2008-03-11

Skill Evaluation in Women's Volleyball

Lindsay Walker Florence
Brigham Young University - Provo

SKILL EVALUATION IN WOMEN'S VOLLEYBALL

by

Lindsay W. Florence

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 2008

BRIGHAM YOUNG UNIVERSITY

GRADUATE COMMITTEE APPROVAL

of a project submitted by

Lindsay W. Florence

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	William F. Christensen

BRIGHAM YOUNG UNIVERSITY

As chair of the candidate's graduate committee, I have read the project of Lindsay W. Florence 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

SKILL EVALUATION IN WOMEN'S VOLLEYBALL

Lindsay W. Florence

Department of Statistics

Master of Science

The Brigham Young University Women's Volleyball Team recorded and rated all skills (pass, set, attack, etc.) and recorded rally outcomes (point for BYU, rally continues, point for opponent) for the entire 2006 home volleyball season. Only sequences of events occurring on BYU's side of the net were considered. Events followed one of these general patterns: serve-outcome, pass-set-attack-outcome, or block-dig-set-attack-outcome. These sequences of events were assumed to be first-order Markov chains where the quality of each contact depended only explicitly on the quality of the previous contact but not on contacts further removed in the sequence. We represented these sequences in an extensive matrix of transition probabilities where the elements of the matrix were the probabilities of moving from one state to another. The count matrix consisted of the number of times play moved from one transition state to another during the season. Data in the count matrix were assumed to have a multinomial distribution. A Dirichlet prior was formulated for each row of the count matrix, so posterior estimates of the transition probabilities were then available using Gibbs sampling. The different paths in the transition probability

matrix were followed through the possible sequences of events at each step of the MCMC process to compute the posterior probability density that a perfect pass results in a point, a perfect set results in a point, and so forth. These posterior probability densities are used to address questions about skill performance in BYU women's volleyball.

ACKNOWLEDGEMENTS

I would like to thank those individuals who made it possible for me to accomplish this project: Dr. Gilbert Fellingham for his patience and guidance through this master's project and the article we published; Nina Mortensen and Sammi Stuart for the countless hours they spent gathering the data and verifying it for accuracy; the BYU Statistics Department for encouragement to always do my best and the needed financial assistance while in graduate school; my husband, Trevor, for his genuine interest in my research and loving patience; and especially my Heavenly Father for giving me the necessary strength this year to accomplish as much as I have.

CONTENTS

TABLES

Table

FIGURES

Figure

1. INTRODUCTION

Statistical analysis in the field of sports could provide valuable information to athletes and coaching staffs. With appropriate analysis of relevant data, practice sessions could focus on the most important skills. Players could be grouped together to form optimal teams. Superior players could receive the recognition they deserve. Coaches could obtain feedback necessary to make immediate changes during the game or throughout the season (Byra and Scott 1983). A quantitative analysis would be beneficial for all sports at any level.

However, there has not been a great deal of quantitative research published on volleyball skills (Daniel and Hughes 2003). This is unfortunate because in 2004 there were an estimated 200 million players worldwide (Verhagen et al. 2004). With the continually growing popularity of the sport, researchers should do everything possible to understand the techniques of the game (Vojik 1980). This would improve the quality of the game in general and allow for higher level of play and satisfaction.

The Brigham Young University Women's Volleyball Team, a Division I intercollegiate team, used a notational analysis system to measure their skill performance during the 2006 home volleyball season; every serve, pass, attack, and dig was recorded and graded in real time, while sets were graded after viewing the matches on film. Every touch made by the team was graded on a scale as fine as 0–5 points in order to quantify how well the skill was performed. The resulting data set consisted of 13 matches and over 7,300 touches of the ball for the BYU team.

The purpose of this project was to calculate various unconditional probabilities of certain skills leading to either a point for BYU, continuation of the rally, or a point for the opponent. The sequences of hits were assumed to follow a first-order Markov chain, where the quality of each hit depended only on the quality of the previous contact and not explicity on contacts further removed in the sequence. A count matrix was constructed which consisted of the number of times play moved from one transition state to another during the season. A multinomial likelihood distribution was assumed for each row in the count matrix with a Dirichlet prior distribution for the associated probabilities. The posterior distribution for the transition probabilities in each row is then proportional to the product of the likelihood and prior distributions. Gibbs sampling was implemented to calculate the posterior distributions of the probabilities of moving from one state to another. The mean of the posterior distribution of the probability was calculated as a point estimate to insert into the transition matrix. The unconditional probabilities associated with performing a particular skill at various levels were then estimated from the transition matrix at each iteration of the sampling process. The posterior distributions of the unconditional probabilities were then available to quantify the uncertainty in the probability point estimates.

The outline of this project follows. Chapter 2 reviews the literature associated with previous notational systems used in volleyball, the properties of Markov chains, estimating transition probabilities, the Bayesian methods used in this analysis, and estimating transition probabilities using Bayesian methods. Chapter 3 consists of the paper submitted to the Journal of Quantitative Analysis in Sports. Appendix A discusses combining certain rows and columns of the original count matrix in order to provide better estimates of the desired probabilities. Appendix B gives a small portion of the raw data provided by the software *Data Volley* (Data Project, Salerno, Italy, release 2.1.9). Appendix C contains the R code used to clean the data and perform the analyses.

2. REVIEW OF LITERATURE

This chapter is divided into four sections. Section 2.1 describes some of the previous notational analysis systems used in volleyball and statistical analyses performed on volleyball skills. Section 2.2 discusses some properties of Markov chains and methods for estimating transition probabilities. Section 2.3 gives a brief overview of the Bayesian methods used to calculate the posterior distributions for each transition probability. Section 2.4 discusses some previous literature estimating transition probabilities in a Bayesian framework.

2.1 Previous Research in Volleyball

In order to fully comprehend a sport team's performance, it seems reasonable that the skills used must be recorded, graded, and analyzed quantitatively. According to Daniel and Hughes (2003), there has not been a considerable amount of quantitative analyses published concerning the performance of volleyball skills. However, various notational analysis systems have been developed for the purpose of analyzing volleyball skills (Coleman et al. 1971; Coleman 1975; Sawula 1977; Lirdla 1980; Vojik 1980; Rose 1983; Eom and Schutz 1992; Zetou et al. 2007). Most notational analysis systems grade skills according to the outcome of a rally or the opponent's performance (Mortensen 2007). For example, serves are graded based on the performance of the opponent's pass. Attacks are graded according to how well the opposing team responds to the attack. Setting is unique in the game of volleyball because it does not have a direct influence on the opponent's performance (Coleman 1975). Thus, setting is more difficult to define a grading system based solely on the quality of the contact.

Daniel and Hughes (2003) performed an analysis on the differences between elite (international teams) and non-elite (university teams) volleyball players. They used chi-square tests to compare the two groups and found that the elite players performed significantly better in serving and passing. For the elite players, they also found the quality of the set depended on the quality of the pass, and the quality of the attack depended on the quality of the set. This was also noticeable in the non-elite players, but the relationship was not as strong.

Eom and Schutz (1992) analyzed eight national men's volleyball teams that participated in the 1987 Federation of International Volleyball Korean Cup. The purpose of their analysis was to determine which skills are the best predictors of a successful team. Using discriminant analysis they found the block, the spike in the attack process (responding to an opponent's serve), and the spike in the counterattack process (responding to an opponent's attack) to be the most significant skills in determining whether a team will be successful. They also analyzed the differences in the set-spike sequence in the attack process and the counterattack process. Using multivariate analysis of variance, they found the attack and counterattack processes to be significantly different. Thus, they advised treating set-attack sequences as separate events when coming from either a dig or a pass.

Another paper by Zetou et al. (2007) analyzed the skills performed in 38 Men's Olympic Volleyball games. They performed separate discriminant analyses for passing and attack from reception (the first attack of the rally) in order to determine the most significant skills contributing to scoring points. They used stepwise methods for selecting variables and estimated the classification based on the jackknife (leave-one-out) approach. In the analysis involving passing, they found that the individual receiving the serve should either make the best pass possible so the setter can set an up-tempo attack or make a good pass so the setter can set a high set to an outside hitter in zone 4 or 2. In the analysis based on attack from reception, the "ace-point," or point directly following the attack, was the most important factor in predicting the win of the rally.

This project builds on this previous work and adds an extra dimension. Although notational systems have long been used to quantify volleyball performance in some dimensions, there has never been an extensive attempt to grade setting precision. Even though setting was not incorporated into their analysis, Zetou et al. (2007) discussed the need to evaluate setting due to its direct influence on attacks. Currently, the only grades recorded by the NCAA for volleyball are assists to the hitter and setting errors. Using their grading system, it is possible for a setter to have a perfect set that is not counted as an assist if the hitter performed poorly. The setter could also receive an assist if a hitter recovered from a poor set resulting in a positive outcome. The data set used for this analysis was produced by and for the BYU Women's Volleyball Team, and included an independent rating of every skill performed by team members during the 2006 home season. By grading setting independently of the attack and outcome, the natural association between the performance of one skill and the performance of subsequent skills can be examined.

2.2 Markov Chains

2.2.1 Properties of Markov Chains

Because volleyball skills are performed in a fairly rigid time sequence pattern (pass-set-attack, etc.), it seemed natural to treat these patterns as Markov chains. That is, the problem was approached as estimating the probability of transitioning from one state to another while the ball was on BYU's side of the net. A Markov chain is a sequence of random variables in which the current state only depends explicitly upon the previous state. This can also be defined as

$$
\begin{aligned}
Pr(X_{n+1} = x_{n+1} | X_0 = x_0, X_1 = x_1, \ldots, X_n = x_n) \\
= Pr(X_{n+1} = x_{n+1} | X_n = x_n),
\end{aligned}
\tag{2.1}
$$

where X represents a state in the Markov chain sequence and n represents the time at which the state occurred (Stewart 1994). All the possible probabilities of moving from one state to another are comprised in the transition probability matrix.

Properties of Markov chains make it possible to classify every state in the transition probability matrix. In a *recurrent* state, the probability of eventually returning to that state is 1. In other words, it is possible to return to that state multiple times. A *transient* state has a probability less than 1 of returning to the given state. If it is possible to eventually arrive at any state in the transition probability matrix given the current state, the Markov chain is said to be *irreducible* (Ross 1996). A state that can transition to the same state in one step is known to be *aperiodic*. An irreducible and aperiodic Markov chain where the states are positive recurrent produces a *stationary distribution*. A probability distribution $\mathbf{z}$ for a Markov chain, where $\mathbf{z}$ is a vector of elements containing the probabilities of transitioning from state i to another state j, is defined as a stationary distribution if and only if $\mathbf{zP} = \mathbf{z}$, where $\mathbf{P}$ is the transition probability matrix (Stewart 1994). Knowing a transition probability matrix will converge to the stationary distribution is essential for Markov chain Monte Carlo methods to be successful (see Section 2.3 for more information on MCMC).

2.2.2 Estimating Transition Probabilities

In the last half century, different methods have been utilized to estimate the probabilities in a transition matrix. In earlier analyses, such as Miller (1952) and Telser (1963), least squares estimators were implemented when only sample proportions from aggregate time series data were available. One dilemma with least squares estimates, though, is that transition probability estimates could be negative. To compensate for this problem, Judge and Takayama (1966), Theil and Rey (1966), and Lee et al. (1968) discussed using restricted least squares estimators based on a quadratic programming iteration method. Another problem with least squares estimators for proportional data is heteroscedasticity (Theil and Rey 1966; Lee et al. 1969). Madansky (1959) used weighted least squares estimators to try to correct this problem. In addition, Theil and Rey (1966) proposed using weighted restricted least squares estimators.

Lee et al. (1969) performed a simulation study to compare various least squares estimators when using sample proportions from aggregate time series data. They simulated 50 data sets from a four-state transition matrix using sample sizes of 25, 50, 75, and 100. They found that weighted restricted least squares estimators performed better over unweighted restricted least squares and unweighted unrestricted least squares. They also found that the restricted least squares estimator was far superior to the unrestricted least squares estimator. These results were based on statistical tests including chi-square and Kolmogorov-Smirnov goodness-of-fit test, Kendall's coefficient of concordance, and Wilcoxon matched-pairs signed-rank test.

