AN ABSTRACT OF THE THESIS OF

James D. Kolsky for the degree of Doctor of Philosophy in Statistics presented on May 3, 1996. Title: Extensions for Paired Comparisons Models.

Redacted for privacy

Abstract approved: _

Daniel W. Schafer

The Thurstone-Mosteller and Bradley-Terry Models are commonly used to rank items from paired comparisons experiments in which one item in each pair "wins," and to assess the importance of time-independent explanatory variables on such rankings. The first part of this thesis clarifies the use of probit and logistic regression models for such designs, including the incorporation of time-dependent explanatory variables and the analysis of unbalanced designs. In addition, likelihood inference, using the EM Algorithm, is proposed for Thurstone's Case III Model allowing the estimation of variance parameters to account for variable item performances.

The second half of this thesis presents an extension of the model to permitting the "performances" or "worths" of each competitor to be serially correlated. As an example, the performance of a basketball team in its current game is allowed to be correlated with its performance from the previous game. The Thurstone-Mosteller Model is sometimes motivated through the use of an underlying, normally-distributed performance distribution for each item or competitor, with a competitor winning a trial if a draw from its performance distribution exceeds that from its competitor's. The observed outcome is solely the win or loss for each team, but regression models, using either time-dependent or time-independent explanatory variables, may be specified for the performance means. The extension in this thesis comes from supposing the error structure for the performance distribution for each team is normal with first-order autocorrelation. The EM Algorithm is used, treating the underlying draws from the performance distributions as "missing data." This provides approximate maximum likelihood estimates; the approximation is due to the use of Monte Carlo integration in the E-step of the algorithm. Unfortunately, the heavy computational requirement and the inability to calculate the maximized likelihood function or the information matrix, make the approach unattractive for practical use. Two approximations are presented, however, which can be carried out with standard routines and some minor programming.

Keywords: auto-regressive model, Bradley-Terry Model, EM Algorithm, generalized linear model, logistic regression, MCEM Algorithm, probit regression, serial correlation, Thurstone-Mosteller Model.

Extensions for Paired Comparisons Models

by

James D. Kolsky

A THESIS

submitted to

Oregon State University

in partial fulfillment of
the requirements for the
degree of

Doctor of Philosophy

Presented May 3, 1996
Commencement June 1996

© Copyright by James D. Kolsky
May 3, 1996
All Rights Reserved

Doctor of Philosophy thesis of James D. Kolsky presented on May 3, 1996

APPROVED:

Redacted for privacy

Major Professor, representing Statistics

Redacted for privacy

Chair of Department of Statistics

Redacted for privacy

Dean of Graduate School

I understand that my thesis will become part of the permanent collection of Oregon State University libraries. My signature below authorizes release of my thesis to any reader upon request.

Redacted for privacy

James D. Kolsky, Author

ACKNOWLEDGMENTS

As with any project of this magnitude, it could not have been completed without the help of numerous people and it is not possible here to thank everyone who had a hand in its successful completion.

I would like to thank, first and foremost, Dr. Daniel W. Schafer. He has provided invaluable advice, suggestions, and guidance often meeting with me with little or no advance warning. In addition, it was only with his impetus that I was able to incorporate my long-standing interest in baseball with my thesis.

Even with Dan's technical support, this project would have been much more difficult to complete with the encouragement and support, both emotional and financial, of my wife, Cathy.

And even with all the advice and support from Dan and Cathy, it may not have been possible to finish in time for graduation this year had Thom Maan not submitted and monitored S batch files for me over the weekends for the last three months.

I would also like to thank all of the students, staff, and faculty both past and present for providing such a pleasant atmosphere which greatly enhanced my experience at Oregon State University. In particular, I would like to thank Dr. Cliff Pereira, from whom I have gained invaluable experience in the art of consulting and who greatly enhanced my knowledge of statistics during our weekly tutoring sessions, and Genevieve Downing and Ali Gray, who provided a haven away from the trials of graduate student responsibilities without ever seeming to be tired of my continued appearances in the office.

And I would like to thank Dr. Justus Seely and the Department of Statistics for providing me with a position as a research assistant which has funded my education for the last four years.

Lastly, a great deal of thanks goes to my family who have always, and without reservation, provided moral, emotional, and financial support so that I might pursue my educational goals.

TABLE OF CONTENTS

TABLE OF CONTENTS (Continued)

TABLE OF CONTENTS (Continued)

LIST OF FIGURES

LIST OF TABLES

This thesis is dedicated in loving memory of
Mom

Extensions For Paired Comparisons Models

1. Introduction

Paired comparisons describes a data structure in which *r* treatments, items, or individuals are compared in pairs. A completely balanced paired comparison design can be conceptualized as an incomplete block design with a block size of 2 and $\binom{r}{2}$ blocks (Bradley, 1984). However, in many paired comparisons experiments, the response associated with each block is binary, indicating which item of the pair was preferred. Such data can be collected either actively or passively. For the former, an impartial judge, or judges, compares two items or treatments at a time with respect to a specified characteristic, such as taste, which cannot be quantified on a physical scale. For the latter, outcomes from each comparison are observed, the winning team at a sporting event for example. Because there is no quantitative scale on which to "score" the objects, the outcomes of such comparisons are most often binary, indicating only which of the treatments, items, or individuals in that pair was preferred. A food-preference study for example, comparing food products with differing levels of MSG (Atkinson, 1972), is a simple example of a paired comparison design. In such studies an independent and unbiased panel decides which of the two items in each comparison tastes better. Other examples can be found in the biological fields. For example, there may be interest in determining the factors which influence the outcomes of battles for dominance among a particular species. Since no quantitative measure of the ability to dominate or taste exists, the only meaningful observation is the "winner" from each comparison. Analytical tools, used for the analysis of these designs, must continually be improved in such a way that eliminates restrictions on the data and increases the amount of information that can be obtained from the analysis.

It is convenient to envision the judge's decision to "choose" item i over item j as being made in accordance with an unobserved variable which quantitatively measures the characteristic upon which the comparison is being made. The perception of the judges is influenced by this unobserved variable and the item with the greatest effect on the observer is "chosen" or "preferred." This is the rational used by L. L. Thurstone (1927) when describing the analysis of paired comparison models in the context of psychological scaling experiments. Such an idea is easily extended to the context of sporting events by assuming that each team performs according to an unobserved, performance variable. The team or individual with the higher realization of that variable wins that particular game or match.

Thurstone (1927) is commonly acknowledged as developing the first mathematically based method of analysis for paired comparisons experiments. However, because his work primarily involved psychological scaling studies, paired comparison studies were used for little else for the first half of this century. A renewed interest in paired comparison models over the last 40 years, however, has greatly increased the number of fields in which such designs are commonly used. Currently, paired comparison designs are used in fields as diverse as sports, acoustics, animal ecology, economics, and preference testing (David, 1988).

The Thurstone-Mosteller Model (Thurstone, 1927, Mosteller, 1951a,b,c) and the Bradley-Terry Model (Bradley and Terry, 1952) are the most commonly used models for the analysis of these designs. In the former the probability, π_{ij}, that item i is preferred over treatment j can be modeled as:

$$probit(\pi_{ij}) = \alpha_i - \alpha_j \tag{1.1.1}$$

and in the latter as:

$$logit(\pi_{ij}) = \alpha_i - \alpha_j, \tag{1.1.2}$$

where the $\alpha' s$ are unknown parameters, which imply an ordering of the treatments or items on a linear scale defined by the probit and logit links, respectively.

1.1 Previous Research

Many models, both parametric (Thurstone-Mosteller and Bradley-Terry) and non-parametric (see for examples, David, 1988, Kendall and Babington Smith, 1940, and Kendall, 1955), have been proposed over the years for the analysis of paired comparisons designs. However, many extensions for the Bradley-Terry and Thurstone-Mosteller Models often assume a completely balanced design and the "abundance of [non-parametric] methods becomes dearth when the paired comparison experiment is not balanced" (David, 1987). This thesis deals only with the parametric models mentioned above. While these parametric models can be used with unbalanced data, many of the extensions for unbalanced data still require a great deal of replication of the comparisons that are observed. In addition, the response analyzed over the years has been the proportion of times item *i* is preferred over item *j* rather than the binary response resulting from each individual comparison. Balanced designs greatly restrict the number of items that can be compared while the analysis of proportions potentially introduces extra-binomial variation into the models.

The first extensions to the Thurstone Model were made by Mosteller. He demonstrated that, in balanced designs, the estimates proposed by Thurstone could be derived using Ordinary Least Squares (Mosteller, 1951a). He also showed the consequences of assuming a constant variance among the items' performance variables when, in fact, at least one variance is quite different (Mosteller, 1951b). Sandasivan (1982) extended the Thurstone-Mosteller Model to unbalanced data. Dykstra (1960) gave a typical example of paired comparison data and extended the Bradley-Terry Model to include unbalanced data. Other early extensions focused on the inclusion of ties in both the Bradley-Terry Model (Davidson 1970, Rao and Kupper 1967) and the Thurstone-

Mosteller Model (Glenn and David 1960, Greenberg, 1965). In the latter, the inverse sine transformation was used, thereby limiting its effectiveness to large samples. Of the two methods proposed for the former, Davidson's seems to be preferred. Under the Davidson Model the probability of a tie is maximized when two items of equal "ability" are compared. Nevertheless, both versions give very similar results (Bradley, 1984).

A second class of extensions involved the order in which the items were presented for comparison. Several methods have been proposed in attempts to determine if the order of presentation influences the judges. Beaver and Gokhale (1975) proposed an additive order effect for the Bradley-Terry Model, while Davidson and Beaver (1977) extended the Bradley-Terry Model to include both ties and a multiplicative order effect. The latter has several advantages "though explicit methodology [for its use] does not appear in the statistical literature" (Bradley, 1984). Harris (1957) extended the Thurstone-Mosteller Model to include order effects.

Numerous articles proposing goodness-of-fit tests have also been published for both models. Such tests are based primarily on the use of large sample chi-square approximations and assume no further structure to the data beyond the proportion of times item i is preferred to item j. Davidson and Bradley (1970) used large sample theory to obtain variance and covariance's for the strength parameter estimates for general n_{ij} and Raghavarao (1971, Sections 4.3 and 4.5) showed that the efficiency of a paired comparison model to the usual ANOVA for a balanced incomplete block design is $\frac{2}{\pi}$.

