"}        #Signifies when home hits
```

```
}

##
###Identify the outcomes (Good, Bad, Continue)
##
for(i in 1:length(vb$team)){
  if(is.na(vb$outcome[i])){
    if(vb$team[i]=="HOME"){
        if(vb$team[i+1]=="HOME" && substr(vb$skill[i],1,1)=="A"){
    vb$outcome[i] <- "Continue"}
        else if(vb$team[i+1]=="HOMESERVE")    vb$outcome[i] <- "Good"
        else if(vb$team[i+1]=="OPPSERVE") vb$outcome[i] <- "Bad"
        else if(vb$team[i+1]=="OPP"){
            ###  Determines if the ball ever returns to BYU team.
            #     If not, then outcome is recorded
            j <- 0
            while(1)    {
                if(vb$team[i+2+j]=="HOME") {
                    #If the play goes back to Home team,
                    # then it was a continued rally
                    vb$outcome[i]<-"Continue"
                    break }        #Break gets out of the loop
                else if(vb$team[i+2+j]=="HOMESERVE") {
                    #Ball never came back to Home side of net.
                    vb$outcome[i] <- "Good"
                    break }
                else if(vb$team[i+2+j]=="OPPSERVE") {
                    vb$outcome[i] <- "Bad"
                    break }
                else {j <- j+1}
            }
        }
        else {vb$outcome[i] <- "NA"}
    }
    else if(vb$team[i]=="HOMESERVE"){
        if(vb$team[i+1]=="HOMESERVE") vb$outcome[i] <- "Good"
        else if(vb$team[i+1]=="OPPSERVE") vb$outcome[i] <- "Bad"
        else if(vb$team[i+1]=="OPP"){
            ###  Determines if the ball ever returns to BYU team.
            ##     If not, then outcome is recorded
            j <- 0
            while(1)    {
                if(vb$team[i+2+j]=="HOME") {
                    # If the play goes back to Home team,
                    #  then it was a continued rally
```

```
                    vb$outcome[i]<-"Continue"
                    break }    #Break gets out of the loop
                else if(vb$team[i+2+j]=="HOMESERVE") {
                    #Ball never came back to Home side of net.
                    vb$outcome[i] <- "Good"
                    break }
                else if(vb$team[i+2+j]=="OPPSERVE") {
                    vb$outcome[i] <- "Bad"
                    break }
                else {j <- j+1}
            }
        }
        else {vb$outcome[i] <- "NA"}
    }
    #If Opponent Serves
    else if(vb$team[i]=="OPPSERVE"){
        if(vb$team[i+1]=="HOMESERVE") vb$outcome[i] <- "Good"
        else if(vb$team[i+1]=="OPPSERVE") vb$outcome[i] <- "Bad"
        else {vb$outcome[i] <- "NA"}
    }
    else {vb$outcome[i] <- "NA"}
  }
}

##
### Change the opponent Serve from "SQ" and "SH" to "OQ" and "OH" ###
### This allows us to distinguish between Home and Opponent Serves
##
for (i in 1:length(vb$players)){
    if(vb$players[i] > 50 && substr(vb$skill[i],1,1)=="S")
        vb$skill[i] <- paste("O",substr(vb$skill[i],2,2),sep="")
}

#We only care about opponents as "float" and "jump" serves
vb$score[substr(vb$skill,1,1)=="O"] <- "#"

##
###Replaces the skill "Attack" with the actual attacking codes:
##
vb$skill[substr(vb$skill,1,1)=="A"] <-
  substr(as.character(vb$play),6,7)[substr(as.character(vb$skill),
  1,1)=="A"]
vb <-vb[vb$team!="OPP",]         #Disregard opponent hits

# Combine the skill and score together
```

```
vb$skillscore <- paste(vb$skill,vb$score,sep="")

#This makes it so I don't have to keep running the previous code if
#  I just want to look at something in the dataset
save(vb,file="volleyclean.txt")
load("R Code Cleaned Up/volleyclean.txt")

# This creates one long sequence of hits and outcomes ready to analyze
transitions <- NA
for(i in 1:length(vb$play)){
    if(vb$outcome[i]=="GAMEOVER"){transitions <-
         rbind(transitions, "GAMEOVER")}
    else if(vb$outcome[i]=="NA")
        #If no outcome, just put in the skill/score
           {transitions <- rbind(transitions, vb$skillscore[i])}
    else         #This is anything that has an outcome
        #Have skill/score first, then the outcome
        transitions <- rbind(transitions,vb$skillscore[i],vb$outcome[i])
}

#Write the game to a file
write(t(transitions[-1]), "transitions.txt",ncol=1,sep = "\t")
```

B.1.1 Create a Count Matrix

```
transitions <- as.matrix(read.table("R Code Cleaned Up/transitions.txt",
    comment.char=""))  #Read in the game

## Defines the names of all the different hits possible
## Will be used in the transition matrix
hits <- c("OH#","OQ#",
"SH#","SH/","SH+","SH!","SH-","SH=","SQ#","SQ/","SQ+","SQ!","SQ-",
"SQ=","RH#","RH+","RH!","RH-","RH=","RH/","RQ#","RQ+","RQ!","RQ-",
"RQ=","RQ/","EQ#","EQ+","EQ!","EH#","EH+","EH!","EH-","EH/","EH=",
"ET#","ET+","ET!","ET-","ET/","ET=","EM#","EM+","EM!","EM-","EM/",
"EM=","EL#","EL+","EL!","EL-","EL/","EL=","E","P2#","P2+","P2=",
"P2/","P3#","P3+","P3=","P3/","P5#","P5+","P6#","P6+","P6/","P8#",
"P8+","P8=","P8/","PA#","PA+","PA=","PB#","PB+","PB=","PB/","PD#",
"PD+","PD=","PD/","PG#","PG+","PG=","PG/","PH#","PH+","PH=","PH/",
"PK#","PK+","PK=","PK/","PM#","PM+","PM=","PM/","PO#","PO+","PO=",
"PP#","PP+","PP=","PR#","PR+","PS#","PS+","PS=","PS/","PW#","PW+",
"PW=","PW/","PX#","PX+","PX=","PX/","DH#","DH+","DH!","DH-","DH/",
"DH=","Good","Continue","Bad")

#To do the same thing do
```

```
unique(transitions) #Take out GAMEOVER and you get the same results

#####################################################################
#Create the count matrix from from the list of touches and outcomes#
#####################################################################

# Function to calculate the actual counts from the data
#   for every transition in the matrix
#Row is where it started
#Column is where it transitioned to

counts <- function(transitions){
    c.mat <- as.data.frame(matrix(0,length(hits),length(hits)),
        row.names=hits)
    names(c.mat) <- hits    #Name the columns of the data frame
    for(i in 1:(length(transitions)-1) ){
        if(transitions[i]=="GAMEOVER" ||
            transitions[i+1]=="GAMEOVER"){temp<-NA}
        else c.mat[transitions[i], transitions[i+1]] <-
            c.mat[transitions[i], transitions[i+1]] + 1
    }
    return(c.mat)
}

c.mat <- counts(transitions)

# Constrain some of the counts to be zero (data typos):
c.mat["RH#","Continue"] <- 0  #Perfect Pass
c.mat["RQ#","Good"] <- 0
c.mat["RH+","Good"] <- 0    #3 Pt Pass
c.mat["RQ+","Good"] <- 0
c.mat["RQ+","Continue"] <- 0
c.mat["RH!","Good"] <- 0    #2 Pt Pass
c.mat["RH!","Bad"] <- 0
c.mat["RQ!","Continue"] <- 0
c.mat["RQ!","Bad"] <- 0
c.mat["RH/","Good"] <- 0
c.mat["RQ-","DH-"] <- 0

# Write the game to a file:
write(t(c.mat), "cmat.txt",ncol=ncol(c.mat),sep = "\t")

#Need to constrain the same counts in the prior transition
#    count matrix to be zero:
#Prior Non-Informative
```

```
a.mat<-matrix(1,nrow=nrow(c.mat),ncol=ncol(c.mat))
a.mat <- as.data.frame(a.mat,row.names=hits)
names(a.mat) <- hits          #Names the columns of the data frame
a.mat["RH#","Continue"] <- 0  #Perfect Pass
a.mat["RQ#","Good"] <- 0
a.mat["RH+","Good"] <- 0    #3 Pt Pass
a.mat["RQ+","Good"] <- 0
a.mat["RQ+","Continue"] <- 0
a.mat["RH!","Good"] <- 0    #2 Pt Pass
a.mat["RH!","Bad"] <- 0
a.mat["RQ!","Continue"] <- 0
a.mat["RQ!","Bad"] <- 0
a.mat["RH/","Good"] <- 0
a.mat["RQ-","DH-"] <- 0

write(t(a.mat), "amat.txt",ncol=ncol(a.mat),sep = "\t")
```

B.1.2 Collapsing Count Matrix by Set Distance

```
## Defines the names of all the different hits possible -
#Will be used in the transition matrix
hits <- c("OH#","OQ#",
"SH#","SH/","SH+","SH!","SH-","SH=","SQ#","SQ/","SQ+","SQ!","SQ-",
"SQ=","RH#","RH+","RH!","RH-","RH=","RH/","RQ#","RQ+","RQ!","RQ-",
"RQ=","RQ/","EQ#","EQ+","EQ!","EH#","EH+","EH!","EH-","EH/","EH=",
"ET#","ET+","ET!","ET-","ET/","ET=","EM#","EM+","EM!","EM-","EM/",
"EM=","EL#","EL+","EL!","EL-","EL/","EL=","E","P2#","P2+","P2=",
"P2/","P3#","P3+","P3=","P3/","P5#","P5+","P6#","P6+","P6/","P8#",
"P8+","P8=","P8/","PA#","PA+","PA=","PB#","PB+","PB=","PB/","PD#",
"PD+","PD=","PD/","PG#","PG+","PG=","PG/","PH#","PH+","PH=","PH/",
"PK#","PK+","PK=","PK/","PM#","PM+","PM=","PM/","PO#","PO+","PO=",
"PP#","PP+","PP=","PR#","PR+","PS#","PS+","PS=","PS/","PW#","PW+",
"PW=","PW/","PX#","PX+","PX=","PX/","DH#","DH+","DH!","DH-","DH/",
"DH=","Good","Continue","Bad")

#############################################################
## Read in the counts matrix
c.mat <- read.table("Full Matrix/cmat.txt", comment.char="")
#Read in the prior counts
c.mat <- as.data.frame(c.mat,row.names=hits)
names(c.mat) <- hits      #Names the columns of the data frame

### Collapse the count matrix:
newcmat <- c.mat["OQ#",] + c.mat["OH#",]
```

```
#Move over the Float Serves for the Primary Team
newcmat["Float 5-pt",] <- c.mat["SH#",]
newcmat["Float 4-pt",] <- c.mat["SH/",]
newcmat["Float 3-pt",] <- c.mat["SH+",]
newcmat["Float 2-pt",] <-c.mat["SH!",]
newcmat["Float 1-pt",] <-c.mat["SH-",]
newcmat["Float 0-pt",] <- c.mat["SH=",]

newcmat["Jump 5-pt",] <- c.mat["SQ#",]
newcmat["Jump 4-pt",] <- c.mat["SQ/",]
newcmat["Jump 3-pt",] <- c.mat["SQ+",]
newcmat["Jump 2-pt",] <-c.mat["SQ!",]
newcmat["Jump 1-pt",] <-c.mat["SQ-",]
newcmat["Jump 0-pt",] <- c.mat["SQ=",]


#This combines the passes received from float and jump serves
newcmat["4pt",] <- c.mat["RQ#",] + c.mat["RH#",]
newcmat["3pt",] <- c.mat["RQ+",] + c.mat["RH+",]
newcmat["2pt",] <- c.mat["RQ!",] + c.mat["RH!",]
newcmat["1pt",] <- c.mat["RQ-",] + c.mat["RH-",]
newcmat["0pt",] <- c.mat["RQ=",] + c.mat["RH=",]
newcmat["PassOverpass",] <- c.mat["RQ/",] + c.mat["RH/",]