Along with the various forms of least squares estimators, maximum likelihood estimators have continually been used throughout the last half century when individual measurements are available as opposed to aggregate proportions (Anderson and Goodman 1957; Duncan and Lin 1972; Craig and Sendi 2002). With the advancement of computer capabilities, Bayesian models have also become a common method to estimate transition probabilities (Lee et al. 1968; Boender and Rinnooy-Kan 1983; Fahrmeir 1992; Assoudou and Essebbar 2003).

A paper by Lee et al. (1968) compared different methods of estimating transition probabilities including least squares, weighted least squares, maximum likelihood, and

Bayesian models using a multinomial likelihood and a Dirichlet prior distribution. They simulated 50 data sets for a first-order stationary Markov chain with four states using sample sizes of 25, 50, and 100. To gauge the performance of the different estimators, they calculated the mean square error for each transition probability. They also calculated an overall mean square error for each estimated transition matrix by summing the mean square errors associated with the transition probabilities in the matrix. They found that the Bayesian estimators performed better than maximum likelihood, least squares, and weighted least squares estimators. These results were based on the mean square error, absolute value of the error, and various nonparametric tests including Wilcoxon's matched-pairs signed-rank test, Kendall's coefficient of concordance, and Kolmogorov-Smirnov's goodness-of-fit test.

2.3 Bayesian Methods

Bayesian models are based on Bayes' Theorem, which states that

$$\pi(\boldsymbol{\theta}|y) = \frac{f(y|\boldsymbol{\theta})\pi(\boldsymbol{\theta})}{\int f(y|\boldsymbol{\theta})\pi(\boldsymbol{\theta})d\boldsymbol{\theta}}, \tag{2.2}$$

where $\pi(\boldsymbol{\theta}|y)$ is the posterior distribution, $f(y|\boldsymbol{\theta})$ is the likelihood, and $p(\boldsymbol{\theta})$ is the prior distribution. The denominator in Equation 2.2 is also known as the normalizing constant. Thus, the posterior distribution is proportional to the product of the likelihood and the prior distribution. The parameters of the prior distribution are based on *a priori* knowledge or belief.

The definition of a conjugate prior distribution is if F is a class of sampling distributions $p(y|\boldsymbol{\theta})$ and P is a class of prior distributions for $\boldsymbol{\theta}$, then the class P is conjugate for F if

$$p(\boldsymbol{\theta}|y) \in P \text{ for all } p(\cdot|\boldsymbol{\theta}) \in F \text{ and } p(\cdot) \in P$$

(Gelman et al. 2004). Because it is possible to always choose P to be conjugate according to this definition, natural conjugate prior distributions are a subject of interest. A natural conjugate prior distribution implies that the prior distribution has the same functional form as the likelihood (Gelman et al. 2004). Natural conjugate prior distributions make it possible to obtain draws directly from the posterior distribution using Markov chain Monte Carlo (MCMC) simulation and a Gibbs sampler. Another advantage to using a conjugate prior distribution is the parameters of the posterior distribution are easier to interpret.

Markov chain Monte Carlo simulation is the most common method used to sample from the posterior distribution. The goal of MCMC is to estimate the parameters, $\boldsymbol{\theta}$, and obtain draws from the posterior distribution, $p(\boldsymbol{\theta}|y)$. This method is useful when it is not possible to sample $\boldsymbol{\theta}$ directly from $p(\boldsymbol{\theta}|y)$. Each distribution of draws is updated from the previous iteration forming the Markov chain. After enough iterations, the distribution will converge to the unique stationary distribution and arrive at the posterior distribution (Gelman et al. 2004).

Because the Dirichlet distribution is a natural conjugate prior distribution for the multinomial likelihood distribution, draws can be obtained directly from the posterior distribution using a Gibbs sampler. Gibbs sampling is an iterative process which samples from each of the conditional posterior distributions instead of integrating over the entire joint posterior distribution. Gibbs sampling can also be considered a special case of the Metropolis-Hastings algorithm where every iteration is accepted (Gelman et al. 2004). The steps of a Gibbs sampler are found in Ross (1996) and listed as follows:

(1) Let $\boldsymbol{\theta}^0 = (\theta_1^0, \theta_2^0, \ldots, \theta_n^0)$ be any vector where the values are located inside the parameter space.

(2) Let $i = 1$.

(3) Randomly generate an observation, θ_1^1, from $f(\theta_1|\theta_2^{i-1}, \ldots, \theta_n^{i-1})$, which is the full conditional distribution of θ_1 given the most recent values of the other parameters.

(4) Randomly generate an observation, θ_2^1, from the conditional distribution $f(\theta_2|\theta_1^i, \theta_3^{i-1}, \ldots, \theta_n^{i-1})$.

(5) Continue until the observation, θ_n^1, has been generated from the conditional distribution $f(\theta_n|\theta_1^i, \theta_2^i, \ldots, \theta_{n-1}^i)$.

(6) Store the vector of generated observations into $\boldsymbol{\theta}^1 = (\theta_1^1, \theta_2^1, \ldots, \theta_n^1)$.

(7) Let $i = i + 1$.

(8) Repeat steps 3 through 7 N times.

As the limit of N goes to infinity, $\boldsymbol{\theta}^N$ converges to the joint posterior distribution $p(\boldsymbol{\theta}|y)$, assuming the Markov chain is irreducible and aperiodic. A more detailed explanation of the Gibbs sampler can be found in Casella and George (1992).

2.4 Bayesian Estimation of Markov Processes

Most previous work using Bayesian models to estimate transition probabilities assumed a multinomial likelihood distribution and Dirichlet prior distribution (Lee et al. 1968; Satia and Lave 1973; Ezzati 1974; Meshkani and Billard 1992; McKeigue et al. 2000; Assoudou and Essebbar 2003; Ozekici and Soyer 2003; Zhao et al. 2005). Other models, such as those used by Cargnoni et al. (1997) and Assoudou and Essebbar (2003), assumed different prior distributions including the normal distribution and Jeffreys' prior distribution, respectively.

Assoudou and Essebbar (2003) performed a simulation study on estimating transition probabilities comparing maximum likelihood estimators with Bayesian estimators using the Dirichlet prior distribution and Jeffreys' prior distribution. They simulated 20 data sets using a sample size of $n = 21$ for a two-state model and ten data sets with a sample size of $n = 61$ for a three-state model. They found that both Bayesian estimators performed better than maximum likelihood estimators and had a lower mean square error for the two- and three-state models they simulated. In comparing the model using Dirichlet and Jeffreys' noninformative prior distributions, the Jeffreys' prior distribution gave slightly better estimates than the Dirichlet distribution. This may be influenced by the relatively small simulated sample sizes and number of data sets generated.

Based on the work performed by Anderson and Goodman (1957) and Lee et al. (1968), Ezzati (1974) analyzed aggregate time series data for home heating units using both methods of maximum likelihood and Bayesian estimation. For their Bayesian model, they used a multinomial likelihood and a Dirichlet prior distribution. Their estimates were based on the posterior mean and variance. The purpose of their analysis was to forecast market shares of annual sales for home heating units including oil burners, gas burners, and electric heat. One concern with their model was assuming the transition probabilities remained constant over time. A change in consumer behavior would alter the transition probabilities and make it unlikely to accurately forecast future observations. In order to alleviate this problem, they incorporated various marketing variables such as income or price elasticity of alternative heating units into the prior distribution. They found that their models performed well when compared to actual historical data.

To calculate Bayesian point estimates of the transition probabilities, several earlier methods used the posterior mean or mode (Lee et al. 1968; Boender and Rinnooy-Kan 1983; Fahrmeir 1992; McKeigue et al. 2000; Ozekici and Soyer 2003). DeGroot (1970) showed that the posterior expectation is the optimal Bayesian estimator with respect to the quadratic loss function. The quadratic loss function L is defined as

$$L(w, d) = a(w - d)^2, \tag{2.3}$$

where a is a constant, w is the parameter of interest, and d is the estimate of w.

3. PAPER FOR THE JOURNAL OF QUANTITATIVE ANALYSIS IN SPORTS

3.1 Introduction

The Brigham Young University Women's Volleyball Team, a Division I intercollegiate team, used a notational analysis system to measure skill performance during the 2006 home volleyball season; every serve, pass, attack, and dig was recorded and graded in real time, while sets were graded after viewing the matches on film. Every touch made by the team was graded on a scale ranging as fine as 0–5 points in order to quantify how well the skill was performed.

We assumed the sequences of hits followed a first-order Markov chain, where the quality of each hit depended only on the quality of the previous contact and not explicitly on contacts further removed in the sequence. We assumed a multinomial likelihood distribution for each row in the count matrix and a Dirichlet prior distribution for the associated probabilities. The count matrix consisted of the number of times play moved from one transition state to another during the season. The posterior distribution for the probabilities in each row is then proportional to the product of the likelihood and prior distributions. Gibbs sampling was implemented to calculate the posterior distributions of the probabilities of moving from one state to another. We used the mean of the posterior distribution of the probability as a point estimate to insert into the transition matrix. The transition probability matrix can then be used to estimate probabilities of various sequences of events. We used the transition probability matrix to estimate the unconditional probabilities associated with performing a particular skill at various levels.

Section 3.2 examines previous work on volleyball analysis and estimating transition probabilities. Section 3.3 discusses the data and the notational grading system used for the BYU Women's Volleyball Team. Section 3.4 discusses the transitional probability matrix, our Bayesian model, and the methods used to calculate posterior distributions of unconditional probabilities for a certain skill resulting in a point for BYU, continuation of rally, or a point for the opponent. Section 3.5 presents the resulting point estimates and posterior distributions for the unconditional probabilities. Section 3.6 discusses ways the methodology might be used to improve play.

3.2 Previous Literature

In order to fully comprehend a sport team's performance, the skills used must be recorded, graded, and analyzed quantitatively. According to Daniel and Hughes (2003), there has not been a considerable amount of quantitative analyses published concerning the performance of volleyball skills. However, various notational analysis systems have been developed for the purpose of analyzing volleyball skills (Coleman et al. 1971; Coleman 1975; Sawula 1977; Lirdla 1980; Vojik 1980; Rose 1983; Eom and Schutz 1992; Zetou et al. 2007). This paper builds on this previous work and adds an extra dimension: although notational systems have long been used to quantify volleyball performance in some dimensions, there has never been an extensive attempt to include the grading of setting in the systems. By grading setting independently of the attack and outcome, the natural association between the performance of one skill and the performance of subsequent skills can be examined. The data set we used was produced by and for the BYU Women's Volleyball Team and included a rating of every skill performed by team members during the 2006 home season.

Because volleyball skills are performed in a fairly rigid time sequence pattern (pass-set-attack, etc.), it seemed natural to treat these patterns as Markov chains. That is, we approached the problem as one of estimating the probability of transitioning from one state to another while the ball was on BYU's side of the net. Common methods used in estimating transition probabilities have included maximum likelihood (Anderson and Goodman 1957; Duncan and Lin 1972; Craig and Sendi 2002), Bayesian methods (Lee et al. 1968; Boender and Rinnooy-Kan 1983; Fahrmeir 1992; Assoudou and Essebbar 2003), least squares (Miller 1952; Telser 1963), weighted least squares (Madansky 1959), restricted least squares (Theil and Rey 1966; Lee et al. 1968), and weighted restricted least squares (Theil and Rey 1966). Lee et al. (1968) compared different methods of estimating transition probabilities including least squares, weighted least squares, maximum likelihood, and Bayesian models. They found that Bayesian estimators performed better than maximum likelihood, least squares, and weighted least squares estimators. These results were based on the mean square error and absolute value of the error from various nonparametric tests. Assoudou and Essebbar (2003) also found that Bayesian estimators performed better than maximum likelihood and had a lower mean square error for two- and three-state models.

Most work using Bayesian models to estimate transition probabilities has assumed a multinomial likelihood distribution and a Dirichlet prior distribution (Lee et al. 1968; Satia and Lave 1973; Ezzati 1974; Meshkani and Billard 1992; McKeigue et al. 2000; Ozekici and Soyer 2003; Zhao et al. 2005). The models used by Cargnoni et al. (1997) and Assoudou and Essebbar (2003) assumed different prior distributions including the normal distribution and Jeffreys' prior distribution, respectively. To calculate Bayesian point estimates of the transition probabilities, several earlier methods used the posterior mean or mode (Lee et al. 1968; Boender and Rinnooy-Kan 1983; Fahrmeir 1992; McKeigue et al. 2000). DeGroot (1970) showed that the posterior expectation is the optimal Bayesian estimator with respect to the quadratic loss function.