Several of the above methods require the repeated observation of all possible comparisons. It has been shown that, with enough replication, the same rules used in the formulation of balanced incomplete block and partially balanced incomplete block designs can be applied to create incomplete paired comparison designs. Methods of analysis for these incomplete designs include the use of m standards, in which all comparisons involving the standards are made, the choice of non-overlapping subsets in which analysis is conducted on the individual subsets, and the creation of subsets which overlap to some degree and are then "linked" together to provide an overall ranking over all the subsets (see for example Torgerson, 1960, Chapter 9, Section 7; Uhrbrock and Richardson, 1933; McCormick and Bachus, 1952).

More recently, covariates have been introduced into both the Bradley-Terry and Thurstone-Mosteller Models. Analysis of the case in which the items represented a factorial treatment combination was solved by Bradley and El-Helbawy (1976) for the Bradley-Terry Model. They also demonstrated how possible interactions could be tested using orthogonal contrasts. Abelson and Bradley (1954) presented the first application of paired comparisons to a $2x2$ factorial treatment structure. Critchlow and Fligner (1991) discussed covariates, including order effects, in both the Bradley-Terry and Thurstone-Mosteller Models. They also pointed out the relationship between these two models and logistic and probit regression, respectively. Others (Fienberg and Larntz, 1976; Fienberg, 1979; Sinclair, 1982; Lindsey, 1989) have illustrated the analysis of the Bradley-Terry Model using log-linear models and shown that a consequence of this formulation is the simple estimation of parameters using standard statistical packages. Kousgaard has also published papers on the inclusion of covariates in the Bradley-Terry Model (1979, 1984).

Burros (1951, 1954) proposed solutions to Thurstone's Case IV and Case III models, respectively. Thurstone (1932) had also derived an approximate solution to the Case IV Model based on analytic geometry arguments. The methods of both Burros and Thurstone are limited to analysis in which a straight ranking is the objective. Thus, the models contain no covariates. All estimates of the variances are functions of the empirical probit values. Two of the estimates (Burros, 1954; Thurstone, 1932) can yield negative estimates of the variance and only Burros (1954), in working with the Case III Model, does not require that the unknown variances be of the same order of magnitude. Gibson (1953) provided a least squares estimate for the Case IV Model but concluded that the computational labor involved in solving the system of equations, especially if the number of items being compared is greater than five, makes the result more of a theoretical exercise than a useful tool. Finally, all the above cases require that a completely balanced design be used and each comparison be made on numerous occasions, so that the observed proportions are stable.

1.2 Thurstone-Mosteller and Bradley-Terry Models

1.2.1 The Thurstone-Mosteller Model

As mentioned above, the Thurstone Model was originally developed for the analysis of psychological scaling problems. For instance, rankings were made of 20 specified crimes in the order of severity from worst to least (according to public sentiment). From the 20 crimes 190 pairs were formed, representing all possible sub-groups of two that can be formed from a group of 20. N people are selected, and each person identifies the worst crime in each of the 190 pairs. Since there is no objective scale that can be used to rate each crime, Thurstone believed the decision to choose one crime over the other arose from sensations created in the observer. The item which created the stronger sensation was subsequently chosen. This is known as Thurstone's *law of comparative judgment.* Briefly, the law of comparative judgment is defined through the following postulates:

> (1) Each of the stimuli [items] follows a "discriminal process" which can be assigned a value on the "psychological continuum of interest." The item that registers a higher value is the one that is judged "favorably" by the observer.
> (2) Random fluctuations are associated with each discriminal process so that the same comparisons do not always result in the same outcome. Further, each of the discriminal processes can be described according to a normal distribution. As such, the probability that item i is preferred over item j can be defined by the cumulative distribution function of a standard normal random variable.
> (3) The mean and standard deviation of each discriminal process are defined as the scale value and discriminal dispersion respectively. The scale values are used to rank the r items.
>
> (Torgerson , 1960)

After all observers, who are typically assumed to be unbiased and able to judge each comparison independently, have made their judgments, the proportion of times item i is favored over item j is used to provide a ranking of the objects.

Thurstone's *law of comparative judgment* can be stated mathematically as

$$R_i - R_j = z_{ij}\sqrt{\sigma_i^2 + \sigma_j^2 - 2r_{ij}\sigma_i\sigma_j} \tag{1.2.1}$$

where
R_i and R_j $=$ the mean values attached to the discriminal processes for objects i and j respectively,
z_{ij} $=$ the distance or deviation from a standardized normal distribution (calculated from the observed proportions),
σ_i and σ_j $=$ discriminal dispersions attached to the discriminal processes for objects i and j respectively, and
r_{ij} $=$ coefficient of correlation between R_{hi} and R_{hj}
(Guilford, 1954).

In order deal with the unknown parameters, $\sigma_i, \sigma_j, R_i, R_j$ and r_{ij}. Thurstone made several assumptions which simplified (1.2.1). The five models, corresponding to various assumptions which were proposed by Thurstone are defined as:

(1) *Case I Model:* (1.2.1) applied in its complete form with a single observer making all comparisons many times.
(2) *Case II Model:* The same as *Case I* except that many observers are used and each observer examines each of the $\binom{r}{2}$ comparisons a single time.
(3) *Case III Model:* Assume that $r_{ij} = 0$.
(4) *Case IV Model:* Assume that the discriminal dispersions are approximately equal, i.e. $\sigma_i \approx \sigma_j$ for all i and j.
(5) *Case V Model:* Assume that $r_{ij} = 0$ and let $\sigma_i^2 = \sigma_j^2$ for all i and j.

Mosteller (1951a) noted that only the assumption of equal correlation is required for the estimation of the model parameters in the Case V Model. Most analysis and research has been based on the Case V Model in which the common σ^2 is taken as the scale unit and usually set equal to unity. Alternatively $\sqrt{2\sigma^2}$ can be used as the unit scale and set to unity.

1.2.2 *The Bradley-Terry Model*

The Bradley-Terry Model (1952) was developed as an extension to the usual binomial model when only two items are being compared, otherwise known as the sign test (Bradley, 1984). Estimation and testing procedures were developed using the likelihood under the binomial assumptions. It can be shown that the probability that item i is preferred over item j, $P(R_i \rightarrow R_j)$, equals $\pi_i/(\pi_i + \pi_j)$ where the $\pi_i's$ are the unknown parameters, subject to the restriction $\sum_{i=1}^{r}\pi_i = 1$. (Zarmelo, in 1929, actually first postulated such a model for chess data. The model was independently discovered by Bradley and Terry.)

The Bradley-Terry Model can also be defined analogously to the Thurstone-Mosteller Model by assuming the discriminal processes follow an extreme-value distribution rather than a normal distribution. Consequently, the proportion of times item i is preferred over item j is described by the logistic density.

1.2.3 Linear Models

Both the Bradley-Terry Model and the Thurstone-Mosteller Model are special cases of a linear model which has the form

$$P(R_i \rightarrow R_j) = H(V_i - V_j) \qquad (1.2.2)$$

where H is a symmetric distribution function. In the case of the Bradley-Terry Model

$$P(R_i \rightarrow R_j) = \frac{1}{4}\int_{-(log\pi_i - log\pi_j)} \frac{sech^2 y}{2} dy \qquad (1.2.3)$$

and for the Thurstone-Mosteller Model

$$P(R_i \rightarrow R_j) = \int_{-(\mu_i - \mu_j)}^{\infty} exp^{-\frac{y^2}{2}} dy. \tag{1.2.4}$$

Latta (1979) has shown that both models give similar results. However, they can differ when comparisons result in proportions that are outside the range of 0.1 to 0.9. Jackson and Fleckenstein (1957) also published a comparison of the two models.

1.3 Purpose of Thesis

This thesis, composed of an introduction, two self-contained articles (Chapters 2 and 3, respectively), and a conclusion, addresses the following issues and extensions for the Thurstone-Mosteller Model:

(1) The class of covariates that can be included in the Thurstone-Mosteller Model will be formally extended to include time-dependent variables, i.e. covariates that are specific to an individual *comparison* and change values over time. In addition, these covariates can be used in both balanced and unbalanced designs. Of particular interest is the formal analysis of these models using generalized linear model routines which are available in all the standard statistical packages. Furthermore, the inclusion of a dispersion parameter in the logit or probit regression model and subsequent use of quasi-likelihood analysis offer a convenient approach for retaining the simplicity of the Bradley-Terry and Thurstone-Mosteller Models while making adjustments for minor model inadequacies. The modeling techniques that are to be illustrated are not profound. However, because the bulk of the research for paired comparisons methodology was carried out before computer programs for generalized linear models became popular, their use of the analysis of the Thurstone-Mosteller and Bradley-Terry Models has been addressed in limited detail only.

(2) Previous extensions to the Thurstone-Mosteller Model have generally applied to Thurstone's Case V Model, in which the performance variables have a constant variance which is usually scaled to unity. The EM Algorithm (Dempster, Laird and Rubin, 1977)

will be used to compute maximum likelihood estimates (M.L.E.'s) of both the parameter estimates and the variances associated with the performance distributions for each of the r items or treatments. Likelihood ratio tests can also be constructed for inferential purposes. To facilitate this the unobserved performance variables will be treated as "missing data."

(3) The models in (1) and (2) assume that comparisons are made independently. The Thurstone-Mosteller Model will be extended to allow the unobserved performance variables to follow an auto-regressive model of order one. Two models, which can be fit with probit regression routines, will also be presented as approximations to the model using the EM Algorithm.

The Thurstone-Mosteller Model, rather than the Bradley-Terry Model, is used for transparency and convenience. The model is conveniently described in terms of a latent, normally-distributed performance variable, W, for each trial of each experiment, which has a linear regression on the explanatory variables. Although the existence of such a latent variable is not required (it is more of a device) the formulation provides a natural way to interpret the results that parallels ordinary regression analysis. In addition, this formulation is a simple format for maximum likelihood estimation via the EM Algorithm. Although there is a specific distributional assumption involved - the normality of the latent performance variables - it is not expected that robustness to this distribution will be a serious issue since the tails of the distribution are relatively unimportant for this purpose.

The first part of Chapter 2 will focus on the use of generalized linear models to analyze both the Thurstone-Mosteller and Bradley-Terry Models. Data from the 1993-94 National Basketball Association season will be used to illustrate the full use of generalized linear models in the analysis of both the Thurstone-Mosteller and Bradley-Terry Models. Emphasis will be given to the use of unbalanced data, time-dependent covariates, and quasi-likelihood.

The latter part of Chapter 2 will extend the use of the Thurstone-Mosteller Model to cases in which the variances of the performance variables are not equal. The EM Algorithm will be used to estimate the maximum likelihood estimates of these variance parameters. Full likelihood analysis will be discussed for hypothesis testing. Data from the 1993 Major League Baseball Season will be used to illustrate these models.