#Identifying Set Distance0
#Combining 0-1 feet from net and 1-3 feet from net
#There were only 4 hits total 0-1 feet from net
newcmat["0to3ft",] <- c.mat["EQ#",] + c.mat["EQ+",] + c.mat["EQ!",] +
c.mat["EH#",] + c.mat["EH+",] + c.mat["EH!",] +
 c.mat["EH-",] + c.mat["EH/",] + c.mat["EH=",]
newcmat["3to5ft",] <- c.mat["ET#",] + c.mat["ET+",] + c.mat["ET!",] +
c.mat["ET-",] + c.mat["ET/",] + c.mat["ET=",]
newcmat["5to8ft",] <- c.mat["EM#",] + c.mat["EM+",] + c.mat["EM!",] +
c.mat["EM-",] + c.mat["EM/",] + c.mat["EM=",]
newcmat["8to10ft",] <- c.mat["EL#",] + c.mat["EL+",] + c.mat["EL!",] +
c.mat["EL-",] + c.mat["EL/",] + c.mat["EL=",]
newcmat["NotSetter",] <- c.mat["E",]

#> P2  Front 2  -  middle
#> P3  Gap Set  - middle
#> P5  High set to RS  - right
#> P6  Back 1 -- middle
#> P8  Fast Slide -- middle
#> PA  Out of system front row attack  - Separate Category
#> PB  Back row B set  -- back row
```

```
#> PD  Back row right side "D"  - -  back
#> PG  Go  -- left
#> PH  Hut  -- left
#> PK  Right Side "Red" -- right
#> PM  Highball "4"  -- left
#> PO  Overpass Attack  -- Separate Category
#> PP  Pipe or BIC  -- back
#> PR  Inside left side set "Rip" -- left
#> PS  Setter Dump  -- Separate Category
#> PW  Slide -- middle
#> PX  "X-series" or Combo  -- right

#Identify the attacks:
newcmat["Middle",] <-
    c.mat["P2#",] + c.mat["P2+",] + c.mat["P2=",] + c.mat["P2/",] +
    c.mat["P3#",] + c.mat["P3+",] + c.mat["P3=",] + c.mat["P3/",] +
    c.mat["P6#",] + c.mat["P6+",] + c.mat["P6/",] +
    c.mat["P8#",] + c.mat["P8+",] + c.mat["P8=",] + c.mat["P8/",] +
    c.mat["PW#",] + c.mat["PW+",] + c.mat["PW=",] + c.mat["PW/",]

newcmat["Right",]  <-  c.mat["P5#",] + c.mat["P5+",] +
    c.mat["PK#",] + c.mat["PK+",] + c.mat["PK=",] + c.mat["PK/",] +
    c.mat["PX#",] + c.mat["PX+",] + c.mat["PX=",] + c.mat["PX/",]

newcmat["Left",]   <-
    c.mat["PG#",] + c.mat["PG+",] + c.mat["PG=",] + c.mat["PG/",] +
    c.mat["PH#",] + c.mat["PH+",] + c.mat["PH=",] + c.mat["PH/",] +
    c.mat["PM#",] + c.mat["PM+",] + c.mat["PM=",] + c.mat["PM/",] +
    c.mat["PR#",] + c.mat["PR+",]

newcmat["Back",]   <-
    c.mat["PB#",] + c.mat["PB+",] + c.mat["PB=",] + c.mat["PB/",] +
    c.mat["PD#",] + c.mat["PD+",] + c.mat["PD=",] + c.mat["PD/",] +
    c.mat["PP#",] + c.mat["PP+",] + c.mat["PP=",]

newcmat["SetDump",]<-
    c.mat["PS#",] + c.mat["PS+",] + c.mat["PS=",] + c.mat["PS/",]
newcmat["OutSystem",]<-c.mat["PA#",] + c.mat["PA+",] + c.mat["PA=",]
newcmat["Overpass",]<-c.mat["PO#",] + c.mat["PO+",] + c.mat["PO=",]

#Dig Scores
newcmat["Dig #",] <- c.mat["DH#",]
newcmat["Dig +",] <- c.mat["DH+",]
newcmat["Dig !",] <- c.mat["DH!",]
newcmat["Dig -",] <- c.mat["DH-",]
```

```
newcmat["Dig /",] <- c.mat["DH/",]
newcmat["Dig =",] <- c.mat["DH=",]

#Move over the outcome for the primary team
newcmat["Good",] <- c.mat["Good",]
newcmat["Continue",] <- c.mat["Continue",]
newcmat["Bad",] <- c.mat["Bad",]

#####################################################
### Do the column collapsing
#####################################################
newhits <- c("O", "Float 5-pt", "Float 4-pt", "Float 3-pt", "Float
2-pt","Float 1-pt", "Float 0-pt", "Jump 5-pt", "Jump 4-pt", "Jump
3-pt", "Jump 2-pt","Jump 1-pt", "Jump 0-pt","4pt",  "3pt",  "2pt",
"1pt",  "0pt",  "PassOverpass","0to3ft", "3to5ft", "5to8ft",
"8to10ft", "NotSetter", "Middle", "Right", "Left", "Back", "SetDump",
 "OutSystem", "Overpass","Dig #","Dig +","Dig !","Dig -","Dig /","Dig
=",  "Good",  "Continue", "Bad")

cmat2 <- as.data.frame(newcmat[,"OQ#"] + newcmat[,"OH#"],
row.names=newhits)
names(cmat2) <- "O"

cmat2[,"Float 5-pt"] <- newcmat[,"SH#"]
cmat2[,"Float 4-pt"] <- newcmat[,"SH/"]
cmat2[,"Float 3-pt"] <- newcmat[,"SH+"]
cmat2[,"Float 2-pt"]<-newcmat[,"SH!"]
cmat2[,"Float 1-pt"]<-newcmat[,"SH-"]
cmat2[,"Float 0-pt"] <- newcmat[,"SH="]

cmat2[,"Jump 5-pt"] <- newcmat[,"SQ#"]
cmat2[,"Jump 4-pt"] <- newcmat[,"SQ/"]
cmat2[,"Jump 3-pt"] <- newcmat[,"SQ+"]
cmat2[,"Jump 2-pt"]<-newcmat[,"SQ!"]
cmat2[,"Jump 1-pt"]<-newcmat[,"SQ-"]
cmat2[,"Jump 0-pt"] <- newcmat[,"SQ="]

cmat2[,"4pt"] <- newcmat[,"RQ#"] + newcmat[,"RH#"]
cmat2[,"3pt"] <- newcmat[,"RQ+"] + newcmat[,"RH+"]
cmat2[,"2pt"] <- newcmat[,"RQ!"] + newcmat[,"RH!"]
cmat2[,"1pt"] <- newcmat[,"RQ-"] + newcmat[,"RH-"]
cmat2[,"0pt"] <- newcmat[,"RQ="] + newcmat[,"RH="]
cmat2[,"PassOverpass"] <- newcmat[,"RQ/"] + newcmat[,"RH/"]

cmat2[,"0to3ft"] <-
```

```
    newcmat[,"EQ#"] + newcmat[,"EQ+"] + newcmat[,"EQ!"] +
    newcmat[,"EH#"] + newcmat[,"EH+"] + newcmat[,"EH!"] +
    newcmat[,"EH-"] + newcmat[,"EH/"] + newcmat[,"EH="]
cmat2[,"3to5ft"] <-
    newcmat[,"ET#"] + newcmat[,"ET+"] + newcmat[,"ET!"] +
    newcmat[,"ET-"] + newcmat[,"ET/"] + newcmat[,"ET="]
cmat2[,"5to8ft"] <-
    newcmat[,"EM#"] + newcmat[,"EM+"] + newcmat[,"EM!"] +
    newcmat[,"EM-"] + newcmat[,"EM/"] + newcmat[,"EM="]
cmat2[,"8to10ft"] <-
    newcmat[,"EL#"] + newcmat[,"EL+"] + newcmat[,"EL!"] +
    newcmat[,"EL-"] + newcmat[,"EL/"] + newcmat[,"EL="]
cmat2[,"NotSetter"] <- newcmat[,"E"]

#Identify the attacks:
cmat2[,"Middle"] <-
    newcmat[,"P2#"] + newcmat[,"P2+"] + newcmat[,"P2="] +
    newcmat[,"P2/"] +
    newcmat[,"P3#"] + newcmat[,"P3+"] + newcmat[,"P3="] +
    newcmat[,"P3/"] +
    newcmat[,"P6#"] + newcmat[,"P6+"] + newcmat[,"P6/"] +
    newcmat[,"P8#"] + newcmat[,"P8+"] + newcmat[,"P8="] +
    newcmat[,"P8/"] +
    newcmat[,"PW#"] + newcmat[,"PW+"] + newcmat[,"PW="] +
    newcmat[,"PW/"]

cmat2[,"Right"]  <-
    newcmat[,"P5#"] + newcmat[,"P5+"] +
    newcmat[,"PK#"] + newcmat[,"PK+"] + newcmat[,"PK="] +
    newcmat[,"PK/"] +
    newcmat[,"PX#"] + newcmat[,"PX+"] + newcmat[,"PX="] +
    newcmat[,"PX/"]

cmat2[,"Left"]   <-
    newcmat[,"PG#"] + newcmat[,"PG+"] + newcmat[,"PG="] +
    newcmat[,"PG/"] +
    newcmat[,"PH#"] + newcmat[,"PH+"] + newcmat[,"PH="] +
    newcmat[,"PH/"] +
    newcmat[,"PM#"] + newcmat[,"PM+"] + newcmat[,"PM="] +
    newcmat[,"PM/"] +
    newcmat[,"PR#"] + newcmat[,"PR+"]

cmat2[,"Back"]   <-
    newcmat[,"PB#"] + newcmat[,"PB+"] + newcmat[,"PB="] +
    newcmat[,"PB/"] +
```

```
    newcmat[,"PD#"] + newcmat[,"PD+"] + newcmat[,"PD="] +
    newcmat[,"PD/"] +
    newcmat[,"PP#"] + newcmat[,"PP+"] + newcmat[,"PP="]

cmat2[,"SetDump"]<-newcmat[,"PS#"] +
    newcmat[,"PS+"] + newcmat[,"PS="] + newcmat[,"PS/"]
cmat2[,"OutSystem"]<-newcmat[,"PA#"] + newcmat[,"PA+"] +
    newcmat[,"PA="]
cmat2[,"Overpass"]<-newcmat[,"PO#"] + newcmat[,"PO+"] +
    newcmat[,"PO="]

cmat2[,"Dig #"] <- newcmat[,"DH#"]
cmat2[,"Dig +"] <- newcmat[,"DH+"]
cmat2[,"Dig !"] <- newcmat[,"DH!"]
cmat2[,"Dig -"] <- newcmat[,"DH-"]
cmat2[,"Dig /"] <- newcmat[,"DH/"]
cmat2[,"Dig ="] <- newcmat[,"DH="]

cmat2[,"Good"] <- newcmat[,"Good"]
cmat2[,"Continue"] <- newcmat[,"Continue"]
cmat2[,"Bad"] <- newcmat[,"Bad"]

c.mat <- cmat2

#write(t(cmat2), "collapsedcmat.txt",ncol=ncol(cmat2),sep = "\t")
#Write the game to a file
save(c.mat, file="collapsedcmatR.txt")

################################################################
###  Collapse A matrix of prior counts
################################################################
##
### Read prior counts into "a.mat" matrix
##

a.mat <- read.table("Full Matrix/amat.txt", comment.char="")
#Read in the prior counts
a.mat <- as.data.frame(a.mat,row.names=hits)
names(a.mat) <- hits   #Names the columns of the data frame

### Collapse the count matrix:


newamat["Float 5-pt",] <- a.mat["SH#",]
newamat["Float 4-pt",] <- a.mat["SH/",]
```