3.3 The Data

The data were recorded into a program called Data Volley (Data Project, Salerno, Italy, release 2.1.9). The grading system was developed based on the number of possible codes Data Volley was capable of handling. Serves were graded on a six-point (0–5) scale, passes on a five-point (0–4) scale, and attacks by position on the court (middle, right side, left side, back row) and outcome (kill, rally continuation, error, block). We evaluated sets according to three variables: distance from the net (0–3 feet, 3–5 feet, etc.), height of the set (high and low), and position of the set in relation to the hitter (inside and outside). Digs and blocks were also noted in the data.

A trained member of the women's volleyball coaching staff graded and recorded in real time every serve, pass, dig, and attack performed by BYU for the 13 home matches during the 2006 season. A default code was inserted for sets, so these could be graded at a later time while viewing the game on film. To grade the sets, the matches were filmed by two cameras observing different angles of the court at the same time. One camera recorded the entire court from behind the end line of the BYU women's team. The other camera was parallel to and approximately five feet away from the net, showing only BYU's side of the court. Questionable sequences found in the data were also verified by viewing the sequences on film. The hits recorded for the opposing team included serves and attacks. This allowed us to track when the ball had crossed the net. The final data set consisted of over 7,300 touches of the ball for the BYU team.

Considerable work was necessary before the data were ready to analyze. The data set contained many unnecessary codes that had to be removed. The information in the data that was necessary for the analysis included the number of the player who made contact with the ball, the skill type and skill grade of the contact, and when the game ended. The team that contacted the ball could be determined by looking at the player's number, which was coded so BYU numbers were less than 50 and opposing team numbers were greater than 50. Although the score was inserted by the person coding the data it was often inaccurate. To alleviate this problem, the score was determined at the conclusion of each rally by identifying the next team to serve. The outcome for the final rally of each game was determined by the final score.

Since only touches on BYU's side of the net were considered, continuation of the rally was determined by observing if the ball returned to BYU's side of the net during a rally. However, because the person recording the data was less interested in the opponent than BYU, sometimes there were no hits recorded for the opposing team in a specific sequence, making it appear as if the BYU team hit the ball more than three times in a row. Such sequences had to be located and corrected before the analysis could be performed.

3.4 Methods

Every time the ball was on BYU's side of the net, a sequence of events occurred that followed one of these patterns: serve-outcome, pass-set-attack-outcome, or dig-set-attack-outcome. The outcome was a point for BYU, a point for the opponent, or continuation of the rally. We assumed these sequences were first-order Markov chains. We represented these sequences in a matrix of transition probabilities where the elements in the matrix represented the probabilities of moving from one state to another (e.g., a four-point pass to a perfect set). Impossible sequences (e.g., a perfect pass to an ace serve) were constrained to have zero probability. Sequences that always occurred (e.g., an attack kill to a point for BYU) were assigned a probability of one. Because setting had two measurements recorded, we calculated the transition probability matrix including set distance from net, set placement, or both according to the measurement we wanted to analyze.

The transition matrix was comprised of 35 states when analyzing set distance, 37 states with set placement, and 55 states with combined set distance and placement. The states specified in the matrix were one opponent serve; six BYU float serves; six BYU jump serves; six passing types; five set distances, seven set placements, or 25 combinations of set distance and placement; seven attack types; one dig type; and three outcomes.

We used a Bayesian paradigm to model the unknown 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, $i = 1, \ldots, m$, in the count matrix, where k is the number of possible states that could occur next in the sequence of touches and m is the number of states in the transition matrix. The probability of moving from state i to another state j in the transition probability matrix is represented by π_{ij}, where $\sum_{j=1}^{k} \pi_{ij} = 1$. The data y_{ij} consist of the number of times play moved from state i to another state j during the season. The count matrix is comprised of all the y_{ij}'s.

We specified our prior probability densities in each row to be 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 each α_{ij} represents how often we expected the women's team to move from state i to state j relative to moving to a different state in the transition probability matrix. Prior counts were determined by one of the project designers, a former volleyball coach. To check for sensitivity to prior assumptions, we also ran an analysis with prior counts (α_{ij}) all equal to one. The results of the sensitivity analysis are discussed in Section 3.5.

We used Markov chain Monte Carlo methods to produce a 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 i in the transition matrix. We used the mean of the posterior distribution, $\frac{y_{ij}+\alpha_{ij}}{\sum_{j=1}^{k}(y_{ij}+\alpha_{ij})}$, for each of the π_{ij}'s as point estimates to insert in the transition probability matrix.

In addition to estimating the transition probability matrix, we calculated the unconditional probabilities of moving from a certain state (e.g., a perfect pass) to an outcome (e.g., a point for BYU). To obtain a point estimate for the unconditional probability, we considered all possible sequences of touches that could occur between the state and outcome in the transition probability matrix. For each sequence, we multiplied the corresponding probabilities in the transition matrix. Using the law of total probability, we summed the probability of each sequence to get the unconditional probability of going from a certain state to an outcome.

In order to understand how much variability existed in our unconditional probability point estimates, we calculated the distribution for each unconditional probability using Gibbs sampling. To efficiently draw values from the posterior distribution, we drew $x_1, x_2, \ldots, x_k$ from independent gamma distributions with shape parameters $y_{i1} + \alpha_{i1}, y_{i2} + \alpha_{i2}, \ldots, y_{ik} + \alpha_{ik}$ and common scale parameter and calculated $\pi_{ij} = x_j / \sum_{j=1}^{k} x_j$ (Gelman et al. 2004). We computed a draw of the unconditional probability using the current state of the transition probability matrix at each step of the MCMC process. The unconditional probability distributions were based on 100,000 realizations from each row's posterior distribution.

3.5 Results

We summarize results by focusing on the unconditional probabilities of moving from a certain skill to a rally outcome. Figure 3.1 shows the posterior distributions for the unconditional probability of the present rally sequence ending in a point for BYU following a pass of the given point rating. A 0-point pass is not shown because it can never end in a point for BYU. Similarly, Figures 3.2 and 3.3 show the posterior distributions for the unconditional probabilities of various set types leading to an immediate point for BYU. Finally, Figure 3.4 shows the posterior distributions of the probability of attacks by position on the court leading to a point for BYU. Point estimates for these probabilities, as well as the probability of the rally continuing and a point being scored by the opposition, are shown in Table 3.1.

We also performed a sensitivity analysis on the influence of the prior specification by setting the prior counts to 1 for every state where the transition probability was not constrained to be zero or one. The probabilities of passes with the various ratings leading to outcomes using these two prior distributions are shown in Figures 3.1 and 3.6 (posterior densities) and Tables 3.1 and 3.2 (point estimates). The outcomes are virtually indistinguishable for the two prior specifications. Thus, we have little reason to believe that the prior specifications we used had a marked influence on the posterior distributions. Similar differences were observed for all other unconditional probabilities.

3.6 Discussion

We recognize that this analysis is applicable only to BYU women's volleyball. Nonetheless, it is not unreasonable to look for generalizations that might be applicable to other teams. We also recognize that there are many types of questions that could be asked based on the analysis that we have presented. We consider just four areas that may provide useful information for coaches.

Many coaches rate passers based on their passing average. This system seems to be problematic based on our results. The passing average assumes that the difference between a 1-point pass and a 2-point pass is equivalent to that between a 2-point pass and a 3-point pass, etc. This is obviously not the case. For example, a player with a 3.0 passing average who earns that average with equal numbers of 2-point, 3-point, and 4-point passes would have a point probability that the rally would terminate with

Table 3.1: The unconditional probability point estimates for pass types, sets certain distances from the net, set placements, and attack positions resulting in the various outcomes.

Pass Types

Pass	Score Point	Continue Rally	Opponent Score
4-Point	0.505	0.260	0.235
3-Point	0.496	0.259	0.245
2-Point	0.489	0.262	0.249
1-Point	0.394	0.278	0.328

Sets Certain Distances from Net

Set Distance	Score Point	Continue Rally	Opponent Score
0–3 Feet	0.506	0.239	0.255
3–5 Feet	0.511	0.258	0.231
5–8 Feet	0.498	0.267	0.235
8–10+ Feet	0.426	0.293	0.281
Set not by Setter	0.456	0.290	0.254

Set Placements

Set Placement	Score Point	Continue Rally	Opponent Score
Perfect	0.509	0.259	0.232
Low and Inside	0.510	0.258	0.232
High and Outside	0.492	0.271	0.237
Low and Outside	0.495	0.260	0.245
High and Inside	0.472	0.284	0.244

Attack Positions

Attack	Score Point	Continue Rally	Opponent Score
Middle	0.530	0.243	0.227
Right Side	0.545	0.207	0.248
Left Side	0.495	0.283	0.222
Back Row	0.384	0.296	0.320

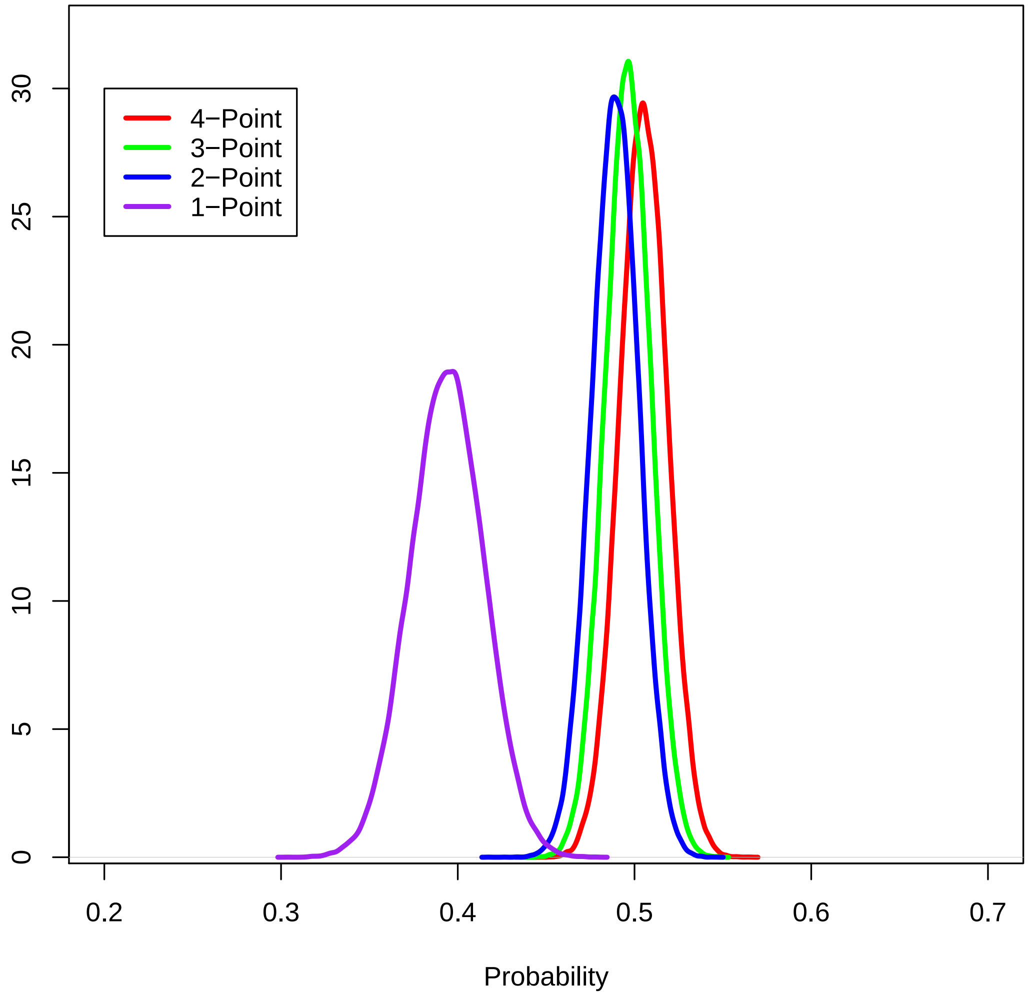

Figure 3.1: Posterior distributions for the unconditional probabilities of pass types leading to scoring a point.

Table 3.2: Probability point estimates for passing to certain outcomes when prior counts were all assumed to be 1.