Chapter 3 will address the assumption of the independence of the performance variables. The EM Algorithm will again be used to estimate the correlation between the performance variables under the assumption that the performance variables follow an auto-regressive model of order one. Two additional techniques, which can be fit with a standard probit regression routine, will also be introduced, as convenient approximations to the estimates provided by the EM Algorithm.

Chapter 2

Thurstone's Case III Model for Paired Comparisons

James D. Kolsky and Daniel W. Schafer

Submitted to *Biometrics*,
International Biometric Society, Washington D.C.
January 1996.

2. *Thurstone's Case III Model for Paired Comparisons*

Abstract

The Thurstone-Mosteller and Bradley-Terry Models are commonly used to rank items from paired comparisons experiments in which one item in each pair "wins," and to assess the importance of explanatory variables on such rankings. This paper clarifies the use of probit and logistic regression models for such designs, including the incorporation of time-dependent explanatory variables and the analysis of unbalanced designs. In addition, likelihood inference, using the EM Algorithm, is proposed for Thurstone's Case III Model, which includes variance parameters to account for variable item performances.

Keywords: Bradley-Terry Model, EM Algorithm, generalized linear model, logistic regression, probit regression, Thurstone-Mosteller Model.

2.1 Introduction

"Paired comparisons" describes a data structure in which r items or individuals are compared in pairs. A completely balanced paired comparison design can be thought of as an incomplete block design with block size of 2 and $\binom{r}{2}$ blocks (Bradley, 1984). The term "paired comparisons," however, usually implies a single, binary response for each block. That is, one of the items in each block wins or is preferred to the other. In many cases, it is of interest to rank the items or to investigate explanatory variables that may affect the ranking. A food preference study, for example, may be conducted in order to rank six pies based on taste, or to model the taste preference as a function of the amount of sugar in the pies.

An extensive body of literature exists regarding the design and analysis of paired comparison experiments. Important summaries are the book by David (1988) and the article by Bradley (1984). Extensive bibliographies on the subject are provided by David and also by Davidson and Farquar (1976).

The two most popular models are the Thurstone-Mosteller Model (Thurstone, 1927; Mosteller, 1951a,b,c) and the Bradley-Terry Model (Bradley and Terry, 1952). Let π_{stk} represent the probability that item s is preferred to item t at the time of their k^{th} comparison. The Thurstone-Mosteller Model can then be expressed as

$$\Phi^{-1}(\pi_{stk}) = \alpha_{sk} - \alpha_{tk} \tag{2.1.1}$$

and the Bradley-Terry Model as

$$log\left(\frac{\pi_{stk}}{1-\pi_{stk}}\right) = \alpha_{sk} - \alpha_{tk} \tag{2.1.2}$$

where $\Phi()$ is the standard normal distribution function and the $\alpha_i's$ are unknown parameters, sometimes called "merit, " "worth, " or "strength" parameters. These parameters, if independent of k, imply a linear ordering of the items. This paper discusses maximum likelihood estimation when (1) the effects of explanatory variables are modeled as $\alpha_{sk} = \beta' X_{sk}$, and (2) the Thurstone-Mosteller Model is extended so that differing variability of item performance is allowed, i.e. when

$$\Phi^{-1}(\pi_{stk}) = \frac{\alpha_{sk}-\alpha_{tk}}{(\sigma_s^2+\sigma_t^2)^{\frac{1}{2}}}. \tag{2.1.3}$$

Regarding the inclusion of explanatory variables, Section 2.2 emphasizes the ease with which these models can be fit using ordinary probit and logistic regression. Although this must be known to many who work with paired comparison designs, it is not well documented in the statistical literature. In particular, the analysis is straightforward with unbalanced as well as balanced designs, both time-independent and time-dependent explanatory variables can be modeled, and the quasi-likelihood approach can be used to compensate for minor model inadequacies.

Suppose, for example, that vectors of explanatory variables associated with items s and t are observed at the time of their k^{th} comparison and labeled as X_{sk} and X_{tk}. The Thurstone-Mosteller Model, in (2.1.1), then implies

$$\Phi^{-1}(\pi_{stk}) = \beta' X_{sk} - \beta' X_{tk} = \beta'(X_{sk} - X_{tk}). \tag{2.1.4}$$

It is not difficult to see how maximum likelihood estimates of the unknown vector of regression coefficients, β, can be computed using a probit regression routine. Similarly, the Bradley-Terry Model can be expressed as

$$log\left(\frac{\pi_{stk}}{1-\pi_{stk}}\right) = \beta'(X_{sk} - X_{tk}), \tag{2.1.5}$$

and β can be estimated using a logistic regression routine.

The model defined by (2.1.3) was first proposed by Thurstone (1927, 1932) and is sometimes referred to as Thurstone's Case III Model. Rationale for this model will be discussed in Section 2.3.1. Previous estimation procedures for this model have not been practically useful (Thurstone, 1932; Gibson, 1953; Burros, 1951 and 1954). It is shown in Section 2.3, however, that relatively straightforward computations necessary for likelihood analysis of this model can be accomplished with the EM Algorithm (Dempster, Laird and Rubin, 1977).

2.2 Explanatory Variables in the Thurstone-Mosteller and Bradley-Terry Models

Atkinson (1972) previously demonstrated the use of the linear logistic model to analyze paired comparison data according to the Bradley-Terry Model. Others (Fienberg and Larntz, 1976; Fienberg, 1979; Sinclair, 1982; Lindsey, 1989) have illustrated the analysis of the Bradley-Terry Model using log-linear models and shown that a consequence of this formulation is the simple estimation of parameters using standard statistical packages.

However, paired comparison data is more appropriately defined as proportions rather than counts. Critchlow and Fligner (1991) proposed the use of generalized linear models, namely logistic and probit regression, to analyze the Bradley-Terry and Thurstone-Mosteller Models, respectively, for the special case of balanced designs and time-independent covariates. This section extends Critchlow and Fligner's approach. Several examples are used demonstrating some rather obvious extensions to include over-dispersion, unbalanced designs, and time-dependent covariates.

2.2.1 Applesauce Taste Preference Experiment

Atkinson (1972) presented data from a small, completely balanced, paired comparisons experiment on the effect of monosodium glutamate (MSG) on applesauce taste. Four preparations of applesauce, corresponding to MSG levels 0, 1, 2, and 3, were compared pairwise. The number of times that preparation s was preferred to preparation t, out of four independent comparisons, is shown in Table 2.1. The four preparations correspond to applesauce with increasing amounts of MSG.

Table 2.1: Atkinson Data

The entry in row s *and column* t *is the number of times that applesauce preparation* s *was preferred to preparation* t *out of four comparisons, for preparations with no (0), low (1), medium (2), and high (3) additions of MSG.*

	MSG Level		
	0	***1***	***2***
1	3		
2	4	1	
3	0	1	1

When ranking the four treatments, the Bradley-Terry Model describes the probability that preparation s is preferred to preparation t, π_{st}, as a function of the preference parameters for each preparation. A logit version of the model is

$$log\left(\frac{\pi_{st}}{1-\pi_{st}}\right) = \alpha_s - \alpha_t, \tag{2.2.1}$$

for $s, t = 0, 1, 2, 3$; with the constraint that $\alpha_0 = 0$.

2.2.1.1 Logistic Regression

To cast this in the generalized linear model framework, the six entries in Table 2.1 are independent binomial observations, $Y_i \sim Bin(4, \pi_i)$, where π_i is the probability that the row preparation is preferred over the column preparation, and

$$log\left(\frac{\pi_i}{1-\pi_i}\right) = \alpha_1 X_{i1} + \alpha_2 X_{i2} + \alpha_3 X_{i3}, \tag{2.2.2}$$

for $i = 1, ..., 6$, where X_{ij} equals 1 if MSG preparation level j is the row preparation in the i^{th} binomial observation, -1 if MSG preparation level j is the column preparation, and 0 otherwise. The maximum likelihood estimates of the preference parameters, from fitting the logistic regression model in (2.2.2) (without an intercept), are shown in Table 2.2.

Table 2.2
Parameter Estimates and Standard Errors for Applesauce Taste Experiment

MSG level	$\hat{\alpha}$	$SE(\hat{\alpha})$
0 None	*0.00*	----
1 Low	*1.21*	*0.84*
2 Medium	*0.89*	*0.81*
3 High	*-1.00*	*0.87*

The deviance goodness-of-fit test, for the adequacy of the model, is the likelihood ratio test comparing the Bradley-Terry Model above to the 6-parameter model with separate relative preference probabilities for each pair. If the number of comparisons for each pair of items is large enough, this test should detect inadequacies of the assumption of a one-dimensional linear ordering of the preparation preferences as well as inadequacies of the binomial assumption, such as a lack-of-independence of the Bernoulli trials due to differences between judges. With a binomial index of only 4 in this problem, the test may be used with caution. The deviance statistic is 6.936 on 3 degrees of freedom for a p-value of 0.074.

2.2.1.2 Quasi-likelihood

Two possible actions for dealing with a poor fit are (1) attempting to use the Thurstone-Mosteller Model, which results in a linear ordering on an alternate scale, or (2) inclusion of a dispersion parameter and use of quasi-likelihood to account for all the sources of overdispersion. The former merely implies the use of the probit link rather than the logit link, and is unlikely to provide any difference in fit unless the relative preference probabilities are close to zero or one. The latter provides a simple way to account for judge effects, non-independence of trials, or other unmodeled sources of variation, without abandoning the simple form of the Bradley-Terry (or Thurstone-Mosteller) Model. The quasi-likelihood specification replaces the binomial assumption with the assumption that $Var(Y_i) = m_i \phi \pi_i (1 - \pi_i)$, where ϕ is the additional dispersion parameter (McCullagh and Nelder, 1989, Sect. 4.5). The main effect of the quasi-likelihood analysis is the inflation of the standard errors to account for the extra-binomial variation. If the assumption of a linear ordering on some one-dimensional scale is a minor model inadequacy, then quasi-likelihood analysis with the Bradley-Terry or Thurstone-Mosteller Models may be used to maintain the one-dimensional ranking, with the more complicated aspects of ordering absorbed into "extra-binomial variation."

The estimated parameters of the model in Table 2.2 and the results of the deviance "goodness-of-fit" test suggest a linear ordering of the taste preference of the four preparations, as shown in Figure 2.1.