```
newamat["Float 3-pt",] <- a.mat["SH+",]
newamat["Float 2-pt",] <-a.mat["SH!",]
newamat["Float 1-pt",] <-a.mat["SH-",]
newamat["Float 0-pt",] <- a.mat["SH=",]
newamat["Jump 5-pt",] <- a.mat["SQ#",]
newamat["Jump 4-pt",] <- a.mat["SQ/",]
newamat["Jump 3-pt",] <- a.mat["SQ+",]
newamat["Jump 2-pt",] <-a.mat["SQ!",]
newamat["Jump 1-pt",] <-a.mat["SQ-",]
newamat["Jump 0-pt",] <- a.mat["SQ=",]

newamat["4pt",] <- a.mat["RQ#",] + a.mat["RH#",]
newamat["3pt",] <- a.mat["RQ+",] + a.mat["RH+",]
newamat["2pt",] <- a.mat["RQ!",] + a.mat["RH!",]
newamat["1pt",] <- a.mat["RQ-",] + a.mat["RH-",]
newamat["0pt",] <- a.mat["RQ=",] + a.mat["RH=",]
newamat["PassOverpass",] <- a.mat["RQ/",] + a.mat["RH/",]

newamat["0to3ft",] <-
    a.mat["EQ#",] + a.mat["EQ+",] + a.mat["EQ!",] +
    a.mat["EH#",] + a.mat["EH+",] + a.mat["EH!",] +
    a.mat["EH-",] + a.mat["EH/",] + a.mat["EH=",]

newamat["3to5ft",] <-
    a.mat["ET#",] + a.mat["ET+",] + a.mat["ET!",] +
    a.mat["ET-",] + a.mat["ET/",] + a.mat["ET=",]

newamat["5to8ft",] <-
    a.mat["EM#",] + a.mat["EM+",] + a.mat["EM!",] +
    a.mat["EM-",] + a.mat["EM/",] + a.mat["EM=",]

newamat["8to10ft",] <-
    a.mat["EL#",] + a.mat["EL+",] + a.mat["EL!",] +
    a.mat["EL-",] + a.mat["EL/",] + a.mat["EL=",]

newamat["NotSetter",] <- a.mat["E",]

newamat["Middle",] <-
    a.mat["P2#",] + a.mat["P2+",] + a.mat["P2=",] + a.mat["P2/",] +
    a.mat["P3#",] + a.mat["P3+",] + a.mat["P3=",] + a.mat["P3/",] +
    a.mat["P6#",] + a.mat["P6+",] + a.mat["P6/",] +
    a.mat["P8#",] + a.mat["P8+",] + a.mat["P8=",] + a.mat["P8/",] +
    a.mat["PW#",] + a.mat["PW+",] + a.mat["PW=",] + a.mat["PW/",]

newamat["Right",]  <-  a.mat["P5#",] + a.mat["P5+",] +
```

```
    a.mat["PK#",] + a.mat["PK+",] + a.mat["PK=",] + a.mat["PK/",] +
    a.mat["PX#",] + a.mat["PX+",] + a.mat["PX=",] + a.mat["PX/",]

newamat["Left",]   <-
    a.mat["PG#",] + a.mat["PG+",] + a.mat["PG=",] + a.mat["PG/",] +
    a.mat["PH#",] + a.mat["PH+",] + a.mat["PH=",] + a.mat["PH/",] +
    a.mat["PM#",] + a.mat["PM+",] + a.mat["PM=",] + a.mat["PM/",] +
    a.mat["PR#",] + a.mat["PR+",]

newamat["Back",]   <-
    a.mat["PB#",] + a.mat["PB+",] + a.mat["PB=",] + a.mat["PB/",] +
    a.mat["PD#",] + a.mat["PD+",] + a.mat["PD=",] + a.mat["PD/",] +
    a.mat["PP#",] + a.mat["PP+",] + a.mat["PP=",]

newamat["SetDump",]<-
    a.mat["PS#",] + a.mat["PS+",] + a.mat["PS=",] + a.mat["PS/",]
newamat["OutSystem",]<-a.mat["PA#",] + a.mat["PA+",] + a.mat["PA=",]
newamat["Overpass",]<-a.mat["PO#",] + a.mat["PO+",] + a.mat["PO=",]

newamat["Dig #",] <- a.mat["DH#",]
newamat["Dig +",] <- a.mat["DH+",]
newamat["Dig !",] <- a.mat["DH!",]
newamat["Dig -",] <- a.mat["DH-",]
newamat["Dig /",] <- a.mat["DH/",]
newamat["Dig =",] <- a.mat["DH=",]

newamat["Good",] <- a.mat["Good",]
newamat["Continue",] <- a.mat["Continue",]
newamat["Bad",] <- a.mat["Bad",]

#####################################################
### Do the column collapsing
#####################################################

amat2 <- as.data.frame(newamat[,"OQ#"] + newamat[,"OH#"],
    row.names=newhits)
names(amat2) <- "O"

amat2[,"Float 5-pt"] <- newamat[,"SH#"]
amat2[,"Float 4-pt"] <- newamat[,"SH/"]
amat2[,"Float 3-pt"] <- newamat[,"SH+"]
amat2[,"Float 2-pt"]<-newamat[,"SH!"]
amat2[,"Float 1-pt"]<-newamat[,"SH-"]
amat2[,"Float 0-pt"] <- newamat[,"SH="]
amat2[,"Jump 5-pt"] <- newamat[,"SQ#"]
```

```
amat2[,"Jump 4-pt"] <- newamat[,"SQ/"]
amat2[,"Jump 3-pt"] <- newamat[,"SQ+"]
amat2[,"Jump 2-pt"]<-newamat[,"SQ!"]
amat2[,"Jump 1-pt"]<-newamat[,"SQ-"]
amat2[,"Jump 0-pt"] <- newamat[,"SQ="]

amat2[,"4pt"] <- newamat[,"RQ#"] + newamat[,"RH#"]
amat2[,"3pt"] <- newamat[,"RQ+"] + newamat[,"RH+"]
amat2[,"2pt"] <- newamat[,"RQ!"] + newamat[,"RH!"]
amat2[,"1pt"] <- newamat[,"RQ-"] + newamat[,"RH-"]
amat2[,"0pt"] <- newamat[,"RQ="] + newamat[,"RH="]
amat2[,"PassOverpass"] <- newamat[,"RQ/"] + newamat[,"RH/"]

amat2[,"0to3ft"] <-
    newamat[,"EQ#"] + newamat[,"EQ+"] + newamat[,"EQ!"] +
    newamat[,"EH#"] + newamat[,"EH+"] + newamat[,"EH!"] +
    newamat[,"EH-"] + newamat[,"EH/"] + newamat[,"EH="]

amat2[,"3to5ft"] <-
    newamat[,"ET#"] + newamat[,"ET+"] + newamat[,"ET!"] +
    newamat[,"ET-"] + newamat[,"ET/"] + newamat[,"ET="]

amat2[,"5to8ft"] <-
    newamat[,"EM#"] + newamat[,"EM+"] + newamat[,"EM!"] +
    newamat[,"EM-"] + newamat[,"EM/"] + newamat[,"EM="]

amat2[,"8to10ft"] <-
    newamat[,"EL#"] + newamat[,"EL+"] + newamat[,"EL!"] +
    newamat[,"EL-"] + newamat[,"EL/"] + newamat[,"EL="]

amat2[,"NotSetter"] <- newamat[,"E"]

amat2[,"Middle"] <-
    newamat[,"P2#"] + newamat[,"P2+"] + newamat[,"P2="] +
    newamat[,"P2/"] + newamat[,"P3#"] + newamat[,"P3+"] +
    newamat[,"P3="] + newamat[,"P3/"] + newamat[,"P6#"] +
    newamat[,"P6+"] + newamat[,"P6/"] + newamat[,"P8#"] +
    newamat[,"P8+"] + newamat[,"P8="] + newamat[,"P8/"] +
    newamat[,"PW#"] + newamat[,"PW+"] + newamat[,"PW="] +
    newamat[,"PW/"]

amat2[,"Right"]  <-newamat[,"P5#"] + newamat[,"P5+"] +
    newamat[,"PK#"] + newamat[,"PK+"] + newamat[,"PK="] +
    newamat[,"PK/"] + newamat[,"PX#"] + newamat[,"PX+"] +
    newamat[,"PX="] + newamat[,"PX/"]
```

```
amat2[,"Left"]   <-newamat[,"PG#"] + newamat[,"PG+"] +
    newamat[,"PG="] + newamat[,"PG/"] + newamat[,"PH#"] +
    newamat[,"PH+"] + newamat[,"PH="] + newamat[,"PH/"] +
    newamat[,"PM#"] + newamat[,"PM+"] + newamat[,"PM="] +
    newamat[,"PM/"] +newamat[,"PR#"] + newamat[,"PR+"]

amat2[,"Back"]   <-newamat[,"PB#"] + newamat[,"PB+"] +
    newamat[,"PB="] + newamat[,"PB/"] + newamat[,"PD#"] +
    newamat[,"PD+"] + newamat[,"PD="] + newamat[,"PD/"] +
    newamat[,"PP#"] + newamat[,"PP+"] + newamat[,"PP="]

amat2[,"SetDump"]<-newamat[,"PS#"] + newamat[,"PS+"] +
    newamat[,"PS="] + newamat[,"PS/"]
amat2[,"OutSystem"]<-newamat[,"PA#"] + newamat[,"PA+"] +
    newamat[,"PA="]
amat2[,"Overpass"]<-newamat[,"PO#"] + newamat[,"PO+"] +
    newamat[,"PO="]

amat2[,"Dig #"] <- newamat[,"DH#"]
amat2[,"Dig +"] <- newamat[,"DH+"]
amat2[,"Dig !"] <- newamat[,"DH!"]
amat2[,"Dig -"] <- newamat[,"DH-"]
amat2[,"Dig /"] <- newamat[,"DH/"]
amat2[,"Dig ="] <- newamat[,"DH="]

amat2[,"Good"] <- newamat[,"Good"]
amat2[,"Continue"] <- newamat[,"Continue"]
amat2[,"Bad"] <- newamat[,"Bad"]

amat3 <- amat2
# Make sure that each state in the transition matrix has at least 1
prior count:
for(row in 1:nrow(amat2)){
    for(col in 1:ncol(amat2)){
        if(cmat2[row,col]>0 && amat2[row,col]==0) amat3[row,col] <- 1
}
}
apply(amat3,1,sum)

a.mat <- amat3
save(a.mat,file="collapsedamatR.txt")
#write(t(amat3), "collapsedamat.txt",ncol=ncol(amat2),sep = "\t")
#Write the game to a file
```

B.1.3 Calculating Unconditional Probabilities

```
#####################################################################
### Calculating the unconditional probability distributions
##       for passing, set distance, attack
#####################################################################
setwd("/Users/gradstudent/Documents/Master's Project/Women's VB Code
 and Data")
##  NOTE:  Be sure to install the package "abind" otherwise
## this code will NOT work  #####
library(abind)
# This loads c.mat and a.mat from the Collapsed Trans. matrix file
load("Collapsed Transition Matrices/By Set Distance/
collapsedcmatR-Final.txt")
load("Collapsed Transition Matrices/By Set Distance/
collapsedamatR-Final.txt")

c.matcol <- ncol(c.mat)
c.matrow <- nrow(c.mat)
###
## Assuming prior counts all equal to one
###
a.matweak <- a.mat
for(row in 1:c.matrow){
    for(col in 1:c.matcol){
        if(c.mat[row,col]>0) a.matweak[row,col] <- 1
        else a.matweak[row,col] <- 0
        }
}
a.mat <- a.matweak

#This just indicates what columns of c.mat are greater than zero
#and then it stores the column indexes in a matrix for each row.
indmat <- matrix(-1, nrow=c.matrow, ncol=c.matcol)
for(row in 1:c.matrow){
    index <- 1
    for(col in 1:c.matcol){
        if(c.mat[row,col]>0){
            indmat[row,index] <- col
            index <- index + 1
        }
    }
}

newhits <- c("0","Float 5-pt","Float 4-pt","Float 3-pt","Float 2-pt",
```

```
"Float 1-pt","Float 0-pt","Jump 5-pt","Jump 4-pt","Jump 3-pt",
"Jump 2-pt","Jump 1-pt","Jump 0-pt","4pt","3pt","2pt","1pt","0pt",
"PassOverpass","0to3ft","3to5ft","5to8ft","8to10ft","NotSetter","Middle",
"Right","Left","Back","SetDump","OutSystem","Overpass","Dig #","Dig
+","Dig !","Dig -","Dig /","Dig =","Good","Continue","Bad" )

#Specify what you want to look at:
serve<-c("Float 5-pt","Float 4-pt","Float 3-pt","Float 2-pt","Float
1-pt","Float 0-pt","Jump 5-pt","Jump 4-pt","Jump 3-pt",
 "Jump 2-pt","Jump 1-pt","Jump 0-pt")
pass <- c("4pt",  "3pt",  "2pt",  "1pt",  "0pt",  "PassOverpass","Dig
#","Dig +","Dig !","Dig -","Dig /","Dig =")
set <- c("0to3ft", "3to5ft", "5to8ft", "8to10ft", "NotSetter")
attack <- c("Middle", "Right", "Left", "Back",
            "SetDump", "OutSystem", "Overpass")
outcome <- c("Good")

#Total number of draws from posterior distribution
timestamp()
nloops <- 100000

post <- matrix(NA, nrow=c.matrow, ncol=c.matcol,
    dimnames=list(newhits,newhits))