Pass Score	BYU Score	Continue Rally	Opponent Score
4-Point	0.507	0.258	0.235
3-Point	0.500	0.257	0.243
2-Point	0.492	0.261	0.247
1-Point	0.380	0.279	0.341

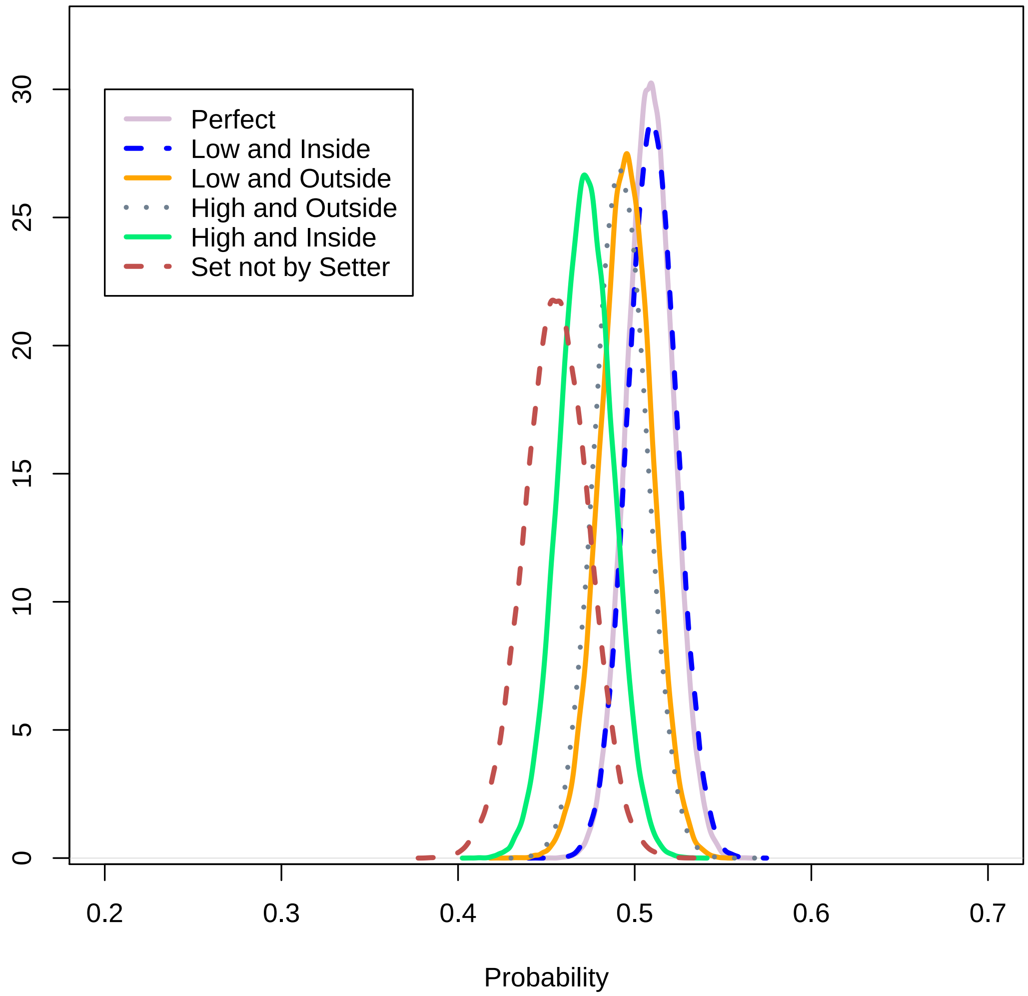

Figure 3.2: Posterior distributions for the unconditional probabilities of set placements leading to scoring a point.

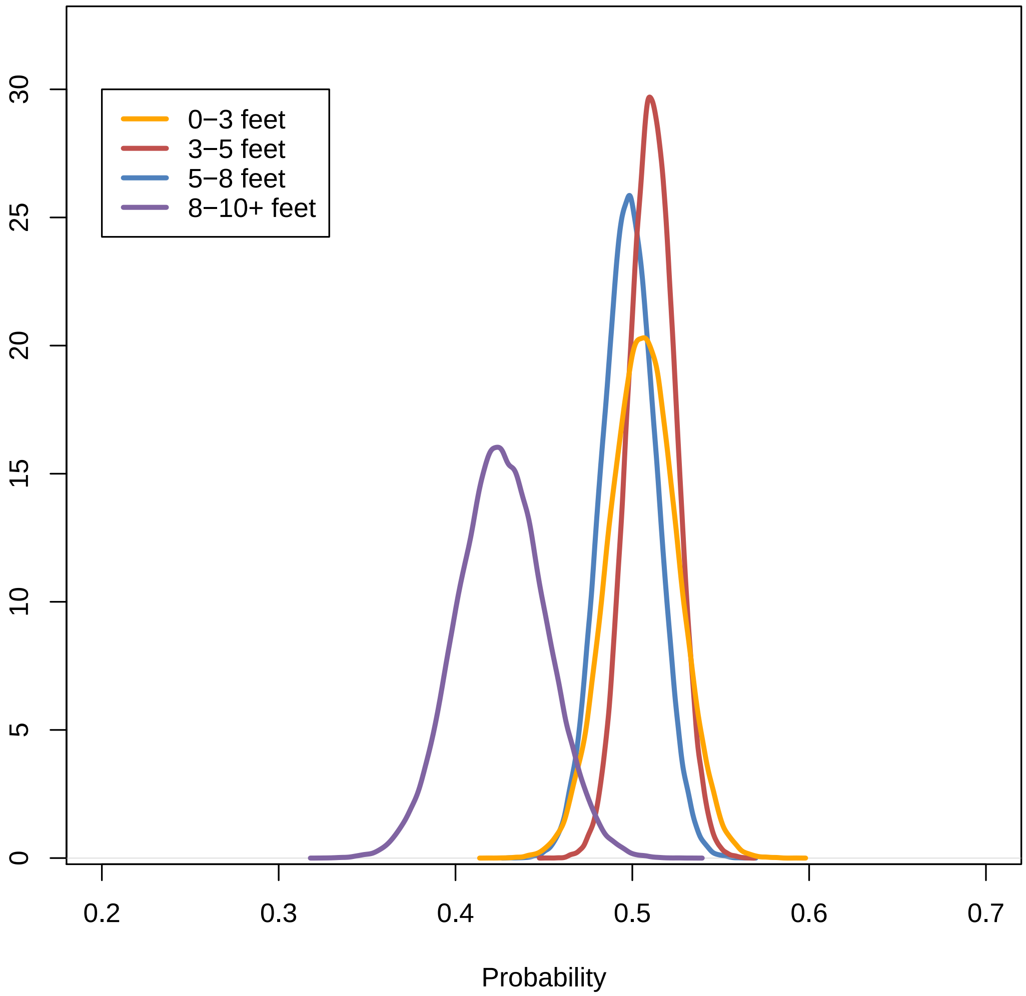

Figure 3.3: Posterior distributions for the unconditional probabilities of sets from various distances leading to scoring a point.

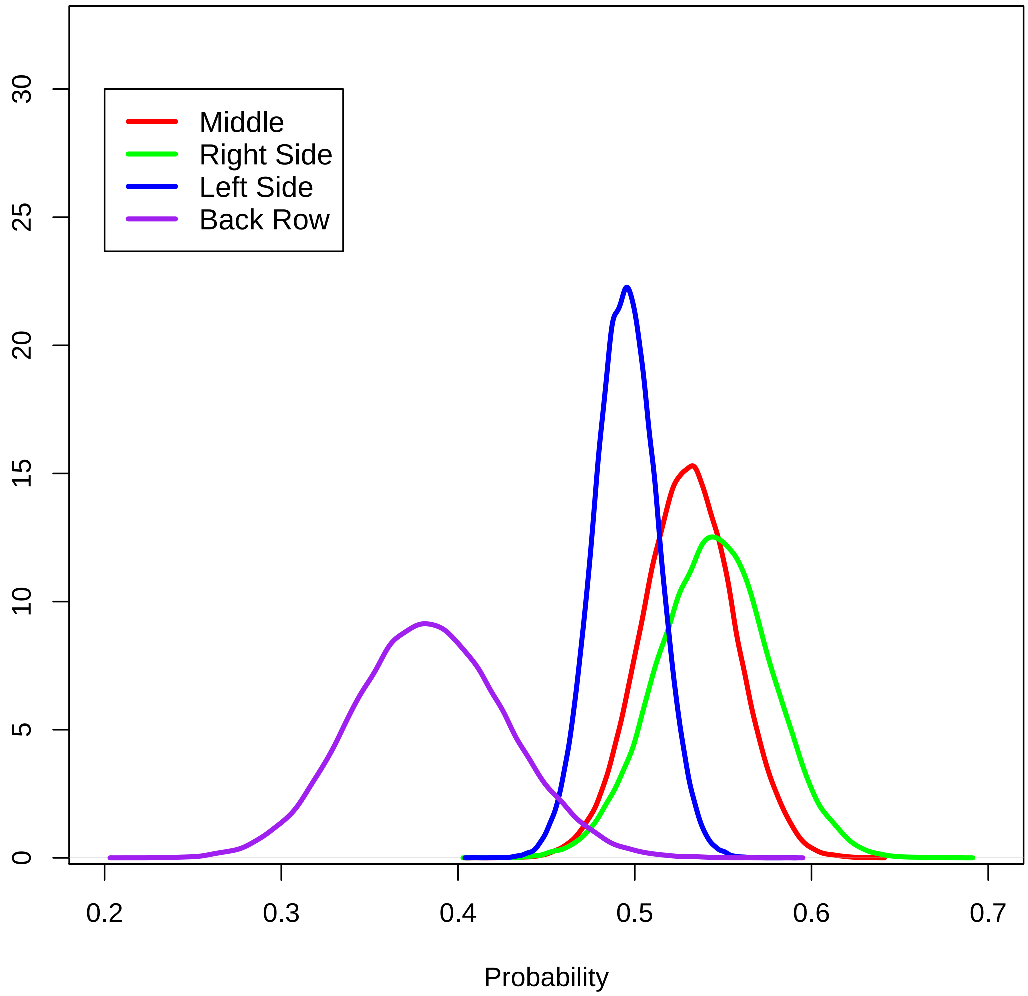

Figure 3.4: Posterior distributions of unconditional probabilities of attacks from various positions leading to scoring a point.

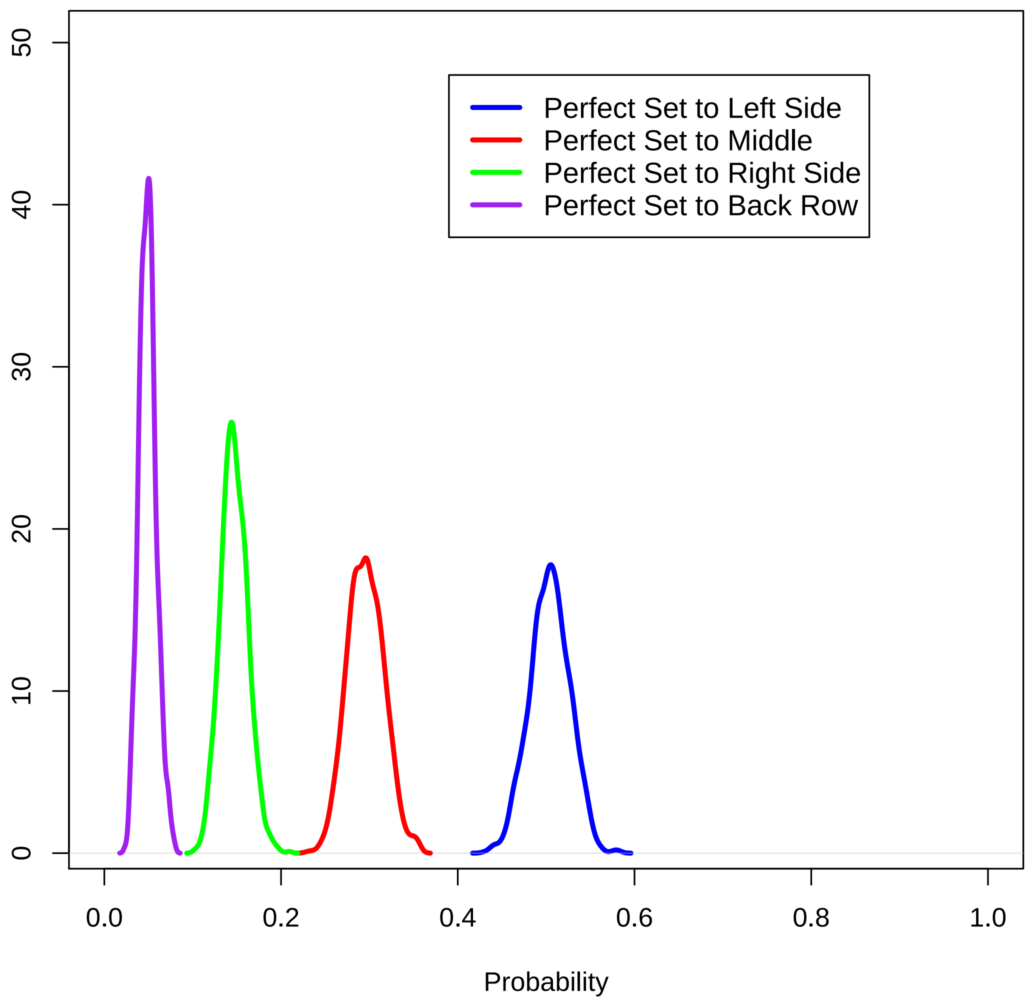

Figure 3.5: Posterior distributions of the transition probabilities of a perfect set to the various attack positions.