Figure 2.1
The estimated linear ordering and the Bradley-Terry preference scores for the four MSG additions

---

```
High MSG        No MSG      Med MSG  Low MSG
 —*—————————————*————————————*————*————————> better taste
 -1.00          0.00         0.89  1.21
```

Apparently, the low addition of MSG improves the taste of applesauce, the medium level also improves the taste, though not as much as the low level, and the high addition preparation tastes worse than the preparation with no MSG at all. Statements about the relative preferences may be conveniently expressed in terms of odds ratios. For example, it is estimated that the odds the low MSG applesauce is preferred to the medium MSG applesauce is $exp(1.21 - 0.89)$, or 1.38. Or, roughly, there are 7 judges who prefer the low MSG for every 5 judges who prefer the medium MSG. (An approximate 95% confidence interval for the odds is 0.18 to 10.70 using the quasi-likelihood standard errors.).

2.2.1.3 Optimal Level

If the levels of MSG are equally-spaced amounts further modeling may be appropriate. In particular, if the linear preference scores are quadratic in the amount of added MSG then

$$\alpha_s = \beta_0 + \beta_1 MSG_s + \beta_2 MSG_s^2, \tag{2.2.3}$$

and

$$log\left(\frac{\pi_i}{1-\pi_i}\right) = \beta_1(Row\,MSG_i - Column\,MSG_i) + \beta_2\left[(Row\,MSG_i)^2 - (Column\,MSG_i)^2\right] \tag{2.2.4}$$

for $i = 1, ..., 6$. This model may be compared to the more general model (2.2.2) above through the significance of an additional cubic term. The p-value from the likelihood ratio test is 0.975 indicating no problem with the quadratic-in-MSG model. Parameter estimates from the quadratic model are $\widehat{\beta}_1 = 1.996$ $(S.E. = 0.98)$ and $\widehat{\beta}_2 =$ -0.775 $(S.E. = 0.33)$. According to this model, preference for applesauce is maximized when MSG is added in the amount $-\beta_1/(2\beta_2)$, which is estimated to be 1.29 units of MSG. (95% confidence interval is 0.53 to 1.63; obtained by inverting the likelihood ratio test for the hypothesis $-\beta_1/(2\beta_2) = C$, or equivalently for the linear hypothesis $2\beta_2 C + \beta_1 = 0$). See Figure 2.2.

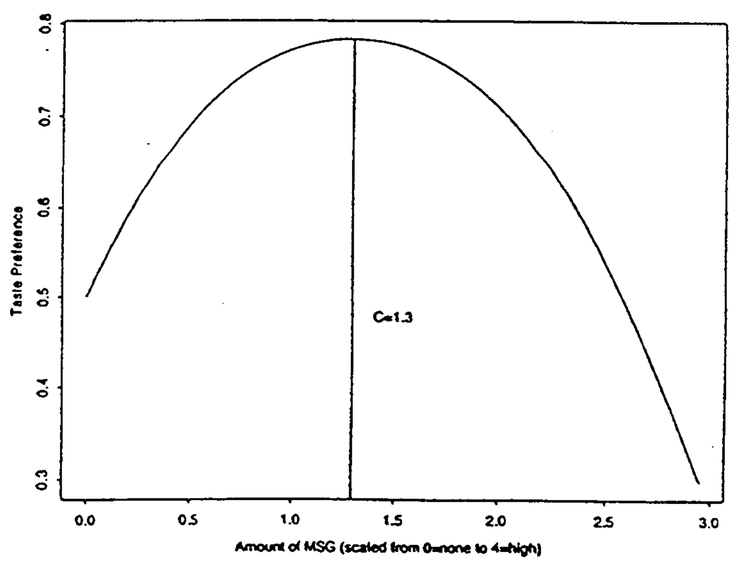

Figure 2.2
Estimated taste preference when modeled as a quadratic function of MSG

2.2.1.4 Order Effect

Although the order in which the applesauce preparations are presented to the judges is unknown in this particular experiment, it is often useful to incorporate an order effect into the model. If the rows represent the preparation which is given first to the judges, and the columns represent the preparation given second, then the data would be displayed using all 12 off-diagonal cells in a 4x4 table, rather than just the six cells displayed in Table 2.1. The model for preference probabilities could then include an order effect:

$$log\left(\frac{\pi_{st}}{1-\pi_{st}}\right) = \beta_0^* + \alpha_s - \alpha_t, \qquad (2.2.5)$$

where β_0^* represents the additional log odds of preferring one preparation over another simply because it was the first one presented in the pairwise trial. Each of the logistic models above could be modified to include an order effect by the addition of an intercept term.

2.2.1.5 Summary

It should be emphasized that the modeling techniques illustrated in this example are not profound. However, because the bulk of the research for paired comparisons methodology was carried out before computer programs for generalized linear models became popular, their use for analysis of the Bradley-Terry and Thurstone-Mosteller Models has been addressed in limited detail only. Currently important discussions about maximum likelihood fitting of preference parameters in the Bradley-Terry and Thurstone-Mosteller Models have focused on using marginal totals from balanced paired comparisons experiments (David, 1988, Sect. 4.3). Consequently, modeling has been restricted unnecessarily to cases in which the parameters can be estimated by linear contrasts of marginal totals from balanced data (Bradley, 1984; Bradley and El-Helbawy, 1976; Critchlow and Fligner, 1991). Probit and logistic regression, on the other hand, offer a convenient, unified, and flexible approach for all aspects of paired comparisons analysis. Furthermore, the inclusion of a dispersion parameter in the logit or probit regression model and subsequent use of quasi-likelihood analysis offer a convenient approach for retaining the simplicity of the Bradley-Terry and Thurstone-Mosteller Models while making adjustments for minor model inadequacies.

2.2.2 Ranking and Modeling Superiority in the National Basketball Association

Data are available as indicator variables for home team victory and various explanatory variables associated with each game of the 1993-94 National Basketball Association season. (The data was provided by the National Basketball Association.) The "items" in this example are the 27 teams. The data are unbalanced since each team played each other team either 2 or 5 times. Part of the analysis that follows illustrates the ranking of teams after accounting for explanatory variables, such as home court advantage (analogous to the order effect above). Further analysis considers factors that may influence team rankings, such as team-specific home court advantages and attendance. Although numerical scores for each game are available, there is some controversy as to the relevance of this additional information. For the purposes of illustration in this paper, only the win-loss outcomes will be used.

A starting point is a simple ranking of the teams based on the Bradley-Terry Model. Let π_{st} represent the probability that team s defeats team t. Thus,

$$log\left(\frac{\pi_{st}}{1-\pi_{st}}\right) = \alpha_s - \alpha_t, \qquad (2.2.6)$$

where the $\alpha' s$ are the "strength parameters" used to rank the teams. Define Y_i equal to 1 if the home team won the i^{th} game in the list $(i = 1, ..., 1107)$ and let π_i be the probability that the home team wins, then (2.2.6) may be re-expressed as

$$log\left(\frac{\pi_i}{1-\pi_i}\right) = \sum_{j=2}^{27} \alpha_j X^*_{ij} \tag{2.2.7}$$

where $X^*_{ij} = 1$ if team j was the home team in the i^{th} game, -1 if team j was the visiting team in the i^{th} game, and 0 otherwise. To ensure identifiability, the constraint $\alpha_1 = 0$ is used in fitting the model. (The model is unchanged by the addition of a constant to all the $\alpha' s$.) Maximum likelihood fitting of the logistic regression model with 26 strength parameters to distinguish the 27 teams required 3 Fisher Scoring iterations and produced the ranking and estimates shown in Table 2.3.

Table 2.3
Rankings based on Bradley-Terry strength parameters (α_j) for the 1993-94 NBA Season

Team	α	*Team*	α	*Team*	α
Dallas	0.00	L.A. Lakers	1.39	Golden State	2.31
Milwaukee	0.52	Charlotte	1.76	Utah	2.48
Detroit	0.53	Miami	1.80	Chicago	2.52
Minnesota	0.54	Denver	1.87	San Antonio	2.58
Washington	0.79	New Jersey	1.98	New York	2.63
Philadelphia	0.85	Cleveland	2.05	Atlanta	2.64
L.A. Clippers	1.04	Indiana	2.09	Phoenix	2.65
Sacramento	1.10	Portland	2.15	Houston	2.77
Boston	1.24	Orlando	2.24	Seattle	3.09

Further study of the effect explanatory variables have on the strength parameters can be accomplished through linear models. If π_i represents the probability that the home team wins the i^{th} game in the list,

$$log\left(\frac{\pi_i}{1-\pi_i}\right) = \beta'(X_i^H - X_i^V) = \beta' X_i^* \tag{2.2.8}$$

where X_i^H and X_i^V are vectors of explanatory variables observed for the home and visiting teams, respectively for game i. In the simple ranking model above, $X_{ij}^H = 1$ if team j is the home team in the i^{th} game, for $j = 2, ..., 27$, and $X_{ij}^V = 1$ if team j is the visiting team in the i^{th} game. A model that includes a home court effect, common to all teams, defines $X_{i,28} = 1$ if team j is the home team. In other words,

$$log\left(\frac{\pi_i}{1-\pi_i}\right) = \sum_{j=2}^{27}\beta_j\left(X_{ij}^H - X_{ij}^V\right) + \beta_{28}\left(X_{i,28}^H - X_{i,28}^V\right). \tag{2.2.9}$$

Since $X_{i,28}^H = 1$ and $X_{i,28}^V = 0$ for every i, the last term is incorporated in the logistic regression model as an "intercept." The estimates from this model are shown in Table 2.4. Notice that the range of the estimated "strength" parameters is a bit wider than the range resulting from the previous model. This suggests a better discrimination between teams is achieved after accounting for the home court advantage. Note, however, that the ranking of the teams is identical to the one obtained by ignoring home court advantage. The effect of the home court advantage is estimated to be 0.61 (SE = 0.07). Thus, the odds that a team beats an opponent at home is estimated to be 1.84 times the odds that the team beats the same opponent on a neutral court. (An approximate 95% confidence interval for the odds is 1.60 to 2.12.)

Different home-court advantages for the 27 teams can be modeled with

$$log\left(\frac{\pi_i}{1-\pi_i}\right) = \sum_{j=2}^{27}\beta_j\left(X_{ij}^H - X_{ij}^V\right) + \sum_{j=1}^{27}\beta_{j+27}X_{ij}^H = \beta' X_i^*. \tag{2.2.10}$$

Here, $\beta' = (\beta_2, ..., \beta_{54})$ and $X_i^* = \left(X_{ij}^H - X_{ij}^V,\ X_{ij}^H\right)$. The log odds that team s defeats team t when team s is the home team is $\beta_s - \beta_t + \beta_{s+27}$. The log odds that team s defeats team t when team s is the visiting team is $\beta_s - \beta_t - \beta_{t+27}$. To ensure identifiability, the models here are fit under the constraint $\beta_{28} = 0$, where the coefficient

β_{28} represents the home court advantage of team 1. β_{s+27} represents the difference in log odds of the team s home court advantage from the team 1 home court advantage. Estimates are shown in Table 2.5. The likelihood ratio test statistic for equality of home court advantage parameters is 25.736 on 25 degrees of freedom, for a p-value of 0.42.