#This will store the all the simulated transition matrices:
#allsetplacemat <- matrix(0, nrow=c.matrow, ncol=c.matcol,
#    dimnames=list(newhits,newhits))

name.col<-NULL
for(i in 1:length(newhits)){
for(j in 1:length(newhits)){
name.col<-c(name.col,paste(newhits[i],newhits[j],sep="/"))
}
}
allsetplacemat<-matrix(NA,nrow=nloops,ncol=length(name.col),
    dimnames=list(c(1:nloops),name.col))
lserve<-length(serve)
lset <- length(set)
lpass <- length(pass)
loutcome <- length(outcome)
lattack <- length(attack)

#Create a matrix to store the unconditional probabilities for serving
serveoverall <- matrix(NA, nrow=lserve, ncol=loutcome,
    dimnames=list(serve, outcome))
```

```
serveposts<-matrix(NA,nrow=100000,ncol= lserve,
    dimnames=list(c(1:100000),serve))


# Create a matrix to store the unconditional probabilities for passing
passoverall <- matrix(NA, nrow=lpass, ncol=loutcome,
    dimnames=list(pass, outcome))
passposts<-matrix(NA,nrow=100000,ncol=lpass,dimnames=list(c(1:100000)
    ,pass))

# Matrix to store unconditional probabilities for set placement
setplaceoverall <- matrix(NA, nrow=lset, ncol=loutcome,
    dimnames=list(set, outcome))
setplaceposts
<-matrix(NA,nrow=100000,ncol=lset,dimnames=list(c(1:100000)
,set))

# Matrix to store the unconditional probabilities for attack
attackoverall <- matrix(NA, nrow=lattack, ncol=loutcome,
    dimnames=list(attack,outcome))
attackposts<-matrix(NA,nrow=100000,ncol=lattack,
    dimnames=list(c(1:100000), attack))

timestamp()
for(loop in 1:nloops){
    # Generate a whole new matrix
    # Generate values from a gamma distribution -
    #   Convert to Dirichlet distribution
    for(row in 1:c.matrow){
        draws <- matrix(0, nrow=1, ncol=c.matcol)
        for(col in 1:c.matcol){
            index <- indmat[row,col]
            if(index == -1) {break}
            draws[index] <- rgamma(1, c.mat[row,index] +
                a.mat[row,index], 1)
        }
        # Convert to a dirichlet distribution
        post[row,] <- draws/sum(draws)
    }
    # Save the generated transition matrix
    allsettemp<-NULL
    for(i in 1:length(newhits)){
        allsettemp<-c(allsettemp,post[i,])
    }
```

```
 allsetplacemat[loop,] <- allsettemp

 ##Calculate the unconditional probabilities for Serving types
for(i in 1:lserve){
     for(j in 1:loutcome){
         prob <- 0
         prob <- prob + post[serve[i],outcome[j]]
         serveoverall[i,j] <- prob
     }
 }
 serveposts[loop,]<-serveoverall

 ###
 #Calculates the unconditional probabilities for passing types
 # based on the simulated transition matrix
 ###
 for(p in 1:lpass){
     passp <- pass[p]
     for(j in 1:loutcome){
         outcomej <- outcome[j]
         prob <- 0
         for(i in 1:lattack){
           attacki <- attack[i]
           for(k in 1:lset){
             setk <- set[k]
             prob <- prob
               + post[passp,setk]*post[setk,attacki]
               *post[attacki,outcomej]+post[passp,setk]
               *post[setk,"NotSetter"]*post["NotSetter",attacki]
               *post[attacki,outcomej]
               }
               prob <- prob  + post[passp,attacki]*
               post[attacki,outcomej]
            }
            for(k in 1:lset){
              setk <- set[k]
              prob <- prob +
              post[passp,setk]*post[setk,"NotSetter"]
              *post["NotSetter",outcomej] + post[passp,setk]*
              post[setk,outcomej]
            }
            prob <- prob + post[passp,outcomej]
            passoverall[p,j] <- prob
             }
     }
```

```
        passposts[loop,]<-passoverall
    }

    ###

    ## Compute unconditionals for set placement:

    ###
    for(k in 1:lset){
        setk <- set[k]
        for(j in 1:loutcome){
            outcomej <- outcome[j]
            prob <- 0
            for(i in 1:lattack){
                attacki <- attack[i]
                prob<-prob+post[setk,attacki]*post[attacki,outcomej] +
                      post[setk,"NotSetter"]*post["NotSetter",attacki]*
                      post[attacki, outcomej]
            }
            # Include the probability of going directly to an outcome
            #   from the set
            prob <- prob + post[setk,"NotSetter"]*
                   post["NotSetter",outcomej] + post[setk,outcomej]
           setplaceoverall[k,j] <- prob
        }
    }

    setplaceposts[loop,]<-setplaceoverall

    ###
    ## Compute the unconditionals for attacks:
    ###
    for(i in 1:lattack){
        for(j in 1:loutcome){
            prob <- 0
            prob <- prob + post[attack[i],outcome[j]]
            attackoverall[i,j] <- prob
        }
    }
    attackposts[loop,]<-attackoverall
}

timestamp() #About 2.5 hours to run

write.table(allsetplacemat,"allsetplacemat.txt",sep=",")
```

```
write.table(passposts,"passposts.txt",sep=",")
write.table(setplaceposts,"setplaceposts.txt",sep=",")
write.table(attackposts,"attackposts.txt",sep=",")
write.table(serveposts,"serveposts.txt",sep=",")
```

B.1.4 Calculating Goodness of Fit and Importance Scores

```
#Computes the Bayesian Chi-square for each cell of the transition
probability
#matrix.  The reason it does it for each cell is because the Bayesian
#Chi-square is not a multivariate test.

for(k in 0:34){
    row<-length(which(allsetplacemat[1,(1+k*31):(31+k*31)]>0))
    B.X2<-matrix(NA,nrow=n,ncol=row)
    prob<-allsetplacemat[i,(1+k*31):(31+k*31)]
    if(row==1){
        for(i in 1:n){
            prob<-allsetplacemat[i,(1+k*31):(31+k*31)]
            nonZero<-which(prob>0)
            prob.nonZero<-prob[nonZero]
            exp<-prob.nonZero*counts[k+1]
            actProb<-prob.nonZero
                for(j in 1:row){
                   obs<-rbinom(counts[k+1],1,p=actProb[j])
                   tab.obs<-table(obs)
                   int.cell<-exp[j]
                   B.X2[i,j]<-sum((tab.obs-int.cell)^2/int.cell)
                   }
            }
            }else{
        for(i in 1:n){
            prob<-allsetplacemat[i,(1+k*31):(31+k*31)]
            nonZero<-which(prob>0)
            prob.nonZero<-prob[nonZero]
            exp<-prob.nonZero*counts[k+1]
            actProb<-prob.nonZero
                for(j in 1:row){
                   obs<-rbinom(counts[k+1],1,p=actProb[j])
                   tab.obs<-table(obs)
                   int.cell<-exp[j]
                   other.cell<-sum(exp[-j])
                   tab.exp<-c(other.cell,int.cell)
                   B.X2[i,j]<-sum((tab.obs-tab.exp)^2/tab.exp)
                   }
```

```
            }
    }
    write.table(B.X2,file=paste("File",k,".txt",sep=""))
    print(k)
}

B.X2<-read.table("/Users/gradstudent/Documents/Master's Project
/Women's VB Code and Data/Bayes Chi-squared Values/File0.txt"
,header=TRUE)

pchis<-NULL
for(i in 1:ncol(B.X2)){
    pchis<-c(pchis,mean(pchisq(B.X2[,i],1,lower=TRUE)<0.05))
}

B.X2<-read.table("/Users/gradstudent/Documents/Master's Project/
Women's VB Code and Data/Bayes Chi-squared Values/File1.txt",
header=TRUE)
for(i in 1:ncol(B.X2)){
    pchis<-c(pchis,mean(pchisq(B.X2[,i],1,lower=TRUE)<0.05))
}

B.X2<-read.table("/Users/gradstudent/Documents/Master's Project
/Women's VB Code and Data/Bayes Chi-squared Values/File2.txt",
header=TRUE)
for(i in 1:ncol(B.X2)){
    pchis<-c(pchis,mean(pchisq(B.X2[,i],1,lower=TRUE)<0.05))
}

#Repeat process for all files

#Get the names for each chi-square thing I computed

i<-1
lab<-NULL
for(k in 0:30){
prob<-allsetplacemat[i,(1+k*31):(31+k*31)]
nonZero<-which(prob>0)
lab<-c(lab,names(prob[nonZero]))
}

res<-cbind(lab,pchis)
t(res)

#Computes the Importance Scores
```

```
attack<-read.csv("/Users/gradstudent/Documents/Master's Project
/Women's VB Code and Data/Results MCMC Good
Unconditional/attackposts.txt",
header=TRUE)
pass<-read.csv("/Users/gradstudent/Documents/Master's Project/
Women's VB Code and Data/Results MCMC Good
Unconditional/passposts.txt",
header=TRUE)
set<-read.csv("/Users/gradstudent/Documents/Master's Project/
Women's VB Code and Data/Results MCMC Good
Unconditional/setplaceposts.txt",
header=TRUE)
serve<-read.csv("/Users/gradstudent/Documents/Master's Project/
Women's VB Code and Data/Results MCMC Good
Unconditional/serveposts.txt",
header=TRUE)

IS.attack<-apply(attack,2,mean)/sqrt(apply(attack,2,var))
IS.pass<-apply(pass,2,mean)/sqrt(apply(pass,2,var))
IS.set<-apply(set,2,mean)/sqrt(apply(set,2,var))
IS.serve<-apply(serve,2,mean)/sqrt(apply(serve,2,var))

IS.all<-c(IS.attack,IS.pass,IS.set,IS.serve)
mean.all<-c(apply(attack,2,mean),apply(pass,2,mean),apply(set,2,mean),
apply(serve,2,mean))
var.all<-c(apply(attack,2,var),apply(pass,2,var),apply(set,2,var),
apply(serve,2,var))
vals<-cbind(mean.all,var.all,IS.all)
s.vals<-vals[order(vals[,3],decreasing=TRUE),]
round(s.vals,5)
```

B.2 Volleyball Logistic Regression

This section contains the code for the passing logistic regression model that we constructed. The code for the other models follows a similar format to this.

B.2.1 Coding for Data Matrices

```
#Need to create and X and a Y matrix for the volleyball data

####Scan in the data
team1<-read.csv("/Users/gradstudent/Documents/Master's Project/
VolleyballData Logistic/vbFinalLog.txt",header=TRUE)

#Creating an X just with the serves
```

```
#Just want to use the serve data
serve<-which(team1$skill=="Home Jump Serve"|
team1$skill=="Home Float Serve")
serve.data<-team1[serve,]

Skills.TF.Serve<-function(data){
data$HomeJump<-ifelse(data$skill=="Home Jump Serve",TRUE,FALSE)
data$HomeFloat<-ifelse(data$skill=="Home Float Serve",TRUE,FALSE)
return(data)
}

team1.serve<-Skills.TF.Serve(serve.data)

make.XServe<-function(data){
n<-nrow(data)
X.matrix<-matrix(0,nrow=n,ncol=3)
X.matrix[,1]<-1
X.matrix[data$HomeJump,2]<-data$score[data$HomeJump]
X.matrix[data$HomeFloat,3]<-data$score[data$HomeFloat]
return(X.matrix)
}

write.table(make.XServe(team1.serve),"xServe.txt",sep=" ",
col.names=FALSE,row.names=FALSE)
write.table(as.numeric(as.character(team1.serve$outcome)),
"yServe.txt",sep=" ",col.names=FALSE,row.names=FALSE)

#Passing X
pass<-which(team1$skill=="Pass")
pass.data<-team1[pass,]

Skills.TF.pass<-function(data){
data$Pass<-ifelse(data$skill=="Pass",TRUE,FALSE)
return(data)
}

team1.pass<-Skills.TF.pass(pass.data)
make.XPass<-function(data){
n<-nrow(data)
X.matrix<-matrix(0,nrow=n,ncol=2)
X.matrix[,1]<-1
X.matrix[data$Pass,2]<-data$score[data$Pass]
return(X.matrix)
}
```

```
write.table(make.XPass(team1.pass),"xPass.txt",sep=" ",
col.names=FALSE,row.names=FALSE)
write.table(as.numeric(as.character(team1.pass$outcome)),
"yPass.txt",sep=" ",col.names=FALSE,row.names=FALSE)