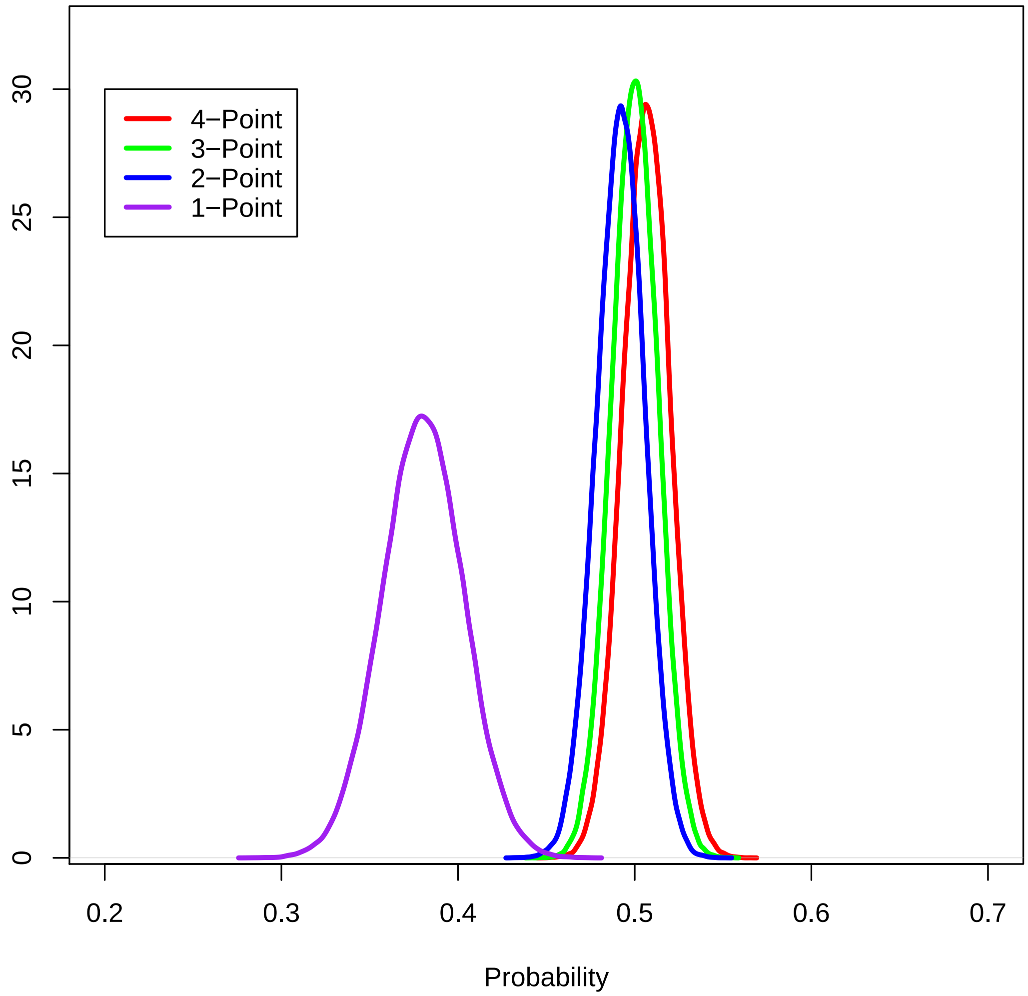

Figure 3.6: Posterior distributions for the unconditional probabilities of pass types leading to scoring a point when setting the prior counts to 1 for every state where the transition probability was not constrained to be 0 or 1.

a point for BYU of $\frac{1}{3}(.505 + .496 + .489) = .497$. Another player who earns a 3.0 passing average by having 70% 4-point passes, 20% 1-point passes, and 10% 0-point passes would only have a point probability of $.7 * .505 + .2 * .394 = .432$. While these examples are admittedly extreme, the deficiencies of the average as a rating system for passers is obvious. The large discrepancy of probability of point production from 0-point and 1-point passes relative to 2-, 3-, and 4-point passes should be taken into account.

In a similar vein, it seems reasonable that the target a passer aims for should be moved further off the net; the penalty paid for a 2-point pass is small compared to that paid for an overpass. Sending a setter close to the net leads to the occasional spectacular play but, based on our analysis, would have a lower expected long-run return.

We now take a brief look at the back set or set to the right side of the court. Figure 3.5 shows the probability of a perfect set being made to the various attack points on the court. It is easy to see that the probability of making a perfect back set is much lower than the probability of making a perfect set either to the left side or to the middle of the court. However, based on the results shown in Figure 3.4, a strategy that avoids the back set because of its difficulty would not be wise. The right side attack has an excellent probability of ending a rally positively. The difficulty of making the set should be tempered by the results found in Figures 3.2 and 3.3. The penalty paid for a less than perfect set is not shown to be high in this analysis. If the high and inside delivery can be avoided (Figure 3.2) the attack has a good probability of being successful.

Finally, we note that, for the BYU women's team, at least, the back row set should be avoided. This attack has significantly lower probability of success (Figures 3.3 and 3.4). We conjecture that this result would generalize well to other women's teams, but have some doubt about applying this generalization to men's teams.

We believe that the methodology described in this paper can be used to assist a coach in allocating practice time, focusing on optimal skill development, and optimizing attack strategies. It seems likely that extensions of this method could be implemented to help a coach determine which players (and the skill sets they bring to the court) should be used to form an optimal team.

BIBLIOGRAPHY

Anderson, T. W. and Goodman, L. A. (1957), "Statistical Inference about Markov Chains," *The Annals of Mathematical Statistics*, 28, 89–110.

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

Boender, C. G. E. and Rinnooy-Kan, A. H. G. (1983), "A Bayesian Analysis of the Number of Cells of a Multinomial Distribution," *The Statistician*, 32, 240–248, proceedings of the 1982 I.O.S. Annual Conference on Practical Bayesian Statistics.

Byra, M. and Scott, A. (1983), "A method for recording team statistics in volleyball," *Volleyball Technical Journal*, 7, 18–23.

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.

Casella, G. and George, E. I. (1992), "Explaining the Gibbs Sampler," *The American Statistician*, 46, 167–174.

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

Coleman, J., Neville, B., and Gordon, B. (1971), "A statistical system for volleyball and its use in Chicago women's association," *International Volleyball Review*, 27, 72–73.

Craig, B. A. and Sendi, P. P. (2002), "Estimation of the transition matrix of a discrete-time Markov chain," *Health Economics*, 11, 33–42.

Daniel, R. and Hughes, M. (2003), "Playing patterns of elite and non-elite volleyball," *Journal of Sports Sciences (London)*, 21, 268.

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

Duncan, G. T. and Lin, L. G. (1972), "Inference for Markov Chains Having Stochastic Entry and Exit," *Journal of the American Statistical Association*, 67, 761–767.

Eom, H. and Schutz, R. (1992), "Statistical analyses of volleyball team performance," *Research Quarterly for Exercise and Sport*, 63, 11–18.

Ezzati, A. (1974), "Forecasting Market Shares of Alternative Home-Heating Units by Markov Process using Transition Probabilities Estimated from Aggregate Time Series Data," *Management Science*, 21, 462–473.

Fahrmeir, L. (1992), "Posterior Mode Estimation by Extended Kalman Filtering for Multivariate Dynamic Generalized Linear Models," *Journal of the American Statistical Association*, 87, 501–509.

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

Judge, G. G. and Takayama, T. (1966), "Inequality Restrictions in Regression Analysis," *Journal of the American Statistical Association*, 61, 166–181.

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

Lirdla, D. (1980), "Statistical study of individual ball play," *Volleyball Technical Journal*, 5, 31–34.

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

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

Mortensen, N. (2007), "Development of a Notational Analysis System to Evaluate Setting Performance in Volleyball," Master's thesis, Brigham Young University, Provo, Utah.

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

Rose, R. (1983), "Statistical analysis at the 1983 Men's N.C.A.A. National Championship," *Volleyball Technical Journal*, 7, 15–17.

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

Satia, J. K. and Lave, R. E. (1973), "Markovian Decision Processes with Uncertain Transition Probabilities," *Operations Research*, 21, 728–740.

Sawula, L. (1977), "Individual action plan," *Volleyball Technical Journal*, 3, 2–8.

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

Telser, L. (1963), *Measurement of Economics*, Stanford: Stanford University Press.

Theil, H. and Rey, G. (1966), "A Quadratic Programming Approach to the Estimation of Transition Probabilities," *Management Science*, 12, 714–721.

Verhagen, E. A. L. M., der Beek, A. J. V., Bouter, L. M., Bahr, R. M., and Van-Mechelen, W. (2004), "A one season prospective cohort study of volleyball injuries," *British Journal of Sports Medicine*, 38, 477–481.

Vojik, J. (1980), "Several remarks to a system of accumulating data in volleyball," *Volleyball Technical Journal*, 5, 35–41.

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 George, E. I. (2005), "Exploratory Bayesian Model Selection for Serial Genetics Data," *Biometrics*, 61, 591–599.

A. COLLAPSING THE COUNT MATRIX

The orginal count matrix using all the codes originally used by the individual rating the skills was 127×127. Due to the large number of skill levels in certain transition states, it was difficult to assimilate desired information from the matrix. There was also concern with lack of data in certain regions of the count matrix. To alleviate this problem, counts of similar rows and columns of the count matrix were added together to create a smaller count matrix. The following changes to the count matrix were made:

(1) Passes received from float and jump serves were combined according to the grade assigned. For example, a 4-point pass from a float serve and a 4-point pass from a jump serve were combined into a 4-point pass.

(2) All digs were combined into a single state.

(3) Setting was combined by either set distance, set placement, or both depending on the skill of interest.

(4) The types of attack were grouped according to position on the court (middle, right side, left side, back row). Table A.1 displays the original attack types and which position of the court they were assigned. Setter-dump, out-of-system front row attack, and overpass categories were kept separate from attack positions.

The resulting count matrix was 35×35 when analyzing set distance, 37×37 when analyzing set placement, and 55×55 when analyzing the possible combinations of set placement and distance. These adjusted count matrices were used in estimating the transition probabilities and unconditional probability distributions discussed in this project.

Table A.1: The original attack types combined according to court position (middle, right, left, back row).