Table 2.4
NBA Rankings for 1993-94 Season, after accounting for Home Court Advantage

Team	α	*Team*	α	*Team*	α
Dallas	0.00	L.A. Lakers	1.48	Golden State	2.47
Milwaukee	0.55	Charlotte	1.89	Utah	2.66
Detroit	0.56	Miami	1.92	Chicago	2.70
Minnesota	0.57	Denver	2.00	San Antonio	2.77
Washington	0.84	New Jersey	2.12	New York	2.82
Philadelphia	0.89	Cleveland	2.20	Atlanta	2.83
L.A. Clippers	1.11	Indiana	2.24	Phoenix	2.85
Sacramento	1.17	Portland	2.30	Houston	2.97
Boston	1.33	Orlando	2.40	Seattle	3.31

Home Court Advantage: 0.61

The additional effect of attendance on the home court advantage may be modeled by the inclusion of the term $\beta_{55}\text{Attendance}_i$. The effect of the team's total salary expenditure can be investigated through the additional term $\beta_{56}(Salary_i^H - Salary_i^V)$. Full modeling of time dependent and time independent explanatory variables can be accomplished with logistic regression (for the Bradley-Terry Model) and probit regression (for the Thurstone-Mosteller Model).

Table 2.5
Individual Home Court Advantages

Team	β_{0i}	*Team*	β_{0i}	*Team*	β_{0i}
Dallas	-1.83	Chicago	-0.70	Utah	0.00
Detroit	-1.51	L.A. Clippers	-0.63	Sacramento	0.00
Miami	-1.44	San Antonio	-0.59	Washington	0.00
Milwaukee	-1.28	Minnesota	-0.58	New Jersey	0.02
Atlanta	-1.06	L.A. Lakers	-0.55	Denver	0.05
Boston	-1.04	Indiana	-0.36	Seattle	0.13
Philadelphia	-0.93	Orlando	-0.17	Charlotte	0.22
Golden State	-0.79	Portland	-0.12	Cleveland	0.27
New York	-0.75	Houston	-0.05	Phoenix	0.53

2.2.3 Identifiability

In some circumstances certain parameters will not be identifiable. Conditions leading to this problem are easily defined:

(1) For models in which no covariates are included, the design must be connected. If it is possible to divide the teams into disjoint groups in which none of the teams in one group compete against the teams in another group, there is no basis for a unified ranking of the teams belonging to the two separate groups. Likewise, if the teams in one set always defeat the teams in another set, estimates for the latter are necessarily zero (David, 1988).

(2) For models that include covariates, identifiability still hinges on the requirements above. In addition, the covariates must satisfy the latter condition in (1). For instance, when estimating the model in (2.2.10), each team must win and lose at least once both at home and on the road.

2.3 Thurstone's Case III Model

2.3.1 Latent Performance Variables

Thurstone's models (1927) were motivated by the consideration of an underlying, continuous "performance" variable for each item. Consider, for example, modeling the outcome of a basketball game between teams s and t through the selection of random performance variables W_s and W_t from continuous "performance distributions" for each team. This performance can be thought of as an abstraction which cannot be measured, although it could be thought of as the sum total of all decisions and actions that will occur during the course of the competition (Elo, 1978). Team s wins the game if $W_s > W_t$ and loses otherwise. The Thurstone-Mosteller Model in the previous section is a consequence of assuming that W_s is normally distributed with mean α_s and variance 1, for $s = 1, ..., r$. The Bradley-Terry Model follows from the assumption that W_s has a logistic distribution.

Thurstone (1927) originally proposed more general versions of the model. His Case III, for example, had $W_s \sim N(\alpha_s, \tau_s)$. The different performance distribution variances may be used to model different variability of performance of the teams (or items) about their long-term mean performance. For instance, a team with a large τ_s might occasionally out-perform a team with a larger α, but might also have a good chance of losing to a team with a smaller strength parameter. There are two reasons for considering the model with unequal variances: (1) as a check to ensure the analysis using the Case V Model is valid, and (2) because there may be some interest in comparing variances, which may be thought of as representing "consistency of performance."

In Thurstone's Case III model the probability that item s is preferred to item t is

$$Pr(W_s > W_t) = \Phi\left(\frac{\alpha_s - \alpha_t}{\sqrt{\tau_s + \tau_t}}\right). \tag{2.3.1}$$

When the $\tau's$ are unknown the data can no longer be analyzed as a generalized linear model because $\Phi^{-1}(\pi_{st}) = (\alpha_s - \alpha_t)/(\tau_s + \tau_t)^{\frac{1}{2}}$ is not linear in the parameters. Previous attempts at estimation (Thurstone, 1932; Burros, 1951; Burros and Gibson, 1954) are unsatisfactory since they apply only to large, completely balanced designs without covariates; and they involve various, very rough approximations. All of these methods involve the empirical probit transformation of the observed proportions. Estimates of the $\tau's$ are functions of the sample variances of the columns in the resulting table of probits.

2.3.2 Likelihood Analysis using the EM Algorithm

Data from the 1993 Major League Baseball season (National League only) will be used to illustrate Thurstone's Case III Model. Suppose, once again, that associated with each game (each comparison) is the binary outcome Y_i, taken to be 1 if the home team won (or, more generally, if the first item in the pair is preferred). Also associated with game i are vectors of explanatory variables for the home and visiting teams, X_i^H and X_i^V as defined in the previous section. Suppose that

$$W_i^H \sim N(\beta' X_i^H, \tau' Z_i^H) \tag{2.3.2}$$

and

$$W_i^V \sim N(\beta' X_i^V, \tau' Z_i^V). \tag{2.3.3}$$

where the vector Z_i^H is composed of elements $Z_{ij}^H = 1$ if team j was the home team in game i, and 0 otherwise for $j = 2, \ldots, 27$; Z_i^V is similarly defined to include an indicator of the visiting team. Let the $W_i^{H'}s$ and $W_i^{V'}s$ be mutually independent and suppose that $Y_i = 1$ if $W_i^H > W_i^V$ and 0 otherwise. The model differs from those of the previous section by the inclusion of the 26 extra parameters, τ_j, as performance variances for teams 2 through 27. Team 1 is constrained to have a performance variance of one. In general, such a model could be used to study either varying degrees of consistency among the 27 teams, or to account for such variation while studying the factors which influence team rankings. By treating the latent performance variables as missing data, the EM Algorithm (Dempster, Laird and Rubin, 1977) can be used to obtain maximum likelihood estimates for both β and τ.

Let W be the $(2n)x1$ vector $(W^{H'} : W^{V'})'$, let Y be the $nx1$ vector with elements Y_i, and let X be the $(2n)xp$ matrix $(X^{H'} : X^{V'})'$. Thus, the home teams' explanatory variables are contained in the first n rows of X and the visiting teams' explanatory variables in the last n rows. Also define Z to be the $(2n)x27$ matrix $(Z^{H'} : Z^{V'})$. Then $W \sim N_{2n}(X\beta, V)$ where V is the $(2n)x(2n)$ diagonal matrix with the i^{th} diagonal element equal to $\tau' Z_i$. The "complete data" are Y and W while the "observed data" is Y. The EM algorithm requires calculation in the E-step of

$$Q(\beta,\tau|\beta^{(r)},\tau^{(r)}) = E\Big[l_c(\beta,\tau;Y,W)|Y;\beta^{(r)},\tau^{(r)}\Big], \tag{2.3.4}$$

where $l_c(\beta,\tau;Y,W)$ is the log-likelihood based on the complete data, and $\beta^{(r)}$ and $\tau^{(r)}$ are the parameter estimates of β and τ after the r^{th} iteration. (The parameters involved in the conditional expectation are replaced by their current parameter estimates, $\beta^{(r)}$ and $\tau^{(r)}$.) Parameters are updated in the M-step by $\beta^{(r+1)}$ and $\tau^{(r+1)}$, the β and τ that maximizes (2.3.4). These two steps are repeated until the estimates converge.

Since the conditional distribution of Y given W does not depend on unknown parameters, the complete data log likelihood for β and τ is the density of W as a function of the parameters:

$$l_c(\beta,\tau) = log\Big[f(W;\beta,\tau)\Big] = K - \tfrac{1}{2}log|V| - \tfrac{1}{2}(W - X\beta)'V^{-1}(W - X\beta), \tag{2.3.5}$$

where K is a constant. So equation (2.3.4) becomes

$$Q\left(\beta,\tau|\beta^{(r)},\tau^{(r)}\right) = -\tfrac{1}{2}log|V| - \tfrac{1}{2}\Big\{tr\Big[V^{-1}Var(W|Y,\beta^{(r)},\tau^{(r)})\Big] + \Big[E(W|Y,\phi^{(r)}) - X\beta\Big]' V^{-1}\Big[E(W|Y,\phi^{(t)}) - X\beta\Big]\Big\}. \tag{2.3.6}$$

The functional forms of the conditional moments in (2.3.6), based on the distribution of W_i^H conditional on $W_i^H > W_i^V$ and W_i^H conditional on $W_i^H < W_i^V$, are shown in the appendix.

The M-step may be accomplished with iteratively weighted least squares. Computationally, it is more convenient to use one iteration of the iteratively weighted least square at each EM iteration, as suggested more generally by Meng and Rubin (1993). With this approach the estimates are updated using

$$\beta^{(r+1)} = \left(X'V^{(r)^{-1}}X\right)^{-1}X'V^{(r)^{-1}}E(W|Y;\beta^{(r)},\tau^{(r)}) \tag{2.3.7}$$

and

$$\tau_s^{(r+1)} = (1/n_s)\sum_{i=1}^{2n}\Big\{Var(W_i|Y;\beta^{(r)},\tau^{(r)}) + \Big[E(W_i|Y;\beta^{(r)},\tau^{(r)}) - X_i'\beta^{(r)}\Big]^2\Big\}Z_{is} \tag{2.3.8}$$

where n_s is the number of games involving team s.