#Set X
team1<-read.csv("/Users/gradstudent/Documents/Master's Project
/VolleyballData Logistic/vbFinalLog2.txt",header=TRUE)
set<-which(team1$skill=="Set")
set.data<-team1[set,]

Skills.TF.set<-function(data){
data$Set<-ifelse(data$skill=="Set",TRUE,FALSE)
return(data)
}

team1.set<-Skills.TF.set(set.data)
make.XSet<-function(data){
n<-nrow(data)
X.matrix<-matrix(0,nrow=n,ncol=2)
X.matrix[,1]<-1
X.matrix[data$Set,2]<-data$score[data$Set]
return(X.matrix)
}

write.table(make.XSet(team1.set),"xSet.txt",sep=" ",
col.names=FALSE,row.names=FALSE)
write.table(as.numeric(as.character(team1.set$outcome)),
"ySet.txt",sep=" ",col.names=FALSE,row.names=FALSE)

#Dig X
dig<-which(team1$skill=="Dig")
dig.data<-team1[dig,]

Skills.TF.dig<-function(data){
data$Dig<-ifelse(data$skill=="Dig",TRUE,FALSE)
return(data)
}

team1.dig<-Skills.TF.dig(dig.data)
make.XSet<-function(data){
n<-nrow(data)
X.matrix<-matrix(0,nrow=n,ncol=2)
X.matrix[,1]<-1
X.matrix[data$Dig,2]<-data$score[data$Dig]
```

```
return(X.matrix)
}

write.table(make.XSet(team1.dig),"xDig.txt",sep=" ",
col.names=FALSE,row.names=FALSE)
write.table(as.numeric(as.character(team1.dig$outcome)),
"yDig.txt",sep=" ",col.names=FALSE,row.names=FALSE)
```

B.2.2 Logistic Regression Model

```
#include<math.h>
#include<gsl/gsl_math.h>
#include<gsl/gsl_statistics.h>
#include<gsl/gsl_sort.h>
#include<gsl/gsl_rng.h>
#include<gsl/gsl_randist.h>
#include<time.h>
/*Where M is the number of simulations*/
const int M=10000;
const int burn=1000;
const int NSIM=100300;
/*Don't need team variable any more*/

const double mu=0.0;
const double sig2=1000;
const int Nbetas=2;
const int NROW1=921; /*Number of rows in the x matrix*/

double Y1[921],One_min_Y1[921];
double x1[921][2];
double sumdot(double x[],double y[],int length);
double likelihood(double y[], double OneMinusY[],double x[][2],
double beta[],int N);
double sum(double x[],int n);
/*a is the needs to match the dim of your x matrix*/
void mvmult(double a[][2],double b[], double res[],int ar,
int ac);
void exponentiate(double x[], double final[], int N);
void logme(double x[], double final[], int N);
void addconst(double y,double x[], double final[],int N);
double g(double x, int beta_int,double betaAll[],double mu,
double sig2);
void getval (double betaAll[],int beta_int, double beta_ar,
double csig,double mu, double sigma2, double betaMat_int[]);
```

```
int main()
{
  gsl_rng *r, *s;
  r=gsl_rng_alloc(gsl_rng_mt19937);
  s=gsl_rng_alloc(gsl_rng_mt19937);
  int i,j,k,l,m,N;
  FILE *fi2,*fi3,*fi4;
  FILE *betas;

  double beta0[2],beta1[2],betaAll[Nbetas];

/*Initializing variables*/
  double beta_temp[Nbetas];
  double Lik, t;
  double csig_0=0.15;
  double csig_1=0.04;
  double beta0_ar=0;
  double beta1_ar=0;

  double old0,old1;
  double accept,cand,g_cand,g_old,u;

    beta0[0]=0;
    beta1[0]=0;

  fi2=fopen("yPass.txt","r");
  fi3=fopen("xPass.txt","r");

  /*Reading in my data files*/
  for(i=0; i<NROW1; i++) {
    fscanf(fi2, "%lf", &Y1[i]);
    One_min_Y1[i]=1-Y1[i];
    fscanf(fi3,"%lf %lf", &x1[i][0],&x1[i][1]);
  }
  fclose(fi2);
  fclose(fi3);

  /*Reading initial values for betas into a file*/
  betas=fopen("betasPass.txt","w");
  fprintf(betas,"%lf ",beta0[0]);
  fprintf(betas,"%lf ",beta1[0]);
  fprintf(betas,"\n");

  /*Starting the Simulation*/
```

```
  time_t t1=time(NULL);
 for(N=0;N<NSIM;N++){
   /*Notifies me where I am in my simulation*/
   if (N % 100 == 0 ){
      printf("%d\n",N);}

   /*Puts the current beta into a betaAll matrix*/
      betaAll[0]=beta0[0];
      betaAll[1]=beta1[0];

      /*Gets new starting values Random Num Gen*/
      gsl_rng_set(r,gsl_rng_get(r));
      gsl_rng_set(s,gsl_rng_get(s));

      /*Beta 0*/
      old0=betaAll[0];
      cand=gsl_ran_gaussian(r, csig_0)+old0;
      g_cand=g(cand,0,betaAll,mu,sig2);
      g_old=g(old0,0,betaAll,mu,sig2);
      accept=g_cand-g_old;
      /*Tests to see if we generated an acceptable value*/
      u=gsl_ran_flat (s, 0.0, 1.0);
      if(log(u)<accept){beta0[1]=cand;
beta0_ar+=1;} /*if the candidate value looks reasonable
        we accept it*/
      else{beta0[1]=old0;}
      betaAll[0]=beta0[1];

      /*Beta 1*/
      old1=betaAll[1];
      gsl_rng_set(r,gsl_rng_get(r));
      cand=gsl_ran_gaussian(r, csig_1)+old1;
      g_cand=g(cand,1,betaAll,mu,sig2);
      g_old=g(old1,1,betaAll,mu,sig2);
      accept=g_cand-g_old;
      gsl_rng_set(s,gsl_rng_get(s));
      /*See if value generated is an acceptable value*/
      u=gsl_ran_flat (s, 0.0, 1.0);
      if(log(u)<accept){beta1[1]=cand;beta1_ar+=1;}
      else{beta1[1]=old1;}
      betaAll[1]=beta1[1];


    /*Print out New Parameters to File*/
fprintf(betas,"%lf ",beta0[1]);
```

```
fprintf(betas,"%lf ",beta1[1]);
fprintf(betas,"\n");

/*Updating parameters so new values will be in the zero slot of
 the matrix*/
beta0[0]=beta0[1];
beta1[0]=beta1[1];
 }
  time_t t2=time(NULL);
  /*Tells me how long it took to run it*/
  printf("%d seconds elapsed\n", t2-t1);

  fclose(betas);

  /*Gives Acceptance rates for Metropolis Hastings part...want these
   to be between .40-.50. Alter corresponding c_sig values to get
   appropriate acceptance rates*/
  printf("Beta0 AR: %lf \n ",beta0_ar/NSIM);
  printf("Beta1 AR: %lf \n ",beta1_ar/NSIM);

  return 0;
}


/*This is the unnormalized posterior distribution*/
/*x is your candidate value you are testing, beta_int tells you what
 beta you are looking at, betaAll is a matrix of all the current
betas for that iteration, mu is the mean for that beta, and sig2
is the variance for that beta*/
double g(double x, int beta_int,double betaAll[],double mu,
 double sig2){
  int i,j;
  double beta_temp[Nbetas];
  double final;
  double Lik=0;
  betaAll[beta_int]=x;
  Lik=likelihood(Y1,One_min_Y1,x1,betaAll,NROW1);
  final=Lik-((x-mu)*(x-mu))/(2*sig2);
  return final;
  }

/*Calculates the log likelihood of the Bernoulli distribution*/
/*y is the response, oneMinus Y is a vector of 1-y, X is your x
matrix, beta is your beta matrix, N indicates the number of rows
in X*/
```

```
double likelihood(double y[], double OneMinusY[],double x[][Nbetas],
double beta[],int N){
  double Xbeta[N],yXbeta[N],expXBeta[N],L1expXB[N];
  double res;
  int i;
  double a,b,c,negY[N],OnePlusXBeta[N];
  mvmult(x,beta,Xbeta,N,Nbetas);
  exponentiate(Xbeta,expXBeta,N);
  addconst(1.0,expXBeta,OnePlusXBeta,N);
  logme(OnePlusXBeta,L1expXB,N);
  b=sumdot(y,L1expXB,N);
  a=sumdot(y,Xbeta,N);
  c=sumdot(OneMinusY,L1expXB,N);
  res=a-b-c;
  return res;
}

/*Calculates the sum of a vector*/
double sum(double x[],int n){

  int i;
  double res=0;
  for (i=0;i<n; i++){
    res+=x[i];
  }
  return res;
}

/*exponentiates a vector*/
void exponentiate(double x[], double final[], int N){
  int i;
  for(i=0;i<N;i++){
    final[i]=exp(x[i]);
  }
}

/*takes the log of a vector*/
void logme(double x[], double final[], int N){
  int i;
  for(i=0;i<N;i++){
    final[i]=log(x[i]);
  }
}

/*Does Vector Vector Multiplication...returns a single value*/
```

```
double sumdot(double x[],double y[],int length){
  int i;
  double res=0.0;
  for(i=0;i<length;i++){
    res+=x[i]*y[i];
  }
  return(res);
}

/*Does Matrix vector multiplication*/
/*ar is the dimension of rows in a, k is the number of columns*/
void mvmult(double a[][Nbetas],double b[], double res[],int ar,
int ac){
  int i, j,k;
  for(i=0; i<ar;i++){
      res[i]=0.0;
      for(k=0;k<ac;k++){
res[i]+=a[i][k]*b[k];
      }
    }
  }

/*Adds a constant to a vector*/
void addconst(double y,double x[], double final[],int N){

  int i;
  for(i=0;i<N;i++){
    final[i]=x[i]+y;
  }
}
```

B.2.3 Calculating Goodness of Fit and Importance Scores

```
#Goodness of Fit
betas<-read.table("/Users/gradstudent/Documents/Master's Project/
VolleyballData Logistic/betasServe.txt",header=FALSE)
names(betas)<-c("Intercept",'Jump','Float')

betas<-betas[-c(1:301),]

x<-read.table("/Users/gradstudent/Documents/Master's Project/
VolleyballData Logistic/xServe.txt",header=FALSE)
names(x)<-c("Intercept",'Jump','Float')
y<-read.table("/Users/gradstudent/Documents/Master's Project/
VolleyballData Logistic/yServe.txt",header=FALSE)
```

```
tabY<-table(y)

n<-nrow(betas)

Beta<-betas
X<-as.matrix(x)
n<-nrow(y)
BX2<-rep(0,n)
for(i in 1:n){
   tabs<-rep(0,2)
   betas<-matrix(as.numeric(Beta[i,]),ncol=1)
   prob<-exp(X%*%betas)/(1+exp(X%*%betas))
   vals<-rbern(nrow(X),prob)
   tabs<-tabs+table(vals)
   BX2[i]<-sum(((tabs-tabY)^2/tabY))
}
   mean(pchisq(BX2,1,lower=TRUE)<0.05)#0.01635514

#Volleyball Logistic Regression--Goodness of Fit Pass
betas<-read.table("/Users/gradstudent/Documents/Master's Project
/VolleyballData Logistic/betasPass.txt",header=FALSE)
names(betas)<-c("Intercept",'Pass')

betas<-betas[-c(1:301),]

x<-read.table("/Users/gradstudent/Documents/Master's Project
/VolleyballData Logistic/xPass.txt",header=FALSE)
names(x)<-c("Intercept",'Pass')

y<-read.table("/Users/gradstudent/Documents/Master's Project/
VolleyballData Logistic/yPass.txt",header=FALSE)
tabY<-table(y)
n<-nrow(betas)

Beta<-betas
X<-as.matrix(x)
n<-nrow(y)
BX2<-rep(0,n)
for(i in 1:n){
   tabs<-rep(0,2)
   betas<-matrix(as.numeric(Beta[i,]),ncol=1)
   prob<-exp(X%*%betas)/(1+exp(X%*%betas))
   vals<-rbern(nrow(X),prob)
   tabs<-tabs+table(vals)
   BX2[i]<-sum(((tabs-tabY)^2/tabY))
```