Attack Type	Court Position
Front 2	Middle
Gap Set	Middle
Back 1	Middle
Slide	Middle
Fast Slide	Middle
“X-series” or Combo	Right
Right Side “Red”	Right
High Set to RS	Right
Go	Left
Hut	Left
Highball “4”	Left
Inside Left Side Set “Rip”	Left
Pipe or BIC	Back Row
Back Row B Set	Back Row
Back Row Right Side ”D”	Back Row

B. SAMPLE OF RAW DATA

The following is a small portion of the raw data produced by the software Data Volley. This excerpt is the beginning of the match between Virginia Commonwealth University and Brigham Young University.


```
[Match]
01/09/2007;;;;;;;;1;1;;;DVSW Release 3.7.5;
[Team]
BYU;Brigham Young University;3;Watson  Jason;Huebner  Aldridge;
VCU;VCU;0;;;
[Oders]
;;;;;;;15;3;
[MatchComments]
;;;;
[Set]
True;9 -10;20-17;25-20;30-22;24;
True;10-8;20-15;25-18;30-21;23;
True;10-7;20-13;25-17;30-21;23;
True;;;;;;
True;;;;;;
[Player1]
0;1;1;2;2;2;;;GOO-CHE;Goodman;;;
0;2;2;;;;;;BEA-JAN;Beaumont;;;
0;3;3;;;;;;EVA-LIN;Evans;;;
0;4;4;;;;;;HAN-ASH;Hansen;;;
0;5;5;;;;;;RIC-LAU;Richards;;;
0;6;6;;;;;;STI-TES;Stimpson;;;
0;7;7;5;;;;;WIL-KIM;Wilson;;;
0;8;8;;;;;;VAN-MAR;Vandersteen;;;
0;9;9;;;;;;BRO-LEX;Brown;;;
0;10;10;4;4;4;;;LOT-ERI;Lott;;;
0;11;11;;5;5;;;KEM-ANI;Kemp;;;
0;12;12;;;;;;PAR-CAT;Parker;;;
0;13;13;;;;;;POR-BRY;Porter;;;
0;14;14;3;3;3;;;HAR-LIN;Hartsock;;;
0;15;15;1;1;1;;;SCH-AMY;Schlauder;;;
0;16;16;;;;;;LAU-STE;Lau;;;
0;20;17;;;;;;JUD-JEN;Judkins;;;
0;24;18;;;;;;DYE-RAC;Dyer;;;
[Player2]
[Scout]
```

```
*P15;;;;;;;09.54.08;1;1;1;;;;
aP3;;;;;;;09.54.08;1;1;1;;;;
*z1;;;;;;;09.54.08;1;1;1;;;;
az1;;;;;;;09.54.08;1;1;1;;;;
53SQ#;p;;;;;;09.54.08;1;1;1;1;39;;
15RQ=;p;;;;;;09.54.08;1;1;1;1;39;;
ap00:01;;;;;;;09.54.17;1;1;1;1;48;;
53SQ=;s;;;;;;09.54.25;1;1;1;1;56;;
49&H#;s;;;;;;09.54.25;1;1;1;1;56;;
*p01:01;;;;;;;09.54.31;1;1;1;1;62;;
*z6;;;;;;;09.54.31;1;6;1;1;62;;
01SQ/;;;;;;;09.54.43;1;6;1;1;74;;
65RQ/;p;;;;;;09.54.43;1;6;1;1;74;;
10AH#PO;p;p;;;;;09.54.51;1;6;1;1;82;;
*p02:01;;;;;;;09.54.54;1;6;1;1;85;;
01SQ+;;;;;;;09.55.06;1;6;1;1;97;;
65RQ!;;;;;;;09.55.06;1;6;1;1;97;;
49&H#;p;;;;;;09.55.06;1;6;1;1;97;;
99&H=;p;;;;;;09.55.06;1;6;1;1;97;;
*p03:01;;;;;;;09.55.18;1;6;1;1;109;;
01SQ#;p;;;;;;09.55.32;1;6;1;1;123;;
65RQ=;p;;;;;;09.55.32;1;6;1;1;123;;
*p04:01;;;;;;;09.55.39;1;6;1;1;130;;
01SQ=;s;;;;;;09.55.56;1;6;1;1;147;;
99&H#;s;;;;;;09.55.56;1;6;1;1;147;;
ap04:02;;;;;;;09.56.02;1;6;1;1;153;;
az6;;;;;;;09.56.02;1;6;6;1;153;;
59SQ=;s;;;;;;09.56.14;1;6;6;1;165;;
49&H#;s;;;;;;09.56.14;1;6;6;1;165;;
*p05:02;;;;;;;09.56.24;1;6;6;1;175;;
*z5;;;;;;;09.56.24;1;5;6;1;175;;
14SH+;;;;;;;09.56.33;1;5;6;1;184;;
65RH!;;;;;;;09.56.33;1;5;6;1;184;;
10DH!;;;;;;;09.56.41;1;5;6;1;192;;
49&H#;p;;;;;;09.56.41;1;5;6;1;192;;
99&H=;p;;;;;;09.56.41;1;5;6;1;192;;
*p06:02;;;;;;;09.56.53;1;5;6;1;204;;
14SH+;;;;;;;09.57.04;1;5;6;1;215;;
65RH+;;;;;;;09.57.04;1;5;6;1;215;;
49&H=;s;;;;;;09.57.04;1;5;6;1;215;;
99&H#;s;;;;;;09.57.04;1;5;6;1;215;;
ap06:03;;;;;;;09.57.21;1;5;6;1;232;;
az5;;;;;;;09.57.21;1;5;5;1;232;;
60SH#;p;;;;;;09.57.28;1;5;5;1;239;;
07RH=;p;;;;;;09.57.28;1;5;5;1;239;;
```

```
ap06:04;;;;;;;09.57.35;1;5;5;1;246;;
60SH!;;;;;;;09.57.42;1;5;5;1;253;;
07RH-;;;;;;;09.57.42;1;5;5;1;253;;
01AL=PB;p;r;;;;;09.57.47;1;5;5;1;258;;
99&H#;p;;;;;;09.57.47;1;5;5;1;258;;
ap06:05;;;;;;;09.57.54;1;5;5;1;265;;
60SH+;;;;;;;09.58.01;1;5;5;1;272;;
10RH+;;;;;;;09.58.01;1;5;5;1;272;;
15EH#;;;;;;;09.58.04;1;5;5;1;275;;
24AQ#P2;s;r;;;;;09.58.05;1;5;5;1;276;;
99&H=;s;;;;;;09.58.05;1;5;5;1;276;;
*p07:05;;;;;;;09.58.16;1;5;5;1;287;;
*z4;;;;;;;09.58.16;1;4;5;1;287;;
10SQ+;;;;;;;09.58.20;1;4;5;1;291;;
65RQ+;;;;;;;09.58.20;1;4;5;1;291;;
49&H=;s;;;;;;09.58.20;1;4;5;1;291;;
99&H#;s;;;;;;09.58.20;1;4;5;1;291;;
ap07:06;;;;;;;09.58.28;1;4;5;1;299;;
az4;;;;;;;09.58.28;1;4;4;1;299;;
65SQ+;;;;;;;09.58.42;1;4;4;1;313;;
10RQ+;;;;;;;09.58.42;1;4;4;1;313;;
15EH#;;;;;;;09.58.45;1;4;4;1;316;;
24AQ#P2;s;r;;;;;09.58.46;1;4;4;1;317;;
99&H=;s;;;;;;09.58.46;1;4;4;1;317;;
*p08:06;;;;;;;09.58.53;1;4;4;1;324;;
*z3;;;;;;;09.58.53;1;3;4;1;324;;
11SQ-;;;;;;;09.59.13;1;3;4;1;344;;
65RQ#;;;;;;;09.59.13;1;3;4;1;344;;
10DH=;s;;;;;;09.59.20;1;3;4;1;351;;
99&H#;s;;;;;;09.59.20;1;3;4;1;351;;
ap08:07;;;;;;;09.59.25;1;3;4;1;356;;
az3;;;;;;;09.59.25;1;3;3;1;356;;
55SQ#;p;;;;;;09.59.39;1;3;3;1;370;;
15RQ=;p;;;;;;09.59.39;1;3;3;1;370;;
ap08:08;;;;;;;09.59.46;1;3;3;1;377;;
55SQ+;;;;;;;09.59.55;1;3;3;1;386;;
15RQ+;;;;;;;09.59.55;1;3;3;1;386;;
15EH+;;;;;;;09.59.59;1;3;3;1;390;;
24AL/PB;p;r;;;;;10.00.01;1;3;3;1;392;;
99&H#;p;;;;;;10.00.01;1;3;3;1;392;;
ap08:09;;;;;;;10.00.11;1;3;3;1;402;;
49&H#;s;;;;;;10.00.11;1;3;3;1;402;;
99&H=;s;;;;;;10.00.11;1;3;3;1;402;;
*p09:09;;;;;;;10.00.32;1;3;3;1;423;;
*z2;;;;;;;10.00.32;1;2;3;1;423;;
```

C. R CODE

The following is the R Code written for this project. Section C.1 is the code to clean the data and prepare for the analysis. Section C.2 displays the code to compute the transition probability matrix depending on whether evaluating set placement, set distance, or a combination of the two. Section C.3 gives the code used to compute the unconditional probability point estimates and distributions. Section C.4 gives the code used to perform a sensitivity analysis on the prior counts of the unconditional probabilities.

C.1 Cleaning the data

```
#################################################################
##  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("New Matches/combined new.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 skill/score
vb$score <- substr(as.character(vb$play),5,5) #Separate 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("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")



transitions <- as.matrix(read.table("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")

######################################################################
# 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
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:
#Read in the prior counts
a.mat <- read.table("amat.txt", comment.char="")
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")



## Defines the names of all the different hits possible -
#  Will be used in the transition matrix
## These are the names associated with the full matrix (127 x 127)
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="")
c.mat <- as.data.frame(c.mat,row.names=hits)
names(c.mat) <- hits      #Names the columns of the data frame

#################################################################
#  This code collapses the count matrix by set placement.      #
# Similar code is run to collapse count matrix by set distance #
# and also combination of set distance and placement           #
#################################################################

### Collapse the count matrix:
#This combines the opponent jump and float serves
newcmat <- c.mat["OQ#",] + c.mat["OH#",]

#Carry over the float serves from BYU
newcmat["SH#",] <- c.mat["SH#",]
```

```
newcmat["SH/",] <- c.mat["SH/",]
newcmat["SH+",] <- c.mat["SH+",]
newcmat["SH!",] <- c.mat["SH!",]
newcmat["SH-",] <- c.mat["SH-",]
newcmat["SH=",] <- c.mat["SH=",]

#Carry over the jump serves from BYU
newcmat["SQ#",] <- c.mat["SQ#",]
newcmat["SQ/",] <- c.mat["SQ/",]
newcmat["SQ+",] <- c.mat["SQ+",]
newcmat["SQ!",] <- c.mat["SQ!",]
newcmat["SQ-",] <- c.mat["SQ-",]
newcmat["SQ=",] <- 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/",]

# Identify the set placements:
#Perfect Set (PS)
newcmat["PS",] <- c.mat["EQ#",] + c.mat["EH#",] + c.mat["ET#",] +
    c.mat["EM#",] + c.mat["EL#",]

#Low and Inside Set
newcmat["LIS",] <- c.mat["EQ+",] + c.mat["EH+",] + c.mat["ET+",] +
    c.mat["EM+",] + c.mat["EL+",]

#High and Outside Set
newcmat["HOS",] <- c.mat["EQ!",] + c.mat["EH!",] + c.mat["ET!",] +
    c.mat["EM!",] + c.mat["EL!",]

#Outside and Low Set
newcmat["OLS",] <- c.mat["EH-",] + c.mat["ET-",] + c.mat["EM-",] +
    c.mat["EL-",]

#Inside and High Set
newcmat["IHS",] <- c.mat["EH/",] + c.mat["ET/",] + c.mat["EM/",] +
    c.mat["EL/",]

newcmat["SetError",] <- c.mat["EH=",] + c.mat["ET=",] +
    c.mat["EM=",] + c.mat["EL=",]
```

```
newcmat["NotSetter",] <- c.mat["E",]

#Key for attack codes
# 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=",]

#Setter Dump
newcmat["SetDump",] <-
    c.mat["PS#",] + c.mat["PS+",] + c.mat["PS=",] + c.mat["PS/",]

#Out-of-system front row attack
newcmat["OutSystem",] <-
    c.mat["PA#",] + c.mat["PA+",] + c.mat["PA=",]

#Overpass
newcmat["Overpass",]<-
    c.mat["PO#",] + c.mat["PO+",] + c.mat["PO=",]

#Combine dig scores into one state
newcmat["Dig",] <-
    c.mat["DH#",] + c.mat["DH+",] + c.mat["DH!",] + c.mat["DH-",] +
    c.mat["DH/",] + c.mat["DH=",]

#Carry over the outcomes
newcmat["Good",] <- c.mat["Good",]
newcmat["Continue",] <- c.mat["Continue",]
newcmat["Bad",] <- c.mat["Bad",]


########################################################
### Do the column collapsing                           #
########################################################
newhits <- c("O", "SH#", "SH/", "SH+", "SH!", "SH-", "SH=",
    "SQ#", "SQ/", "SQ+", "SQ!", "SQ-", "SQ=",
    "4pt",  "3pt",  "2pt",  "1pt",  "0pt",  "PassOverpass",
    "PS", "LIS", "HOS", "OLS", "IHS", "SetError", "NotSetter",
    "Middle", "Right", "Left", "Back",
    "SetDump", "OutSystem", "Overpass",
    "Dig",  "Good",  "Continue", "Bad")

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