2.3.3 Likelihood Ratio Inference

Methods exist to approximate the information matrix (Meng and Rubin, 1991; Louis, 1982) for Wald inferences. Alternatively, the observed log likelihood function can be easily evaluated once the maximum likelihood estimates have been obtained. Note that

$$L(\beta,\tau) = \prod_{i=1}^{n} \pi_i^{Y_i}(1-\pi_i)^{1-Y_i} \tag{2.3.9}$$

with

$$\pi_i = \Phi\left[\frac{\beta'(X_i^H - X_i^V)}{\sqrt{\tau' Z_i^H + \tau' Z_i^V}}\right]. \tag{2.3.10}$$

Therefore, inferences based on likelihood ratios are straightforward in this problem, even though the maximized value of the likelihood function is not a by-product of the EM Algorithm.

2.3.4 Convergence and Initial Estimates

The EM Algorithm requires initial estimates, $\beta^{(0)}$ and $\tau^{(0)}$. Because of the linear convergence of the EM Algorithm, poor initial estimates can seriously delay or even prevent the convergence of the estimates. Good initial estimates for β can be obtained by fitting the model with $\tau_s = 1$ as in the previous section. (Note: due to the scale defined in Section 2.3, the probit regression estimates must be multiplied by $\sqrt{2}$ in order to be equated with the estimates used in this section.) Initial estimates $\tau^{(0)}$ can be computed using the methods for the Case IV Model developed by Burros (1951). (The Case IV Model assumes that item variances are approximately equal). If no covariates are to be included in the model, form an SxS table, where S is the number of items being compared, such that the $(s,t)^{th}$ cell is the proportion of times team s defeated team t. Convert the cell values to their standard normal quantile values (empirical probits) and compute the sample variance, v_i^2, for each of the S columns. The initial estimates are defined by

$$\tau_s^{(0)} = \frac{c}{v_s^2}, \text{ where } c = \frac{S}{\sum_{t=1}^{S}\left(\frac{1}{v_t^2}\right)}. \tag{2.3.11}$$

If covariates are to be included in the model, initial estimates can still be calculated as above, either by temporarily ignoring the covariates, or if replicate comparisons have been observed at each of the possible levels of the covariates, the empirical probit estimates corresponding to these levels can be computed and their variance can be obtained for each of the S teams.

In our experience, parameter estimates converged very slowly or failed to converge with simulated data from models with similar $\tau's$ for all teams. As a practical matter, it seems appropriate to base convergence on the maximized value of the likelihood in exploratory (model selection) stages of the analysis. This criterion converges much more rapidly and may indicate that pursuing a model with different performance variances has little merit.

2.3.5 Results of Case III Analysis for Major League Baseball Data

Estimates for a model that includes separate performance means, a home field effect and separate performance variances are shown in Table 2.6. Notice, as an example, that the estimated probability that San Francisco would have defeated Florida while playing in San Francisco is $Pr(W_{SF} + 0.30 > W_{FL})$ where $W_{SF} \sim N(3.11, 2.88)$ and $W_{FL} \sim N(0, 47.70)$. The estimated probability is 0.68.

Table 2.6
Case III parameter estimates for National League teams from the 1993 season: β's and τ's are performance means and variances, respectively

Team	β	τ	*Team*	β	τ
Florida	0.00	47.70	Chicago	2.22	28.91
New York	0.08	21.27	St. Louis	2.44	26.18
San Diego	0.86	7.98	Montreal	2.45	1.54
Pittsburgh	1.45	14.85	Philadelphia	2.79	5.50
Colorado	1.59	0.05	Atlanta	2.94	1.00
Cincinnati	1.84	0.08	San Francisco	3.11	2.88
Houston	1.90	7.24			
Home Field Advantage:			0.30		

The likelihood ratio test statistic for equality of performance variances is 21.0842 on 13 degrees of freedom, for a p-value of 0.07. Thus, there is slight evidence that the Case V assumption of constant performance variance is not satisfied. The estimated performance distributions for the 14 teams are shown in Figures 2.3 and 2.4 for the constant variance and separate variance models, respectively.

2.4 Discussion

Batchelder and Burshad (1979) believed the following problems diminished the use of paired comparison designs:

- the assumption that ratings do not change over time,
- possibility of ties,
- unstable observations, i.e. unbalanced and scanty data,
- introduction of new items, and
- that most "simple" results involve complicated implicit equations that prevent expansion to more complicated advance work.

The latter three concerns can be addressed using the methods discussed in this article. Generalized linear models provide an established framework for which both the Bradley-Terry and Thurstone-Mosteller Model (Thurstone's Case V Model) are easily analyzed. Logistic and probit regression also facilitate the analysis of more complicated designs. For instance, both balanced and unbalanced designs can be analyzed as well as designs with scanty data, i.e. small $n_{ij}'s$, and both time independent and time dependent covariates can be analyzed.

The relatively simple application of the EM Algorithm allows maximum likelihood estimates to be collected for team performance variances, as postulated by Thurstone in his Case III Model. These estimates are more satisfying than the approximations proposed by Thurstone or Burros, both of whom restricted their methods to large, balanced designs which did not include covariates. Inference using likelihood analysis is easily conducted as well.

Definition of initial estimates, especially those for τ, are critical to rapid convergence of the EM Algorithm. Analysis on simulated data and the baseball data has indicated that the method proposed by Burros provides good values for $\tau^{(0)}$, reducing the number of iterations required for convergence by up to two-thirds when the former are used rather than setting $\tau_i^{(0)} = 1$ for all i. The performance of other possible initial estimates was not examined since satisfactory results were obtained with the estimates obtained from Burros.

The analysis presented within this article indicates that it is difficult to detect differing performance variability unless there is a substantial amount of replication of matches between teams, i.e. large n_{ij}, and in cases in which the variances are substantially different. In addition, in data examined for this paper the variance parameters tended to be greatly overestimated for small n_{ij}. Nevertheless, estimation of the Case III Model serves two important purposes. One, it provides a quantitative method of testing the assumption of equality of performance variances that is required in the Case V Model. Two, inclusion of team performance variability's increases the ability to differentiate between teams, i.e. gives a more accurate ranking of the r teams.

Although either the Bradley-Terry Model or Thurstone-Mosteller Model can be easily analyzed using logistic or probit regression, respectively, and in fact the Bradley-Terry Model is generally preferred due to its simple interpretation in terms of the odds ratio, analysis of performance variability's is greatly facilitated by the use of Thurstone's Case III Model. Estimation using the EM Algorithm requires the computation of $Q\left(\beta, \tau | \beta^{(r)}, \tau^{(r)}\right) = E_{W|Y}\left(l_c(\beta, \tau; Y, W) | Y; \beta^{(r)}, \tau^{(r)}\right)$. The expectations needed are much simpler if W is assumed to have a normal distribution (Thurstone) rather than a logistic distribution (Bradley-Terry).

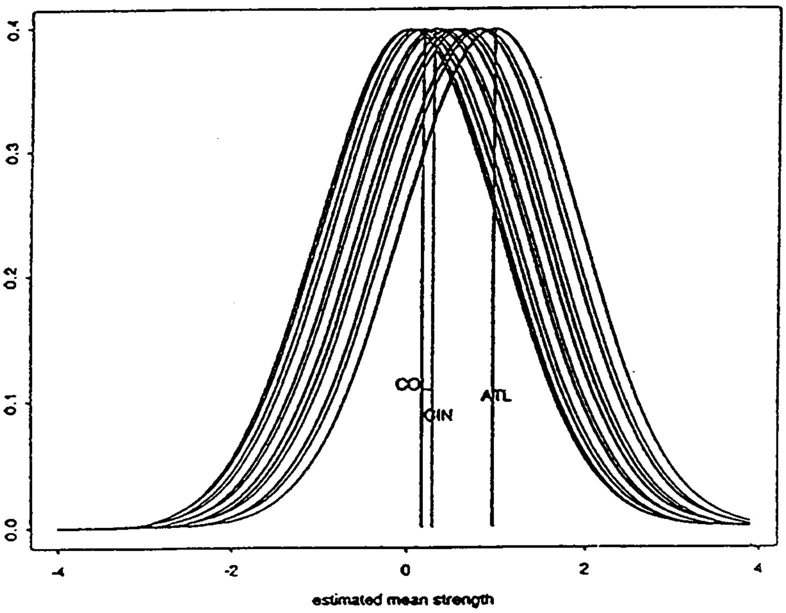

Figure 2.3
Performance Distributions for National League Teams for 1993 season under the Case V Model

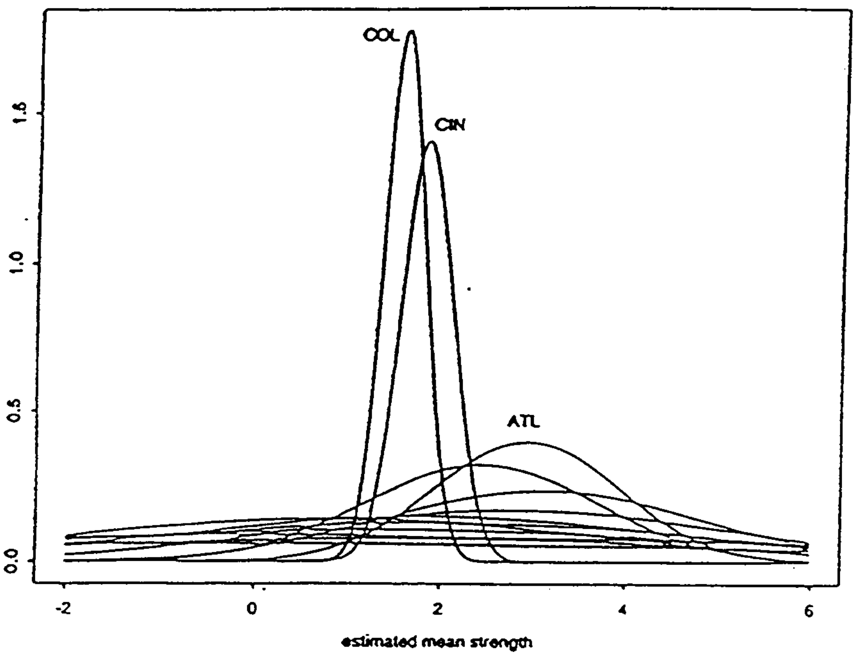

Figure 2.4
Performance Distributions for National League Teams for 1993 season under the Case III Model

APPENDIX

DERIVATION OF CONDITIONAL EXPECTATIONS

Let W_1 and W_2 be independent random variables with means μ_1 and μ_2, and variances σ_1^2 and σ_2^2. By equating W_1 and W_2 with W_i^H and W_i^V, respectively, the functional forms needed to compute equation (2.3.6) can be derived as described below. Specifically, the forms needed are: $E(W_1|W_1 > W_2)$, $E(W_1|W_1 < W_2)$, $E(W_1^2|W_1 > W_2)$, and $E(W_1^2|W_1 < W_2)$.