```
}
   mean(pchisq(BX2,1,lower=TRUE)<0.05) #0.02388708

#Sets
betas<-read.table("/Users/gradstudent/Documents/Master's Project/
VolleyballData Logistic/betasSet.txt",header=FALSE)
names(betas)<-c("Intercept",'Set')
betas<-betas[-c(1:301),]

x<-read.table("/Users/gradstudent/Documents/Master's Project/
VolleyballData Logistic/xSet.txt",header=FALSE)
names(x)<-c("Intercept",'Set')

y<-read.table("/Users/gradstudent/Documents/Master's Project/
VolleyballData Logistic/ySet.txt",header=FALSE)
tabY<-table(y)

n<-nrow(betas)

Beta<-betas
X<-as.matrix(x)
n<-nrow(y)
BX2<-rep(0,n)
for(i in 1:n){
   tabs<-rep(0,2)
   betas<-matrix(as.numeric(Beta[i,]),ncol=1)
   prob<-exp(X%*%betas)/(1+exp(X%*%betas))
   vals<-rbern(nrow(X),prob)
   tabs<-tabs+table(vals)
   BX2[i]<-sum(((tabs-tabY)^2/tabY))
}
mean(pchisq(BX2,1,lower=TRUE)<0.05) #0.04582409

#Dig
betas<-read.table("/Users/gradstudent/Documents/Master's Project
/VolleyballData Logistic/betasDig.txt",header=FALSE)
names(betas)<-c("Intercept",'Dig')

betas<-betas[-c(1:301),]

x<-read.table("/Users/gradstudent/Documents/Master's Project/
VolleyballData Logistic/xDig.txt",header=FALSE)
names(x)<-c("Intercept",'Dig')

y<-read.table("/Users/gradstudent/Documents/Master's Project/
```

```
VolleyballData Logistic/yDig.txt",header=FALSE)
tabY<-table(y)

n<-nrow(betas)
Beta<-betas
X<-as.matrix(x)
n<-nrow(y)
BX2<-rep(0,n)
for(i in 1:n){
   tabs<-rep(0,2)
   betas<-matrix(as.numeric(Beta[i,]),ncol=1)
   prob<-exp(X%*%betas)/(1+exp(X%*%betas))
   vals<-rbern(nrow(X),prob)
   tabs<-tabs+table(vals)
   BX2[i]<-sum(((tabs-tabY)^2/tabY))
   }
mean(pchisq(BX2,1,lower=TRUE)<0.05)

#MCMC
load("Results MCMC Good Unconditional/allsetplacemat.txt")
load("R Code Cleaned Up/volleyclean.txt")#file is called vb
load("Collapsed Transition Matrices/By Set Distance/
collapsedcmatR2.txt")
dims<-colnames(allsetplacemat)

counts<-apply(c.mat,1,sum)
n<-nrow(allsetplacemat)

#Lets try it for a specific row of my matrix
row1<-length(which(allsetplacemat[1,1:31]>0))
B.X2<-matrix(NA,nrow=n,ncol=7)

#Importance Scores
betas<-read.table("/Users/gradstudent/Documents/Master's Project
/VolleyballData Logistic/betasServe.txt",header=FALSE)
plot(betas[1:1000,2],type="l")
betas<-betas[-c(1:301),]
plot(density(betas[,2]),col="red")
colnames(betas)<-c("Intercept","HomeJump","HomeFloat")

#Importance Scores
ImportanceScores<-apply(betas,2,mean)/sqrt(apply(betas,2,var))
sort(ImportanceScores,decreasing=TRUE)

#Pass
```

```
betas<-read.table("/Users/gradstudent/Documents/Master's Project
/VolleyballData Logistic/betasPass.txt",header=FALSE)
plot(betas[1:1000,2],type="l")
betas<-betas[-c(1:301),]
plot(density(betas[,2]),col="red")
colnames(betas)<-c("Intercept","Pass")

#Importance Scores
ImportanceScores<-apply(betas,2,mean)/sqrt(apply(betas,2,var))
sort(ImportanceScores,decreasing=TRUE)

#Set
betas<-read.table("/Users/gradstudent/Documents/Master's Project
/VolleyballData Logistic/betasSet.txt",header=FALSE)
plot(betas[1:5000,2],type="l")
betas<-betas[-c(1:301),]
plot(density(betas[,2]),col="red")
colnames(betas)<-c("Intercept","Set")

#Calculating Importance Scores
ImportanceScores<-apply(betas,2,mean)/sqrt(apply(betas,2,var))
sort(ImportanceScores,decreasing=TRUE)

#Digs
betas<-read.table("/Users/gradstudent/Documents/Master's Project
/VolleyballData Logistic/betasDig.txt",header=FALSE)
plot(betas[1:2000,2],type="l")
betas<-betas[-c(1:301),]
plot(density(betas[,2]),col="red")
colnames(betas)<-c("Intercept","Dig")

#Calculating Importance Scores
ImportanceScores<-apply(betas,2,mean)/sqrt(apply(betas,2,var))
sort(ImportanceScores,decreasing=TRUE)
```

B.3 Soccer Hierarchical Logistic Regression

```
#include<math.h>
#include<gsl/gsl_math.h>
#include<gsl/gsl_statistics.h>
#include<gsl/gsl_sort.h>
#include<gsl/gsl_rng.h>
#include<gsl/gsl_randist.h>
#include<time.h>
/*Where M is the number of simulations*/
```

```
const int M=100000;
const int burn=1000;
const int NSIM=101000;
const int teams=11;
const double mu0_mean=0.0,mu1_mean=0.0,mu2_mean=0.0,mu3_mean=0.0,
mu4_mean=0.0;
const double mu0_sig2=1000,mu1_sig2=1000,mu2_sig2=1000,mu3_sig2=1000,
mu4_sig2=1000;
const double sig0_a=2.001,sig1_a=2.001,sig2_a=2.001,sig3_a=2.001,
sig4_a=2.001;
const double sig0_b=0.0002,sig1_b=0.0002,sig2_b=0.0002,sig3_b=0.0002,
sig4_b=0.0002;
const int Nbetas=5;
const int NROW1=15569;
const int NROW2=1298;
const int NROW3=1455;
const int NROW4=1231;
const int NROW5=1289;
const int NROW6=1383;
const int NROW7=1641;
const int NROW8=1335;
const int NROW9=1482;
const int NROW10=1345;
const int NROW11=1835;
double Y1[15569],One_min_Y1[15569];
double Y2[1298],One_min_Y2[1298];
double Y3[1455],One_min_Y3[1455];
double Y4[1231],One_min_Y4[1231];
double Y5[1289],One_min_Y5[1289];
double Y6[1383],One_min_Y6[1383];
double Y7[1641],One_min_Y7[1641];
double Y8[1335],One_min_Y8[1335];
double Y9[1482],One_min_Y9[1482];
double Y10[1345],One_min_Y10[1345];
double Y11[1835],One_min_Y11[1835];
double x1[15569][5],x2[1298][5],x3[1455][5],x4[1231][5],x5[1289][5],
x6[1383][5];
double x7[1641][5],x8[1335][5],x9[1482][5],x10[1345][5],x11[1835][5];
double sumdot(double x[],double y[],int length);
double likelihood(double y[], double OneMinusY[],double x[][Nbetas],
double beta[],int N);
double sum(double x[],int n);
void mvmult(double a[][5],double b[], double res[],int ar, int ac);
void exponentiate(double x[], double final[], int N);
void logme(double x[], double final[], int N);
```

```
void addconst(double y,double x[], double final[],int N);
double g(double x, int jj, int beta_int,double betaAll[][teams],
double mu, double sig2);
void applymean(double BetaMat[][teams], double meanBeta[Nbetas],
int teams);


int main()
{
  gsl_rng *r, *s;
  r=gsl_rng_alloc(gsl_rng_mt19937);
  s=gsl_rng_alloc(gsl_rng_mt19937);
  int i,j,k,l,m,N;
  FILE *fi2,*fi3,*fi4;
  FILE *b0,*b1,*b2,*b3,*b4,*mu,*sig2;

  /*I ony need to save the previous values...rewrite code to do this*/
    double beta0[2][teams],beta1[2][teams],beta2[2][teams],
beta3[2][teams],beta4[2][teams],betaAll[Nbetas][teams];

  double beta_temp[Nbetas];
  double Lik, t;
  double mu_0[2],mu_1[2],mu_2[2],mu_3[2],mu_4[2];
  double sig2_0[2],sig2_1[2],sig2_2[2],sig2_3[2],sig2_4[2];
  double csig_0[]={.05,0.16,0.5,0.25,0.12,0.2,0.3,
0.2,0.3,0.2,0.15};
  double csig_1[]={0.015, 0.05,0.2,0.055,0.045,0.04, 0.07,
0.05, 0.055, 0.05, .035};
  double csig_2[]={0.02, 0.075,0.37,0.1,0.07,0.06,0.12,0.09,
0.10, 0.1,  .05};
  double csig_3[]={0.04, 0.1,0.348,0.14,0.085,0.10, 0.18, 0.10,
0.12, 0.13,  .07};
  double csig_4[]={0.05, 0.16,.9,0.25,.15,.15,.25, 0.17,
 .25,   .3,  .12};
  double beta0_ar[]={0,0,0,0,0,0,0,0,0,0,0};
  double beta1_ar[]={0,0,0,0,0,0,0,0,0,0,0};
  double beta2_ar[]={0,0,0,0,0,0,0,0,0,0,0};
  double beta3_ar[]={0,0,0,0,0,0,0,0,0,0,0};
  double beta4_ar[]={0,0,0,0,0,0,0,0,0,0,0};
  double old0,old1,old2,old3,old4;
  double accept,cand,g_cand,g_old,u;
  double meanBetas[Nbetas];
  double mustar0,sigstar0,mustar1,sigstar1,mustar2,sigstar2,
mustar3,sigstar3,mustar4,sigstar4;
  double astar_0, bstar_0,astar_1, bstar_1,astar_2, bstar_2,
```

```
astar_3, bstar_3,astar_4, bstar_4;
  double res=0.0,res1=0,res2=0,res3=0,res4=0;

  /*initialize Betas0*/
  for(i=0; i<teams; i++){
    beta0[0][i]=0;
    beta1[0][i]=0;
    beta2[0][i]=0;
    beta3[0][i]=0;
    beta4[0][i]=0;
    }
  /*Initialize mu and sig*/
    mu_0[0]=0;
    mu_1[0]=0;
    mu_2[0]=0;
    mu_3[0]=0;
    mu_4[0]=0;
    sig2_0[0]=1;
    sig2_1[0]=1;
    sig2_2[0]=1;
    sig2_3[0]=1;
    sig2_4[0]=1;

  fi2=fopen("y1.txt","r");
  fi3=fopen("x1.txt","r");
  for(i=0; i<NROW1; i++) {
   fscanf(fi2, "%lf", &Y1[i]);
   One_min_Y1[i]=1-Y1[i];
   fscanf(fi3, "%lf %lf %lf %lf %lf", &x1[i][0],&x1[i][1],&x1[i][2],
&x1[i][3],&x1[i][4]);
   }
  fclose(fi2);
  fclose(fi3);

  fi2=fopen("y2.txt","r");
  fi3=fopen("x2.txt","r");
  for(i=0; i<NROW2; i++) {
   fscanf(fi2, "%lf", &Y2[i]);
   One_min_Y2[i]=1-Y2[i];
   fscanf(fi3, "%lf %lf %lf %lf %lf", &x2[i][0],&x2[i][1],&x2[i][2],
&x2[i][3],&x2[i][4]);
   }
  fclose(fi2);
  fclose(fi3);
```