```
cmat2[,"SH="] <- newcmat[,"SH="]
cmat2[,"SQ#"] <- newcmat[,"SQ#"]
cmat2[,"SQ/"] <- newcmat[,"SQ/"]
cmat2[,"SQ+"] <- newcmat[,"SQ+"]
cmat2[,"SQ!"] <- newcmat[,"SQ!"]
cmat2[,"SQ-"] <- newcmat[,"SQ-"]
cmat2[,"SQ="] <- 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[,"PS"] <-
    newcmat[,"EQ#"] + newcmat[,"EH#"] + newcmat[,"ET#"] +
    newcmat[,"EM#"] + newcmat[,"EL#"]

cmat2[,"LIS"] <-
    newcmat[,"EQ+"] + newcmat[,"EH+"] + newcmat[,"ET+"] +
    newcmat[,"EM+"] + newcmat[,"EL+"]

cmat2[,"HOS"] <-
    newcmat[,"EQ!"] + newcmat[,"EH!"] + newcmat[,"ET!"] +
    newcmat[,"EM!"] + newcmat[,"EL!"]

cmat2[,"OLS"] <-
    newcmat[,"EH-"] + newcmat[,"ET-"] + newcmat[,"EM-"] +
    newcmat[,"EL-"]

cmat2[,"IHS"] <-
    newcmat[,"EH/"] + newcmat[,"ET/"] + newcmat[,"EM/"] +
    newcmat[,"EL/"]

cmat2[,"SetError"] <-
    newcmat[,"EH="] + newcmat[,"ET="] + newcmat[,"EM="] +
    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#"] + newcmat[,"DH+"] + newcmat[,"DH!"] +
    newcmat[,"DH-"] + newcmat[,"DH/"] + newcmat[,"DH="]
```

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

c.mat <- cmat2
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="")
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 <- a.mat["OQ#",] + a.mat["OH#",]

newamat["SH#",] <- a.mat["SH#",]
newamat["SH/",] <- a.mat["SH/",]
newamat["SH+",] <- a.mat["SH+",]
newamat["SH!",] <- a.mat["SH!",]
newamat["SH-",] <- a.mat["SH-",]
newamat["SH=",] <- a.mat["SH=",]
newamat["SQ#",] <- a.mat["SQ#",]
newamat["SQ/",] <- a.mat["SQ/",]
newamat["SQ+",] <- a.mat["SQ+",]
newamat["SQ!",] <- a.mat["SQ!",]
newamat["SQ-",] <- a.mat["SQ-",]
newamat["SQ=",] <- 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["PS",] <- a.mat["EQ#",] + a.mat["EH#",] + a.mat["ET#",] +
    a.mat["EM#",] + a.mat["EL#",]
```

```
newamat["LIS",] <- a.mat["EQ+",] + a.mat["EH+",] + a.mat["ET+",] +
    a.mat["EM+",] + a.mat["EL+",]

newamat["HOS",] <- a.mat["EQ!",] + a.mat["EH!",] + a.mat["ET!",] +
    a.mat["EM!",] + a.mat["EL!",]

newamat["OLS",] <- a.mat["EH-",] + a.mat["ET-",] + a.mat["EM-",] +
    a.mat["EL-",]

newamat["IHS",] <- a.mat["EH/",] + a.mat["ET/",] + a.mat["EM/",] +
    a.mat["EL/",]

newamat["SetError",] <-
    a.mat["EH=",] + a.mat["ET=",] + a.mat["EM=",] + 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#",] + a.mat["DH+",] + a.mat["DH!",] +
    a.mat["DH-",] + a.mat["DH/",] + 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[,"SH#"] <- newamat[,"SH#"]
amat2[,"SH/"] <- newamat[,"SH/"]
amat2[,"SH+"] <- newamat[,"SH+"]
amat2[,"SH!"] <- newamat[,"SH!"]
amat2[,"SH-"] <- newamat[,"SH-"]
amat2[,"SH="] <- newamat[,"SH="]
amat2[,"SQ#"] <- newamat[,"SQ#"]
amat2[,"SQ/"] <- newamat[,"SQ/"]
amat2[,"SQ+"] <- newamat[,"SQ+"]
amat2[,"SQ!"] <- newamat[,"SQ!"]
amat2[,"SQ-"] <- newamat[,"SQ-"]
amat2[,"SQ="] <- 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[,"PS"] <- newamat[,"EQ#"] + newamat[,"EH#"] + newamat[,"ET#"] +
    newamat[,"EM#"] + newamat[,"EL#"]
```

```
amat2[,"LIS"] <- newamat[,"EQ+"] + newamat[,"EH+"] + newamat[,"ET+"] +
    newamat[,"EM+"] + newamat[,"EL+"]

amat2[,"HOS"] <- newamat[,"EQ!"] + newamat[,"EH!"] + newamat[,"ET!"] +
    newamat[,"EM!"] + newamat[,"EL!"]

amat2[,"OLS"] <- newamat[,"EH-"] + newamat[,"ET-"] + newamat[,"EM-"] +
    newamat[,"EL-"]

amat2[,"IHS"] <- newamat[,"EH/"] + newamat[,"ET/"] + newamat[,"EM/"] +
    newamat[,"EL/"]

amat2[,"SetError"] <- newamat[,"EH="] + newamat[,"ET="] +
    newamat[,"EM="] + 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#"] + newamat[,"DH+"] + newamat[,"DH!"] +
    newamat[,"DH-"] + newamat[,"DH/"] + 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 if data exists there:
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
    }
}

a.mat <- amat3

#Save the count matrix for later use
save(a.mat,file="collapsedamatR.txt")

####################################################################
#Similar code is used to combine according to set distance and the #
# combinations.  Only the altered portion of code is shown below.  #
####################################################################

##
### For combining by set distance, replace set placement code with:
```

```
##

#Combining 0-1 feet from net and 1-3 feet from net
#There were only 4 hits total 0-1 feet from net
# Set 0 to 3 feet from the 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=",]

# Set 3 to 5 feet from net
newcmat["3to5ft",] <- c.mat["ET#",] + c.mat["ET+",] + c.mat["ET!",] +
c.mat["ET-",] + c.mat["ET/",] + c.mat["ET=",]

# Set 5 to 8 feet from net
newcmat["5to8ft",] <- c.mat["EM#",] + c.mat["EM+",] + c.mat["EM!",] +
c.mat["EM-",] + c.mat["EM/",] + c.mat["EM=",]

# Set 8 to 10 feet from net
newcmat["8to10ft",] <- c.mat["EL#",] + c.mat["EL+",] + c.mat["EL!",] +
c.mat["EL-",] + c.mat["EL/",] + c.mat["EL=",]

# Set performed by someone who wasn't the setter
newcmat["NotSetter",] <- c.mat["E",]


##
### For combining both set placement and distance:
##

newcmat["0to3PS",] <- c.mat["EQ#",] + c.mat["EH#",]
newcmat["0to3LIS",] <- c.mat["EQ+",] + c.mat["EH+",]
newcmat["0to3HOS",] <- c.mat["EQ!",] + c.mat["EH!",]
newcmat["0to3OLS",] <- c.mat["EH-",]
newcmat["0to3IHS",] <- c.mat["EH/",]
newcmat["0to3SetError",] <- c.mat["EH=",]

newcmat["3to5PS",] <- c.mat["ET#",]
newcmat["3to5LIS",] <- c.mat["ET+",]
newcmat["3to5HOS",] <- c.mat["ET!",]
newcmat["3to5OLS",] <- c.mat["ET-",]
newcmat["3to5IHS",] <- c.mat["ET/",]
newcmat["3to5SetError",] <- c.mat["ET=",]

newcmat["5to8PS",] <- c.mat["EM#",]
newcmat["5to8LIS",] <- c.mat["EM+",]
```

```
newcmat["5to8HOS",] <- c.mat["EM!",]
newcmat["5to8OLS",] <- c.mat["EM-",]
newcmat["5to8IHS",] <- c.mat["EM/",]
newcmat["5to8SetError",] <- c.mat["EM=",]

newcmat["8to10PS",] <- c.mat["EL#",]
newcmat["8to10LIS",] <- c.mat["EL+",]
newcmat["8to10HOS",] <- c.mat["EL!",]
newcmat["8to10OLS",] <- c.mat["EL-",]
newcmat["8to10IHS",] <- c.mat["EL/",]
newcmat["8to10SetError",] <- c.mat["EL=",]
newcmat["NotSetter",] <- c.mat["E",]
```

C.2 Computing the transition probability matrix

```
#This code produces the posterior means to insert into the
# transition probability matrix

##################################################################
### This calculates the point estimates for the                ###
### transition probability matrix  according to set placement:  ###
##################################################################

# Load c.mat and a.mat from the Collapsed Trans. matrix file
load("By Set Placement/collapsedcmatR.txt")
load("By Set Placement/collapsedamatR.txt")

# Define the names of the transition matrix:
newhits <- c("O", "SH#", "SH/", "SH+", "SH!", "SH-", "SH=",
    "SQ#", "SQ/", "SQ+", "SQ!", "SQ-", "SQ=",
    "4pt",  "3pt",  "2pt",  "1pt",  "0pt",  "PassOverpass",
    "PS", "LIS", "HOS", "OLS", "IHS", "SetError", "NotSetter",
    "Middle", "Right", "Left", "Back",
    "SetDump", "OutSystem", "Overpass",
    "Dig",  "Good",  "Continue", "Bad")

c.matcol <- ncol(c.mat)
c.matrow <- nrow(c.mat)

#Initialize transition matrix with zeros to hold point estimates
meanpost <- matrix(0, nrow=c.matrow, ncol=c.matcol,
    dimnames=list(newhits,newhits))

for(row in 1:c.matrow){
```

```
    rowi <- as.matrix(c.mat[row,]) #Only deal with one row at a time

    #This locates nonzero states in transition matrix and
    # records the column numbers
    index <- NA
    for(i in 1:length(rowi)){
        if(rowi[i]>0){index <- c(index,i)}
    }

    # Calculate the posterior mean for each transition probability
    #  Insert the mean into the transition matrix
    for(col in index[-1]){
        meanpost[row,col] <-
        (rowi[col] + a.mat[row,col])/sum(rowi + a.mat[row,])
    }
}

save(meanpost, file="By Set Placement/meanpost.txt")

###################################################################
### This calculates the point estimates for the                 ###
### transition probability matrix according to set distance     ###
###################################################################

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

# This loads c.mat and a.mat from the Collapsed Trans. matrix file
load("By Set Distance/collapsedcmatR.txt")
load("By Set Distance/collapsedamatR.txt")

c.matcol <- ncol(c.mat)
c.matrow <- nrow(c.mat)

#Initialize transition matrix with zeros to hold point estimates
meanpost <- matrix(0, nrow=c.matrow, ncol=c.matcol,
    dimnames=list(newhits,newhits))

for(row in 1:c.matrow){
```

```
    rowi <- as.matrix(c.mat[row,])  # Only deal with one row at a time

    index <- NA
    for(i in 1:length(rowi)){
        if(rowi[i]>0){index <- c(index,i)}
    }

    # Calculate the posterior mean for each transition probability
    #  Insert the mean into the transition matrix

    for(col in index[-1]){
        meanpost[row,col] <-
        (rowi[col] + a.mat[row,col])/sum(rowi + a.mat[row,])
    }
}

meanpostdistance <- meanpost

save(meanpost, file="By Set Distance/meanpost.txt")


###################################################################
### This calculates the point estimates for the                 ###
### transition probability matrix  with sets still separate     ###
###################################################################

newhits <- c("0", "SH#", "SH/", "SH+", "SH!", "SH-", "SH=", "SQ#",
    "SQ/", "SQ+", "SQ!", "SQ-", "SQ=",
    "4pt",  "3pt",  "2pt",  "1pt",  "0pt",  "PassOverpass",
    "0to3PS", "0to3LIS", "0to3HOS", "0to3OLS", "0to3IHS",
    "0to3SetError", "3to5PS", "3to5LIS", "3to5HOS",
    "3to5OLS", "3to5IHS", "3to5SetError",
    "5to8PS", "5to8LIS", "5to8HOS",
    "5to8OLS", "5to8IHS", "5to8SetError",
    "8to10PS", "8to10LIS", "8to10HOS",
    "8to10OLS", "8to10IHS", "8to10SetError", "NotSetter",
    "Middle", "Right", "Left", "Back",
    "SetDump", "OutSystem", "Overpass",
    "Dig", "Good",  "Continue", "Bad")

# This loads "c.mat" and "a.mat" from the Collapsed Trans. matrix file
load("Sets Still Separate/collapsedcmatR.txt")
load("Sets Still Separate/collapsedamatR.txt")
```

```
c.matcol <- ncol(c.mat)
c.matrow <- nrow(c.mat)

#Initialize transition matrix with zeros to hold point estimates
meanpost <- matrix(0, nrow=c.matrow, ncol=c.matcol,
    dimnames=list(newhits,newhits))

for(row in 1:c.matrow){

    rowi <- as.matrix(c.mat[row,])  # Only deal with one row at a time

    index <- NA
    for(i in 1:length(rowi)){
        if(rowi[i]>0){index <- c(index,i)}
    }

    # Calculate the posterior mean for each transition probability
    #  Insert the mean into the transition matrix

    for(col in index[-1]){
        meanpost[row,col] <-
        (rowi[col] + a.mat[row,col])/sum(rowi + a.mat[row,])
    }
}

meanpostsetsep <- meanpost

save(meanpost, file="Sets Still Separate/meanpost.txt")
```