Using the definition of a conditional probability density function, the independence of W_1 and W_2, and the fundamental theorem of calculus, the conditional density of W_1 given that $W_1 > W_2$ can be simplified to

$$(1/K)\phi\left(\tfrac{w_1-\mu_1}{\sigma_1}\right)\Phi\left(\tfrac{w_1-\mu_2}{\sigma_2}\right) \text{ where } K = \Phi\left(\tfrac{\mu_1-\mu_2}{\sqrt{\sigma_1^2+\sigma_2^2}}\right). \tag{1}$$

Therefore,

$$E(W_1|W_1 > W_2) = \tfrac{1}{K}\int_{-\infty}^{\infty} w_1\phi\left(\tfrac{w_1-\mu_1}{\sigma_1}\right)\Phi\left(\tfrac{w_1-\mu_2}{\sigma_2}\right)dw_1. \tag{2}$$

By taking the derivative of $\phi\left(\frac{w_1-\mu_1}{\sigma_1}\right)$ one can show that

$$w_1\phi\left(\tfrac{w_1-\mu_1}{\sigma_1}\right) = \mu_1\phi\left(\tfrac{w_1-\mu_1}{\sigma_1}\right) - \sigma_1^2\left(\tfrac{d}{dw_1}\phi\left(\tfrac{w_1-\mu_1}{\sigma_1}\right)\right). \tag{3}$$

Substitution of (3) into (2) yields

$$E(W_1|W_1 > W_2) = \mu_1 - \tfrac{1}{K}\int_{-\infty}^{\infty}\sigma_1^2\left(\tfrac{d}{dw_1}\phi\left(\tfrac{w_1-\mu_1}{\sigma_1}\right)\right)\Phi\left(\tfrac{w_1-\mu_2}{\sigma_2}\right)dw_1. \tag{4}$$

This can be integrated by parts, letting $u = \sigma_1^2\Phi\left(\frac{w_1-\mu_2}{\sigma_2}\right)$ and $dv = \frac{d}{dw_1}\phi\left(\frac{w_1-\mu_1}{\sigma_1}\right)$. After some minor simplification (4) can be shown to equal

$$\mu_1 + (1/K)(\sigma_1/\sigma_2)\int_{-\infty}^{\infty} \tfrac{1}{2\pi} exp\left(\frac{-(w_1-\mu_1)^2}{2\sigma_1^2} - \frac{(w_1-\mu_2)^2}{2\sigma_2^2}\right) dw_1. \tag{5}$$

After some algebraic manipulation (5) can be expressed as the product of a constant, independent of w_1, and the probability density function of a normal distribution with mean

$$\frac{\sigma_2^2}{\sigma_1^2+\sigma_2^2}\mu_1 + \frac{\sigma_1^2}{\sigma_1^2+\sigma_2^2}\mu_2 \tag{6}$$

and variance

$$(\sigma_1^2\sigma_2^2)/(\sigma_1^2 + \sigma_2^2). \tag{7}$$

After integrating the normal density to unity, the remaining constant is

$$E(W_1|W_1 > W_2) = \mu_1 + \frac{\sigma_1^2}{\sqrt{\sigma_1^2+\sigma_2^2}} \frac{\phi\left(\frac{\mu_1-\mu_2}{\sqrt{\sigma_1^2+\sigma_2^2}}\right)}{\Phi\left(\frac{\mu_1-\mu_2}{\sqrt{\sigma_1^2+\sigma_2^2}}\right)}. \tag{8}$$

Calculation of $E(X_1|X_1 < X_2)$ is very similar. Applying the same arguments as above, the conditional density of W_1 given that $W_1 < W_2$ can be simplified to

$$[1/(1-K)]\phi\left(\frac{w_1-\mu_1}{\sigma_1}\right)\Phi\left(\frac{-(w_1-\mu_2)}{\sigma_2}\right) \tag{9}$$

so that (2) becomes

$$E(W_1|W_1 < W_2) = \frac{1}{1-K}\int_{-\infty}^{\infty} w_1\phi\left(\frac{w_1-\mu_1}{\sigma_1}\right)\Phi\left(\frac{-(w_1-\mu_2)}{\sigma_2}\right)dw_1 \tag{10}$$

The same steps used above can be used to calculate (10). The resulting calculus and algebra yield

$$E(W_1|W_1 < W_2) = \mu_1 - \frac{\sigma_1^2}{\sqrt{\sigma_1^2+\sigma_2^2}} \frac{\phi\left(\frac{\mu_1-\mu_2}{\sqrt{\sigma_1^2+\sigma_2^2}}\right)}{1-\Phi\left(\frac{\mu_1-\mu_2}{\sqrt{\sigma_1^2+\sigma_2^2}}\right)} \tag{11}$$

The $E(W_1^2|W_1 > W_2)$ and $E(W_1^2|W_1 < W_2)$ can be found as consequences of the above derivations. Note that for the former the following integral must be evaluated:

$$\frac{1}{K}\int_{-\infty}^{\infty} w_1^2 \phi\left(\frac{w_1-\mu_1}{\sigma_1}\right)\Phi\left(\frac{w_1-\mu_2}{\sigma_2}\right)dw_1 \tag{12}$$

This can be evaluated using the previous results by first multiplying (3) by w_1 and substituting the result into (12). This results in

$$\begin{aligned}&\frac{1}{K}\int_{-\infty}^{\infty} \mu_1 w_1 \phi\left(\frac{w_1-\mu_1}{\sigma_1}\right)\Phi\left(\frac{w_1-\mu_2}{\sigma_2}\right)dw_1 \\ &\qquad - \frac{1}{K}\int_{-\infty}^{\infty} \sigma_1^2 w_1 \left[\frac{d}{dw_1}\phi\left(\frac{w_1-\mu_1}{\sigma_1}\right)\right]\Phi\left(\frac{w_1-\mu_2}{\sigma_2}\right)dw_1.\end{aligned} \tag{13}$$

The first integral is $\mu_1 E(W_1|W_1 > W_2)$. Integration by parts is again used for the evaluation of the second integral, letting $u = w_1\Phi\left(\frac{w_1-\mu_2}{\sigma_2}\right)$ and $dv = \frac{d}{dw_1}\phi\left(\frac{w_1-\mu_1}{\sigma_1}\right)$. Of the resulting integrals, two are easily evaluated to 0 and $\sigma_1^2 K$, respectively. The third can be expressed as

$$(\sigma_1/\sigma_2)\int_{-\infty}^{\infty} w_1 \frac{1}{2\pi} exp\left(\frac{-(w_1-\mu_1)^2}{2\sigma_1^2} - \frac{(w_1-\mu_2)^2}{2\sigma_2^2}\right)dw_1. \tag{14}$$

This integral is very similar to the integral in (5). Thus, it is easily seen to be the mean of a normal random variable with the mean given in (6) and variance given in (7). Combining the above quantities, and performing some algebra, yields the desired expectation, $E(W_1^2|W_1 > W_2)$, as

$$\mu_1^2+\sigma_1^2+\frac{\sigma_1^2}{\sqrt{\sigma_1^2+\sigma_2^2}}\frac{\phi\left(\frac{\mu_1-\mu_2}{\sqrt{\sigma_1^2+\sigma_2^2}}\right)}{\Phi\left(\frac{\mu_1-\mu_2}{\sqrt{\sigma_1^2+\sigma_2^2}}\right)}\left(\mu_1+\left(\frac{\sigma_2^2}{\sigma_1^2+\sigma_2^2}\mu_1+\frac{\sigma_1^2}{\sigma_1^2+\sigma_2^2}\mu_2\right)\right) \tag{15}$$

The same steps are again followed to calculate the $E(W_1^2|W_1 < W_2)$, replacing (12) by

$$\frac{1}{1-K}\int_{-\infty}^{\infty} w_1^2\phi\left(\frac{w_1-\mu_1}{\sigma_1}\right)\Phi\left(\frac{-(w_1-\mu_2)}{\sigma_2}\right)dw_1. \tag{16}$$

It is easily seen that the $E(W_1^2|W_1 < W_2)$ is

$$\mu_1^2+\sigma_1^2-\frac{\sigma_1^2}{\sqrt{\sigma_1^2+\sigma_2^2}}\frac{\phi\left(\frac{\mu_1-\mu_2}{\sqrt{\sigma_1^2+\sigma_2^2}}\right)}{1-\Phi\left(\frac{\mu_1-\mu_2}{\sqrt{\sigma_1^2+\sigma_2^2}}\right)}\left(\mu_1+\left(\frac{\sigma_2^2}{\sigma_1^2+\sigma_2^2}\mu_1+\frac{\sigma_1^2}{\sigma_1^2+\sigma_2^2}\mu_2\right)\right). \tag{17}$$

References:

Atkinson, A. C. (1972). A test of the linear logistic and Bradley-Terry models. *Biometrika.* **59**, 37-42.

Batchelder, William H., and Bershad, Neil J. (1979). The statistical analysis of a Thurstonian model for rating chess players. *Journal of Mathematical Psychology.* **19**, 39-60.

Bradley, R. A. (1984). Paired comparisons: some basic procedures and examples. *Handbook of Statistics, Vol 4.* ed. P. R. Krishnaiah and P. K. Sen. Amsterdam: North-Holland, 299-326.

Bradley, R. A. and Terry, M. E. (1952). The rank analysis of incomplete block designs. I. The method of paired comparisons. *Biometrika.* **39**, 324-345.

Bradley, R. A. and El-Helbawy, A. T. (1976). Treatment contrasts in paired comparisons: Basic procedures with applications to factorials. *Biometrika,* **63**, 255-262.

Burros, R.H. (1951). The application of the method of paired comparisons to the study of reaction potential. *Psychology Review.* **58**, 60-66.

Burros, R. H. and Gibson, W. A. (1954). A solution for case III of the law of comparative judgement. *Psychometrika.* **19**, 57-64.

Critchlow, D. E. and Fligner, M. A. (1991) Paired comparison, triple comparison, and ranking experiments as generalized linear models, and their implementation on GLIM. *Psychometrika.* **56**, 517-533.

David, Herbert A. (1988). *The Method of Paired Comparisons, 2nd ed.* Charles Griffen and Company LTD, London.