```
  fi2=fopen("y3.txt","r");
  fi3=fopen("x3.txt","r");
  for(i=0; i<NROW3; i++) {
   fscanf(fi2, "%lf", &Y3[i]);
   One_min_Y3[i]=1-Y3[i];
   fscanf(fi3, "%lf %lf %lf %lf %lf", &x3[i][0],&x3[i][1],&x3[i][2],
&x3[i][3],&x3[i][4]);
   }
  fclose(fi2);
  fclose(fi3);

  fi2=fopen("y4.txt","r");
  fi3=fopen("x4.txt","r");
  for(i=0; i<NROW4; i++) {
   fscanf(fi2, "%lf", &Y4[i]);
   One_min_Y4[i]=1-Y4[i];
   fscanf(fi3, "%lf %lf %lf %lf %lf", &x4[i][0],&x4[i][1],&x4[i][2],
&x4[i][3],&x4[i][4]);
   }
  fclose(fi2);
  fclose(fi3);

  fi2=fopen("y5.txt","r");
  fi3=fopen("x5.txt","r");
  for(i=0; i<NROW5; i++) {
   fscanf(fi2, "%lf", &Y5[i]);
   One_min_Y5[i]=1-Y5[i];
   fscanf(fi3, "%lf %lf %lf %lf %lf", &x5[i][0],&x5[i][1],
&x5[i][2],&x5[i][3],&x5[i][4]);
   }
  fclose(fi2);
  fclose(fi3);

  fi2=fopen("y6.txt","r");
  fi3=fopen("x6.txt","r");
  for(i=0; i<NROW6; i++) {
   fscanf(fi2, "%lf", &Y6[i]);
   One_min_Y6[i]=1-Y6[i];
   fscanf(fi3, "%lf %lf %lf %lf %lf", &x6[i][0],&x6[i][1],
&x6[i][2],&x6[i][3],&x6[i][4]);
   }
  fclose(fi2);
  fclose(fi3);

  fi2=fopen("y7.txt","r");
```

```
  fi3=fopen("x7.txt","r");
  for(i=0; i<NROW7; i++) {
   fscanf(fi2, "%lf", &Y7[i]);
   One_min_Y7[i]=1-Y7[i];
   fscanf(fi3, "%lf %lf %lf %lf %lf", &x7[i][0],&x7[i][1],
&x7[i][2],&x7[i][3],&x7[i][4]);
   }
  fclose(fi2);
  fclose(fi3);

   fi2=fopen("y8.txt","r");
  fi3=fopen("x8.txt","r");
  for(i=0; i<NROW8; i++) {
   fscanf(fi2, "%lf", &Y8[i]);
   One_min_Y8[i]=1-Y8[i];
   fscanf(fi3, "%lf %lf %lf %lf %lf", &x8[i][0],&x8[i][1],
&x8[i][2],&x8[i][3],&x8[i][4]);
   }
  fclose(fi2);
  fclose(fi3);

  fi2=fopen("y9.txt","r");
  fi3=fopen("x9.txt","r");
  for(i=0; i<NROW9; i++) {
   fscanf(fi2, "%lf", &Y9[i]);
   One_min_Y9[i]=1-Y9[i];
   fscanf(fi3, "%lf %lf %lf %lf %lf", &x9[i][0],&x9[i][1],
&x9[i][2],&x9[i][3],&x9[i][4]);
   }
  fclose(fi2);
  fclose(fi3);

   fi2=fopen("y10.txt","r");
  fi3=fopen("x10.txt","r");
  for(i=0; i<NROW10; i++) {
   fscanf(fi2, "%lf", &Y10[i]);
   One_min_Y10[i]=1-Y10[i];
   fscanf(fi3, "%lf %lf %lf %lf %lf",
               %&x10[i][0],&x10[i][1],&x10[i][2],
&x10[i][3],&x10[i][4]);
   }
  fclose(fi2);
  fclose(fi3);

   fi2=fopen("y11.txt","r");
```

```
  fi3=fopen("x11.txt","r");
  for(i=0; i<NROW11; i++) {
   fscanf(fi2, "%lf", &Y11[i]);
   One_min_Y11[i]=1-Y11[i];
   fscanf(fi3, "%lf %lf %lf %lf %lf",
                %&x11[i][0],&x11[i][1],&x11[i][2],
&x11[i][3],&x11[i][4]);
   }
  fclose(fi2);
  fclose(fi3);

  /*Printing Inital Betas*/
  b0=fopen("beta0.txt","w");
  b1=fopen("beta1.txt","w");
  b2=fopen("beta2.txt","w");
  b3=fopen("beta3.txt","w");
  b4=fopen("beta4.txt","w");

  for(i=0; i<teams;i++){
    fprintf(b0,"%lf ",beta0[0][i]);
    fprintf(b1,"%lf ",beta1[0][i]);
    fprintf(b2,"%lf ",beta2[0][i]);
    fprintf(b3,"%lf ",beta3[0][i]);
    fprintf(b4,"%lf ",beta4[0][i]);

  }
  fprintf(b0,"\n");
  fprintf(b1,"\n");
  fprintf(b2,"\n");
  fprintf(b3,"\n");
  fprintf(b4,"\n");

  mu=fopen("allmu.txt","w");
  fprintf(mu,"%lf %lf %lf %lf %lf \n",mu_0[0],mu_1[0],mu_2[0],
mu_3[0],mu_4[0]);

  sig2=fopen("allsig2.txt","w");
  fprintf(sig2,"%lf %lf %lf %lf %lf \n",sig2_0[0],sig2_1[0],
sig2_2[0],sig2_3[0],sig2_4[0]);

  time_t t1=time(NULL);
  int NSIM1;
  for(N=0;N<NSIM;N++){
    if (N % 100 == 0 ){
      printf("%d\n",N);}
```

```
    /*Loops around all the teams*/
    /*for(m=0;m<teams; m++){*/
    for(k=0;k<teams;k++){
      betaAll[0][k]=beta0[0][k];
      betaAll[1][k]=beta1[0][k];
      betaAll[2][k]=beta2[0][k];
      betaAll[3][k]=beta3[0][k];
      betaAll[4][k]=beta4[0][k];
    }

    for(m=0; m<teams; m++){
      gsl_rng_set(r,gsl_rng_get(r));
      gsl_rng_set(s,gsl_rng_get(s));
      /*Beta 0 for team m*/
      old0=betaAll[0][m];
      cand=gsl_ran_gaussian(r, csig_0[m])+old0;
      g_cand=g(cand,m,0,betaAll,mu_0[0],sig2_0[0]);
      g_old=g(old0,m,0,betaAll,mu_0[0],sig2_0[0]);
      accept=g_cand-g_old;

      u=gsl_ran_flat (s, 0.0, 1.0);
      if(log(u)<accept){beta0[1][m]=cand;
beta0_ar[m]+=1;
}
      else{beta0[1][m]=old0;}
      betaAll[0][m]=beta0[1][m];

      /*Beta 1 for team m*/
      old1=betaAll[1][m];
      gsl_rng_set(r,gsl_rng_get(r));
      cand=gsl_ran_gaussian(r, csig_1[m])+old1;
      g_cand=g(cand,m,1,betaAll,mu_1[0],sig2_1[0]);
      g_old=g(old1,m,1,betaAll,mu_1[0],sig2_1[0]);
      accept=g_cand-g_old;
      gsl_rng_set(s,gsl_rng_get(s));
      u=gsl_ran_flat (s, 0.0, 1.0);
      if(log(u)<accept){beta1[1][m]=cand;beta1_ar[m]+=1;}
      else{beta1[1][m]=old1;}
      betaAll[1][m]=beta1[1][m];

      /*Beta 2 for team m*/
      old2=betaAll[2][m];
      gsl_rng_set(r,gsl_rng_get(r));
      cand=gsl_ran_gaussian(r, csig_2[m])+old2;
      g_cand=g(cand,m,2,betaAll,mu_2[0],sig2_2[0]);
```

```
  g_old=g(old2,m,2,betaAll,mu_2[0],sig2_2[0]);
  accept=g_cand-g_old;
  gsl_rng_set(s,gsl_rng_get(s));
  u=gsl_ran_flat (s, 0.0, 1.0);
  if(log(u)<accept){beta2[1][m]=cand;beta2_ar[m]+=1;}
  else{beta2[1][m]=old2;}
  betaAll[2][m]=beta2[1][m];

  /*Beta 3 for team m*/
  old3=betaAll[3][m];
  gsl_rng_set(r,gsl_rng_get(r));
  cand=gsl_ran_gaussian(r, csig_3[m])+old3;
  g_cand=g(cand,m,3,betaAll,mu_3[0],sig2_3[0]);
  g_old=g(old3,m,3,betaAll,mu_3[0],sig2_3[0]);
  accept=g_cand-g_old;
  gsl_rng_set(s,gsl_rng_get(s));
  u=gsl_ran_flat (s, 0.0, 1.0);
  if(log(u)<accept){beta3[1][m]=cand;beta3_ar[m]+=1;}
  else{beta3[1][m]=old3;}
  betaAll[3][m]=beta3[1][m];

  /*Beta 4 for team m*/
  old4=betaAll[4][m];
  gsl_rng_set(r,gsl_rng_get(r));
  cand=gsl_ran_gaussian(r, csig_4[m])+old4;
  g_cand=g(cand,m,4,betaAll,mu_4[0],sig2_4[0]);
  g_old=g(old4,m,4,betaAll,mu_4[0],sig2_4[0]);
  accept=g_cand-g_old;
  gsl_rng_set(s,gsl_rng_get(s));
  u=gsl_ran_flat (s, 0.0, 1.0);
  if(log(u)<accept){beta4[1][m]=cand;beta4_ar[m]+=1;}
  else{beta4[1][m]=old4;}
  betaAll[4][m]=beta4[1][m];
}

for(k=0;k<teams;k++){
  betaAll[0][k]=beta0[1][k];
  betaAll[1][k]=beta1[1][k];
  betaAll[2][k]=beta2[1][k];
  betaAll[3][k]=beta3[1][k];
  betaAll[4][k]=beta4[1][k];
}

/*  for(i=0;i<Nbetas;i++){
  for(j=0;j<teams;j++){
```

```
printf("%lf ",betaAll[i][j]);
      }
      printf("\n");
    }
    printf("\n");*/

    /*Generating Data for the complete conditionals for mu*/
    applymean(betaAll,meanBetas,teams);

    /*Generating a value for Mu 0*/

    double num0;
    num0=(teams*meanBetas[0]*mu0_sig2+sig2_0[0]*mu0_mean);
    mustar0=num0/(mu0_sig2*teams+sig2_0[0]);
    sigstar0=(sig2_0[0]*mu0_sig2)/(teams*mu0_sig2+sig2_0[0]);
      gsl_rng_set(r,gsl_rng_get(r));
      mu_0[1]=gsl_ran_gaussian(r,sqrt(sigstar0))+mustar0;

      /*Generating a value for Mu 1*/
    mustar1=(teams*meanBetas[1]*mu1_sig2+sig2_1[0]*mu1_mean)/
(mu1_sig2*teams+sig2_1[0]);
    sigstar1=(sig2_1[0]*mu1_sig2)/(teams*mu1_sig2+sig2_1[0]);
      gsl_rng_set(r,gsl_rng_get(r));
      mu_1[1]=gsl_ran_gaussian(r,sqrt(sigstar1))+mustar1;

     /*Generating a value for Mu 2*/
    mustar2=(teams*meanBetas[2]*mu2_sig2+sig2_2[0]*mu2_mean)/
(mu2_sig2*teams+sig2_2[0]);
    sigstar2=(sig2_2[0]*mu2_sig2)/(teams*mu2_sig2+sig2_2[0]);
      gsl_rng_set(r,gsl_rng_get(r));
      mu_2[1]=gsl_ran_gaussian(r,sqrt(sigstar2))+mustar2;

     /*Generating a value for Mu 3*/
      mustar3=(teams*meanBetas[3]*mu3_sig2+sig2_3[0]*mu3_mean)/
(mu3_sig2*teams+sig2_3[0]);
      sigstar3=(sig2_3[0]*mu3_sig2)/(teams*mu3_sig2+sig2_3[0]);
      gsl_rng_set(r,gsl_rng_get(r));
      mu_3[1]=gsl_ran_gaussian(r,sqrt(sigstar3))+mustar3;

     /*Generating a value for Mu 4*/
      mustar4=(teams*meanBetas[4]*mu4_sig2+sig2_4[0]*mu4_mean)/
(mu4_sig2*teams+sig2_4[0]);
      sigstar4=(sig2_4[0]*mu4_sig2)/(teams*mu4_sig2+sig2_4[0]);
      gsl_rng_set(r,gsl_rng_get(r));
      mu_4[1]=gsl_ran_gaussian(r,sqrt(sigstar4))+mustar4;
```

```
      /*Generating Data for the complete conditional for Sigma 2*/
      /*Sigma 2_0*/
      res=0,res1=0,res2=0,res3=0,res4=0;
    for(i=0; i<teams; i++){
res+=(beta0[1][i]-mu_0[1])*(beta0[1][i]-mu_0[1]);
res1+=(beta1[1][i]-mu_1[1])*(beta1[1][i]-mu_1[1]);
res2+=(beta2[1][i]-mu_2[1])*(beta2[1][i]-mu_2[1]);
res3+=(beta3[1][i]-mu_3[1])*(beta3[1][i]-mu_3[1]);
res4+=(beta4[1][i]-mu_4[1])*(beta4[1][i]-mu_4[1]);
      }
      astar_0=teams/2.0+sig0_a;
      bstar_0=1.0/(res/2.0 + 1.0/sig0_b);
      gsl_rng_set(r,gsl_rng_get(r));
      sig2_0[1]=1.0/gsl_ran_gamma(r,astar_0,bstar_0);