C.3 Computing the unconditional probabilities

```
####################################################################
### Calculating the unconditional probability distributions      ##
##     for passing, set placement, attack                        ##
####################################################################

##  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("By Set Placement/collapsedcmatR.txt")
load("By Set Placement/collapsedamatR.txt")
```

```
c.matcol <- ncol(c.mat)
c.matrow <- nrow(c.mat)

# This code locates in the count matrix the nonzero counts
# and records the column numbers in each row
# Only want to draw probabilities where there are counts

#Initialize the matrix with -1's.
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("O", "SH#", "SH/", "SH+", "SH!", "SH-", "SH=",
    "SQ#", "SQ/", "SQ+", "SQ!", "SQ-", "SQ=",
    "4pt",  "3pt",  "2pt",  "1pt",  "0pt",  "PassOverpass",
    "PS", "LIS", "HOS", "OLS", "IHS", "SetError", "NotSetter",
    "Middle", "Right", "Left", "Back", "SetDump", "OutSystem",
    "Overpass", "Dig",  "Good",  "Continue", "Bad")

#Specify what you want to look at:
pass <- c("4pt",  "3pt",  "2pt",  "1pt",  "0pt",  "PassOverpass")
set <- c("PS", "LIS", "HOS", "OLS", "IHS", "SetError", "NotSetter")
attack <- c("Middle", "Right", "Left", "Back",
            "SetDump", "OutSystem", "Overpass")
outcome <- c("Good", "Continue", "Bad")

#Total number of draws from posterior distribution
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))

lset <- length(set)
lpass <- length(pass)
```

```
loutcome <- length(outcome)
lattack <- length(attack)

# Create a matrix to store the unconditional probabilities for passing
passoverall <- matrix(NA, nrow=lpass, ncol=loutcome,
    dimnames=list(pass, outcome))
passposts <- matrix(0,nrow=lpass, ncol=loutcome,
    dimnames=list(pass, outcome))

# Matrix to store unconditional probabilities for set placement
setplaceoverall <- matrix(NA, nrow=lset, ncol=loutcome,
    dimnames=list(set, outcome))
setplaceposts <- matrix(0,nrow=lset, ncol=loutcome,
    dimnames=list(set, outcome))

# Matrix to store the unconditional probabilities for attack
attackoverall <- matrix(NA, nrow=lattack, ncol=loutcome,
    dimnames=list(attack, outcome))
attackposts <- matrix(0,nrow=lattack, ncol=loutcome,
    dimnames=list(attack, outcome))


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
    allsetplacemat <- abind(allsetplacemat,post, along=3)
```

```
###
# Calculates 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
    }
}
#Combine the matrices of uncond. probs. along the third dimension
passposts <- abind(passposts, passoverall, along=3)

###
## 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 <- abind(setplaceposts,setplaceoverall, along=3)

    ###
    ## 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 <- abind(attackposts,attackoverall, along=3)
}

#Delete the first matrix of zeroes (dummy matrix)
allsetplacemat <- allsetplacemat[,,-1]
passposts <- passposts[,,-1]
setplaceposts <- setplaceposts[,,-1]
attackposts <- attackposts[,,-1]
```

```
save(allsetplacemat,file="allsetplacemat.txt")
save(passposts,file="Passing Results/passposts.txt")
save(setplaceposts,file="Set Placement Results/setplaceposts.txt")
save(attackposts,file="Attack Results/attackposts.txt")




##########################################################
##   Calculate unconditional probability distribution   ##
##    for set distance                                  ##
##########################################################

library(abind)

# This loads c.mat and a.mat from the Collapsed Trans. matrix file
load("By Set Distance/collapsedcmatR.txt")
load("By Set Distance/collapsedamatR.txt")

c.matcol <- ncol(c.mat)
c.matrow <- nrow(c.mat)

# This matrix lists the indices that have counts listed in c.mat
# It is used in the loop to only draw values for those states
#   in which we have counts
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("O", "SH#", "SH/", "SH+", "SH!", "SH-", "SH=",
    "SQ#", "SQ/", "SQ+", "SQ!", "SQ-", "SQ=",
    "4pt",  "3pt",  "2pt",  "1pt",  "0pt",  "PassOverpass",
    "0to3ft", "3to5ft", "5to8ft", "8to10ft", "NotSetter",
    "Middle", "Right", "Left", "Back",
    "SetDump", "OutSystem", "Overpass",
    "Dig",  "Good",  "Continue", "Bad")
```

```
#Specify what you want to look at:
outcome <- c("Good", "Continue", "Bad")
set <- c("0to3ft", "3to5ft", "5to8ft", "8to10ft", "NotSetter")
attack <- c("Middle", "Right", "Left", "Back",
            "SetDump", "OutSystem", "Overpass")

#Total number of draws from posterior distribution
nloops <- 100000
post <- matrix(NA, nrow=c.matrow, ncol=c.matcol,
    dimnames=list(newhits,newhits))

lset <- length(set)
loutcome <- length(outcome)
lattack <- length(attack)

#Create a matrix to store the unconditional probabilities
overallpost <- matrix(NA, nrow=lset, ncol=loutcome,
    dimnames=list(set, outcome))
distsetdists <- matrix(0,nrow=lset, ncol=loutcome,
    dimnames=list(set, outcome))


for(loop in 1:nloops){

    # Generate a whole new matrix of transition probabilities
    # Generate values from a gamma distribution -
        #  convert to Dirichlet dist.
    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 Dirichlet dist. by dividing by row sum total
        post[row,] <- draws/sum(draws)
    }

    # Calculate the unconditional probabilities for
    #     each new transition matrix:
    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]
            }

            # Want to include the probability of
            #  going directly to an outcome from the set
            prob <- prob + post[setk,"NotSetter"]*
                    post["NotSetter",outcomej] + post[setk,outcomej]

            overallpost[k,j] <- prob
        }
    }
    distsetdists <- abind(distsetdists,overallpost, along=3)
}

#Delete the first matrix of zeroes (dummy matrix)
distsetdists <- distsetdists[,,-1]

save(distsetdists,file="Set Distance Results/distsetdists.txt")




##########################################################
##   Calculate unconditional probability distributions  ##
##    for set distance/placement combined              ##
##########################################################

library(abind)

# This loads c.mat and a.mat from the Collapsed Trans. matrix file
load("Sets Still Separate/collapsedcmatR.txt")
load("Sets Still Separate/collapsedamatR.txt")

c.matcol <- ncol(c.mat)
```

```
c.matrow <- nrow(c.mat)

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", "SH#", "SH/", "SH+", "SH!", "SH-", "SH=", "SQ#",
   "SQ/", "SQ+", "SQ!", "SQ-", "SQ=",
   "4pt",  "3pt",  "2pt",  "1pt",  "0pt",  "PassOverpass",
   "0to3PS","0to3LIS","0to3HOS","0to3OLS",
   "0to3IHS", "0to3SetError",
   "3to5PS","3to5LIS","3to5HOS","3to5OLS",
   "3to5IHS", "3to5SetError",
   "5to8PS","5to8LIS","5to8HOS","5to8OLS",
   "5to8IHS", "5to8SetError",
   "8to10PS","8to10LIS", "8to10HOS", "8to10OLS", "8to10IHS",
   "8to10SetError", "NotSetter",
   "Middle", "Right", "Left", "Back",
   "SetDump", "OutSystem", "Overpass",
   "Dig", "Good",  "Continue", "Bad")

#Specify what you want to look at:
pass <- c("4pt",  "3pt",  "2pt",  "1pt",  "0pt",  "PassOverpass")
set<-c("0to3PS","0to3LIS","0to3HOS","0to3OLS",
   "0to3IHS","0to3SetError",
   "3to5PS","3to5LIS","3to5HOS","3to5OLS","3to5IHS","3to5SetError",
   "5to8PS","5to8LIS","5to8HOS","5to8OLS","5to8IHS","5to8SetError",
   "8to10PS", "8to10LIS", "8to10HOS", "8to10OLS",
   "8to10IHS", "8to10SetError", "NotSetter")
attack <- c("Middle", "Right", "Left", "Back",
    "SetDump", "OutSystem", "Overpass")
outcome <- c("Good", "Continue", "Bad")

#Total number of draws from posterior distribution
nloops <- 100000
post <- matrix(NA, nrow=c.matrow, ncol=c.matcol,
    dimnames=list(newhits,newhits))
allpost <- matrix(0, nrow=c.matrow, ncol=c.matcol,
```

```
    dimnames=list(newhits,newhits))

lset <- length(set)
lpass <- length(pass)
loutcome <- length(outcome)
lattack <- length(attack)

#Create a matrix to store the unconditional probabilities
overallpost <- matrix(NA, nrow=lset, ncol=loutcome,
    dimnames=list(set, outcome))
posts <- matrix(0,nrow=lset, ncol=loutcome,
    dimnames=list(set, outcome))

for(loop in 1:nloops){

    # Generate a whole new matrix
    # Generate values from a gamma distribution -
    #    convert to dirichlet dist.

    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)
    }
    allpost <- abind(allpost,post, along=3)

    for(k in 1:length(set)){
        for(j in 1:length(outcome)){
            prob <- 0
            for(i in 1:length(attack)){
                prob <- prob + post[set[k],attack[i]]*
                        post[attack[i],outcome[j]]
                prob <- prob + post[set[k],"NotSetter"]*
                        post["NotSetter",attack[i]]*
                        post[attack[i], outcome[j]]
            }
```

```
            #Include probability of going directly to outcome from set
            prob <- prob + post[set[k],"NotSetter"]*
                    post["NotSetter",outcome[j]]
            prob <- prob + post[set[k],outcome[j]]

            overallpost[k,j] <- prob
        }
    }

    #Concatenate the matrices along the third dimension
    posts <- abind(posts,overallpost, along=3)
}

#Delete the first matrix of zeroes (dummy matrix)
allpost <- allpost[,,-1]
posts <- posts[,,-1]

setSepUncpr <- posts
save(setSepUncpr,file="Sets Still Separate Results/setSepUncpr.txt")

setSepDists <- allpost
save(setSepDists, file="Sets Still Separate Results/setSepDists.txt")

load("Sets Still Separate Results/setSepDists.txt")
load("Sets Still Separate Results/setSepUncpr.txt")
```

C.4 Sensitivity analysis on the prior counts of the transition matrix

```
####################################################################
##  Performing a Sensitivity Analysis on computing the            ##
##    unconditional probability distributions as shown above     ##
##   This code is used for both point estimates and distributions##
####################################################################

# This loads c.mat and a.mat from the Collapsed Trans. matrix file
load("By Set Placement/collapsedcmatR.txt")
load("By Set Placement/collapsedamatR.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(a.mat[row,col]>0) a.matweak[row,col] <- 1
        else a.matweak[row,col] <- 0
    }
}
a.mat <- a.matweak


####################################################################
##  The rest of the code is the same as for calculating the      ##
##    unconditional probabilities as shown above                 ##
####################################################################

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
        }
    }
}

#etc...
```