      /*Sigma 2_1*/
      res=0;
      astar_1=teams/2.0+sig1_a;
      bstar_1=1.0/(res1/2.0 + 1.0/sig1_b);
      gsl_rng_set(r,gsl_rng_get(r));
      sig2_1[1]=1.0/gsl_ran_gamma(r,astar_1,bstar_1);

      /*Sigma 2_2*/
      res=0;
      astar_2=teams/2.0+sig2_a;
      bstar_2=1.0/(res2/2.0 + 1.0/sig2_b);
      gsl_rng_set(r,gsl_rng_get(r));
      sig2_2[1]=1.0/gsl_ran_gamma(r,astar_2,bstar_2);

      /*Sigma 2_3*/
      res=0;
      astar_3=teams/2.0+sig3_a;
      bstar_3=1.0/(res3/2.0 + 1.0/sig3_b);
      gsl_rng_set(r,gsl_rng_get(r));
      sig2_3[1]=1.0/gsl_ran_gamma(r,astar_3,bstar_3);

      /*Sigma 2_4*/
      res=0;
      astar_4=teams/2.0+sig4_a;
      bstar_4=1.0/(res4/2.0 + 1.0/sig4_b);
      gsl_rng_set(r,gsl_rng_get(r));
      sig2_4[1]=1.0/gsl_ran_gamma(r,astar_4,bstar_4);

      /*Print out New Parameters to File*/
```

```
      for(i=0; i<teams;i++){
fprintf(b0,"%lf ",beta0[1][i]);
fprintf(b1,"%lf ",beta1[1][i]);
fprintf(b2,"%lf ",beta2[1][i]);
fprintf(b3,"%lf ",beta3[1][i]);
fprintf(b4,"%lf ",beta4[1][i]);
      }
      fprintf(b0,"\n");
      fprintf(b1,"\n");
      fprintf(b2,"\n");
      fprintf(b3,"\n");
      fprintf(b4,"\n");

      fprintf(mu,"%lf %lf %lf %lf %lf
                  \n",mu_0[1],mu_1[1],mu_2[1],mu_3[1], mu_4[1]);

      fprintf(sig2,"%lf %lf %lf %lf %lf
                    \n",sig2_0[1],sig2_1[1],sig2_2[1], sig2_3[1],
                    sig2_4[1]);

      /*Updating parameters so new values will be in the zero slot
of the matrix*/

      for(i=0;i<teams;i++){
beta0[0][i]=beta0[1][i];
beta1[0][i]=beta1[1][i];
beta2[0][i]=beta2[1][i];
beta3[0][i]=beta3[1][i];
beta4[0][i]=beta4[1][i];
      }

      mu_0[0]=mu_0[1];
      mu_1[0]=mu_1[1];
      mu_2[0]=mu_2[1];
      mu_3[0]=mu_3[1];
      mu_4[0]=mu_4[1];

      sig2_0[0]=sig2_0[1];
      sig2_1[0]=sig2_1[1];
      sig2_2[0]=sig2_2[1];
      sig2_3[0]=sig2_3[1];
      sig2_4[0]=sig2_4[1];
  }
  time_t t2=time(NULL);
  printf("%d seconds elapsed\n", t2-t1);
```

```
  fclose(b0);
  fclose(b1);
  fclose(b2);
  fclose(b3);
  fclose(b4);
  fclose(mu);
  fclose(sig2);
  printf("Beta0 AR: ");
  for(i=0; i<teams; i++){
    printf("%lf ",beta0_ar[i]/NSIM);
  }
  printf("\n");

  printf("Beta1 AR: ");
  for(i=0; i<teams; i++){
    printf("%lf ",beta1_ar[i]/NSIM);
  }
  printf("\n");

  printf("Beta2 AR: ");
  for(i=0; i<teams; i++){
    printf("%lf ",beta2_ar[i]/NSIM);
  }
  printf("\n");

  printf("Beta3 AR: ");
  for(i=0; i<teams; i++){
    printf("%lf ",beta3_ar[i]/NSIM);
  }
  printf("\n");

  printf("Beta4 AR: ");
  for(i=0; i<teams; i++){
    printf("%lf ",beta4_ar[i]/NSIM);
  }
  printf("\n");
  return 0;
}

double g(double x, int jj, int beta_int,double betaAll[][teams],
double mu, double sig2){
  int i,j;
  double beta_temp[Nbetas];
  double final;
```

```
  double Lik=0;
  for(j=0; j<teams; j++){
      if(j==jj){
for(i=0; i<Nbetas; i++){
  if(i==beta_int){
    beta_temp[i]=x;}else{
    beta_temp[i]=betaAll[i][j];}
}
      }else{for(i=0; i<Nbetas; i++){
  beta_temp[i]=betaAll[i][j];
}}
      switch (j)
{
case 0 : Lik+=likelihood(Y1,One_min_Y1,x1,beta_temp,NROW1);
  /*printf("%lf\n",Lik);*/
  break;
case 1 : Lik+=likelihood(Y2,One_min_Y2,x2,beta_temp,NROW2);
  /*printf("%lf\n",Lik);*/
  break;
case 2 : Lik+=likelihood(Y3,One_min_Y3,x3,beta_temp,NROW3);
  /*printf("%lf\n",Lik);*/
  break;
case 3 : Lik+=likelihood(Y4,One_min_Y4,x4,beta_temp,NROW4);
  /* printf("%lf\n",Lik);*/
  break;
case 4 : Lik+=likelihood(Y5,One_min_Y5,x5,beta_temp,NROW5);
  /*printf("%lf\n",Lik);*/
  break;
case 5 : Lik+=likelihood(Y6,One_min_Y6,x6,beta_temp,NROW6);
  /* printf("%lf\n",Lik);*/
  break;
case 6 : Lik+=likelihood(Y7,One_min_Y7,x7,beta_temp,NROW7);
  /*printf("%lf\n",Lik);*/
  break;
case 7 : Lik+=likelihood(Y8,One_min_Y8,x8,beta_temp,NROW8);
  /*printf("%lf\n",Lik);*/
  break;
case 8 : Lik+=likelihood(Y9,One_min_Y9,x9,beta_temp,NROW9);
  /*printf("%lf\n",Lik);*/
  break;
case 9 : Lik+=likelihood(Y10,One_min_Y10,x10,beta_temp,NROW10);
  /*printf("%lf\n",Lik);*/
  break;
case 10 : Lik+=likelihood(Y11,One_min_Y11,x11,beta_temp,NROW11);
  /*printf("%lf\n",Lik);*/
```

```
  break;
default : printf( "Not an available team \n");
  break;
}
  }
  final=Lik-((x-mu)*(x-mu))/(2*sig2);
  return final;
  }

void applymean(double BetaMat[][teams], double meanBeta[Nbetas],
int teams){
  double beta_temp[teams];
  int i,j;
  for(j=0; j<Nbetas; j++){
    for(i=0; i<teams; i++){
      beta_temp[i]=BetaMat[j][i];
    }
    meanBeta[j]=gsl_stats_mean(beta_temp,1,teams);
  }
}

double sum(double x[],int n){
  int i;
  double res=0;
  for (i=0;i<n; i++){
    res+=x[i];
  }
  return res;
}

double likelihood(double y[], double OneMinusY[],double x[][Nbetas],
double beta[],int N){
  double Xbeta[N],yXbeta[N],expXBeta[N],L1expXB[N];
  double res;
  int i;
  double a,b,c,negY[N],OnePlusXBeta[N];
  mvmult(x,beta,Xbeta,N,Nbetas);
  exponentiate(Xbeta,expXBeta,N);
  addconst(1.0,expXBeta,OnePlusXBeta,N);
  logme(OnePlusXBeta,L1expXB,N);
  b=sumdot(y,L1expXB,N);
  a=sumdot(y,Xbeta,N);
  c=sumdot(OneMinusY,L1expXB,N);
  res=a-b-c;
  return res;
```

```
}

void exponentiate(double x[], double final[], int N){
  int i;
  for(i=0;i<N;i++){
    final[i]=exp(x[i]);
  }
}

void logme(double x[], double final[], int N){
  int i;
  for(i=0;i<N;i++){
    final[i]=log(x[i]);
  }
}

double sumdot(double x[],double y[],int length){
  int i;
  double res=0.0;
  for(i=0;i<length;i++){
    res+=x[i]*y[i];
  }
  return(res);
}

void mvmult(double a[][5],double b[], double res[],int ar, int ac){
  int i, j,k;
  for(i=0; i<ar;i++){
      res[i]=0.0;
      for(k=0;k<ac;k++){
res[i]+=a[i][k]*b[k];
      }
    }
  }

void addconst(double y,double x[], double final[],int N){
  int i;
  for(i=0;i<N;i++){
    final[i]=x[i]+y;
  }
}
```

B.3.1 Calculating Goodness of Fit and Importance Scores

```
beta0<-read.table("/Users/gradstudent/Documents/Master's Project/Bayes
```

```
 Project/Results/beta0.txt",header=FALSE)
beta1<-read.table("/Users/gradstudent/Documents/Master's Project/Bayes
 Project/Results/beta1.txt",header=FALSE)

beta2<-read.table("/Users/gradstudent/Documents/Master's Project/Bayes
 Project/Results/beta2.txt",header=FALSE)

beta3<-read.table("/Users/gradstudent/Documents/Master's Project/Bayes
 Project/Results/beta3.txt",header=FALSE)

beta4<-read.table("/Users/gradstudent/Documents/Master's Project/Bayes
 Project/Results/beta4.txt",header=FALSE)
burn<-1001
beta0<-beta0[-c(1:burn),]
beta1<-beta1[-c(1:burn),]
beta2<-beta2[-c(1:burn),]
beta3<-beta3[-c(1:burn),]
beta4<-beta4[-c(1:burn),]

Beta.team1<-cbind(beta0[,1],beta1[,1],beta2[,1],beta3[,1],beta4[,1])


####Scan in the data
x1<-read.table("/Users/gradstudent/Documents/Master's Project/Bayes
 Project/Opt/x1.txt",header=FALSE)

x1<-as.matrix(x1)

X<-list(x1)
Beta<-list(Beta.team1)

y1<-read.csv("/Users/gradstudent/Documents/Master's Project/Bayes
 Project/Opt/y1.txt",header=FALSE)

Y<-rbind(y1)
tabY<-table(Y)

BX2<-rep(0,nrow(beta0))
nteams<-1
for(i in 1:nrow(beta0)){
tabs<-rep(0,2)
for(j in 1:nteams){

vals<-rbern(nrow(X[[j]]),exp(X[[j]]%*%as.matrix(Beta[[j]][i,],
ncol=1))/(1+exp(X[[j]]%*%as.matrix(Beta[[j]][i,],ncol=1))))
```

```
tabs<-tabs+table(vals)
}
BX2[i]<-sum(((tabs-tabY)^2/tabY))
}

#I wrote the BX2 to a file called BayesChi2.txt
mean(pchisq(BX2,1,lower=TRUE)<0.05)
#[1] 0.04147

#Importance Scores
beta1<-read.table("/Users/gradstudent/Documents/Master's Project/
Bayes Project/Results/beta1.txt",header=FALSE)
beta1.good<-beta1[-c(1:1001),]
apply(beta1.good,2,mean)/sqrt(apply(beta1.good,2,var))

beta2<-read.table("/Users/gradstudent/Documents/Master's Project/
Bayes Project/Results/beta2.txt",header=FALSE)
beta2.good<-beta2[-c(1:1001),]
apply(beta2.good,2,mean)/sqrt(apply(beta2.good,2,var))

beta3<-read.table("/Users/gradstudent/Documents/Master's Project/
Bayes Project/Results/beta3.txt",header=FALSE)
beta3.good<-beta3[-c(1:1001),]
apply(beta3.good,2,mean)/sqrt(apply(beta3.good,2,var))

beta4<-read.table("/Users/gradstudent/Documents/Master's Project/
Bayes Project/Results/beta4.txt",header=FALSE)
beta4.good<-beta4[-c(1:1001),]
apply(beta4.good,2,mean)/sqrt(apply(beta4.good,2,var))
```