Davidson, R. R. and Farquhar, P. H. (1976). A bibliography on the method of paired comparisons. *Biometrics.* **32**, 241-252.

Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977). Maximum likelihood from incomplete data via the EM Algorithm. *Journal of the Royal Statistical Society (B).* **39**, 1-22.

Edwards, A.W.F. (1972). *Likelihood: An account of the statistical concept of likelihood and its application to scientific inference.* Cambridge.

Elo, Arpad E. (1978). *The Rating of Chess Players: Past and Present.* Arco Publishing, Inc., New York.

Fienberg, S.E. (1979). Log linear representation for paired comparison models with ties within-pair order effects. *Biometrics.* **35**, 479-481.

Fienberg, S.E. and Larntz, K. (1976). Log linear representation for paired and multiple comparison models. *Biometrika.* **63**, 245-254.

Gibson, W. A. (1953). A least squares solution for case IV of the law of comparative judgment. *Psychometrika.* **18**, 15-21.

Lindsey, J.K. (1989). *The Analysis of Categorical Data Using GLIM (Lecture Notes in Statistics, Volume 56).* Springer-Verlag, New York.

Louis, Thomas, A. (1982). Finding the observed information matrix when using the EM Algorithm. *Journal of the Royal Statistical Society, Series B.* **44**, 226-233.

McCullagh, P. and Nelder, J.A. (1989). *Generalized Linear Models, 2nd ed.* Chapman & Hall, London.

Meng, X.L. and Rubin, D.B. (1993). Maximum likelihood estimation via the ECM algorithm: A general framework. *Biometrika.* **80**, 267-278.

Mosteller, F. (1951a). Remarks on the method of paired comparisons: I. The least squares solution assuming equal standard deviations and equal correlations. *Psychometrika.* **16**, 3-9.

Mosteller, F. (1951b). Remarks on the method of paired comparisons: II. The effect of an aberrant standard deviation when equal standard deviations and equal correlations are assumed. *Pyschometrika.* **16**, 203-206.

Mosteller, F. (1951c). Remarks on the method of paired comparisons: III. A test of significance for paired comparisons when equal standard deviations and equal correlations are assumed. *Psychometrika.* **16**, 207-218.

Sinclair, C.D. (1982). GLIM for preference. *GLIM 82: Proceedings of the international conference on generalized linear models: Lecture notes in Statistics, Volume 14.* ed. R. Gilchrist. Springer-Verlag, New York, 164-178.

The Sporting News: Baseball Guide, 1994 Edition. ed. Craig Carter and Dave Sloan. The Sporting New Publishing Company, St. Louis.

Thurstone, L.L. (1927). Psychophysical analysis. *American Journal of Psychology.* **38**, 368-389.

Thurstone, L.L. (1932). The method of paired comparisons for social values. *Jouranal of Abnormal and Social Psychology.* **21**, 384-400.

Chapter 3

Paired Comparisons Models With Serial Correlation

James D. Kolsky and Daniel W. Schafer

Submitted to *Biometrics*,
International Biometric Society, Washington D.C.,
May 1996.

3. Paired Comparisons Models With Serial Correlation

Abstract

The Thurstone-Mosteller Model is commonly used to rank competitors or items from paired comparisons experiments in which one item from each pair wins or is preferred over the other. This model is sometimes motivated through an underlying, normally-distributed performance distribution for each item or competitor, with a competitor winning a trial if a draw from its performance distribution exceeds that from its competitor's. The observed outcome is solely the win or loss for each team, but regression models may be specified for the performance means, including time-dependent and time-independent explanatory variables. This article presents an extension of the model to permit the "performance" or "worth" variables of each competitor to be serially correlated. As an example, a basketball team's performance in its current game is allowed to be correlated with its performance in the previous game. The extension in this paper comes from supposing the error structure for the performance distribution for each team is normal with first-order autocorrelation. The EM Algorithm is used, treating the underlying draws from the performance distributions as "missing data." This provides approximate maximum likelihood estimates; the approximation is due to the use of Monte Carlo integration in the E-step of the algorithm. Unfortunately, the heavy computational requirement and the inability to provide either the maximized likelihood function or the information matrix, make the approach unattractive for practical use. Two approximations are presented, however, which can be carried out with standard probit regression routines and some minor programming.

Keywords: auto-regressive model, Bradley-Terry Model, EM Algorithm, MCEM Algorithm, probit regression, serial correlation, Thurstone-Mosteller Model.

3.1 Introduction

Paired comparisons describes a data structure in which r items or individuals are compared pairwise. Attention here is to the case where a single binary response is available for each pair, indicating which item was preferred. An extensive body of literature exists regarding the design and analysis of paired comparison experiments when comparisons are made independently. Important summaries are the book by David (1988) and the article by Bradley (1984). Extensive bibliographies on the subject are provided by David and also by Davidson and Farquar (1976).

The paired comparisons problem is prevalent in many fields. The examples in this paper have to do with sports competition, and to convey the new ideas in a convenient manner, sports terminology will be used. For instance, data from the 1993 Major League Baseball season are used in one example. There are 14 teams which compete with each other. Based on the win-loss outcomes of all games, paired comparisons analysis may be used to rank the teams after accounting for covariates or to model team performance as a function of explanatory variables. Explanatory variables may be time-independent, like the average age of the team's players at the beginning of the season, or time-dependent, such as whether a team played on its home field. See Kolsky and Schafer (1996) for further details about modeling the Thurstone-Mosteller Model and Bradley-Terry Model using probit and logistic regression. It should be noted that extensive consideration of paired comparisons analysis has been given to the ranking of chess players (see, for example, Joe (1990), Henery (1992), and Batchelder and Bershad (1979)). This paper has some relevance to that problem, but no allowance is made here for the possibility of draws (or ties).

3.1.1 Model

Thurstone's models (1927) were motivated by the consideration of an underlying, continuous "performance" variable for each item. Consider, for example, modeling the outcome of a basketball game between teams s and t through the selection of random performance variables W_s and W_t drawn from continuous "performance distributions" for each team. This performance can be conceptualized as an abstraction which cannot be quantitatively measured, although it could be thought of as the sum total of all decisions and actions that will occur during the course of the game (Elo, 1978). Team s wins if $W_s > W_t$ and loses otherwise. The Thurstone Model is a consequence of assuming that W_s is normally distributed with mean α_s and variance σ_s, for $s = 1, ...r$, where r is the number of teams, or items, being compared.

Thurstone originally proposed several versions of the model, each requiring different assumptions to be made about the variances and covariance's of the performance variables. The Case V Model, commonly referred to as the Thurstone-Mosteller Model, assumes that $\sigma_s = \sigma$ for all s (and since σ is not identifiable it is commonly taken to be equal to 1), and that each comparison is made independently of all other comparisons. Although the performance of a team was assumed to be independent of its opponent's performance, Mosteller (1951a) later showed that the assumption could be relaxed; the covariance of performances within a trial can be non-zero, but they must be constant.

It is reasonable to believe that some comparisons, especially those made on successive trials, are not independent. For instance, in athletic competitions a team's performance may show some serial correlation. Similarly, in taste preference studies a persons palette may show streaks of liking and disliking certain tastes. There are three reasons for the consideration of a model that quantifies the lack of independence between team performance variables in successive games: (1) to check the assumption of independence typically assumed in the usual Thurstone Case V Model , (2) to draw inference about the serial correlation coefficient, and (3) to draw inference about the regression coefficients after accounting for the serial correlation.

Let Y_i be a binary response taking the value 1 if the "home team" won and 0 if the visiting team won game i, for $i = 1, ..., n$. There need not be "home" and "visiting" items;

this designation may simply reflect the order in which the items in a pair are listed. Consider latent random variables W_i^H and W_i^V, representing the "performance" of the home and visiting teams in game i, and suppose that

$$Y_i = \begin{cases} 1 \text{ if } W_i^H > W_i^V \\ 0 \text{ otherwise} \end{cases} \quad . \tag{3.1.1}$$

The notation that follows is intended to allow the $W's$ to have normal linear regressions on explanatory variables, with separate lag-1 auto-regression error structures for each team or item.

It is first necessary to define some terminology that indicates which team was the home team and which team was the visiting team in game i. Let T_i^H be an $(r-1)$-by-1 vector with the j^{th} element equal to 1 if team $(j+1)$ was the home team in game i, and 0 otherwise, for $j = 1, ..., r-1$, where r is the number of distinct teams. Similarly, let T_i^V be the $(r-1)$-by-1 vector with the j^{th} element equal to 1 if team $(j+1)$ was the visiting team, and 0 otherwise. Next let X_i^H and X_i^V be the p-by-1 vectors of explanatory variables associated with the home and visiting teams in game i. It will typically be the case that $X_i^H = (T_i^H : U_i^H)$, so that the home team's performance is a function of which particular team is the home team and additional covariates associated with that team in the i^{th} game, represented by U_i^H. It need not be the case, however, that T_i^H is contained in X_i^H. X_i^V is defined in an analogous manner.

It is assumed here that

$$\begin{aligned} W_i^H &= X_i^{H\prime}\beta + \epsilon_i(T_i^H) \\ &\quad \epsilon_i(T_i^H) = \rho\epsilon_{prev}(T_i^H) + \delta_i^H \end{aligned} \tag{3.1.2}$$

$$\begin{aligned} W_i^V &= X_i^{V\prime}\beta + \epsilon_i(T_i^V) \\ &\quad \epsilon_i(T_i^V) = \rho\epsilon_{prev}(T_i^V) + \delta_i^V \end{aligned} \tag{3.1.3}$$

where

$$\delta_i^H \sim N(0,1) \quad \text{and} \quad \delta_i^V \sim N(0,1) \tag{3.1.4}$$

It is also assumed that δ_i^H and δ_i^V are independent of one another. The notation $\epsilon_{prev}(T_i^H)$ identifies the error term in the performance regression for the *previous* game of the home team (regardless of whether that team was the home or visiting team in its previous game). The point is that each of the r teams has a performance distribution, which has a regression on explanatory variables (which may be either time-independent or time-dependent) with an AR(1) structure. This model implies that the performance variables for the home and visiting teams corresponding to game i are independent of one another with

$$W_i^H \sim N(\beta' X_i^H, \tau) \tag{3.1.5}$$

and

$$W_i^V \sim N(\beta' X_i^V, \tau). \tag{3.1.6}$$

where τ is $1/(1-\rho^2)$. If $\rho = 0$ then this reduces to the usual Thurstone-Mosteller Model,

$$W_i^H \sim N(\beta' X_i^H, 1) \tag{3.1.7}$$

and

$$W_i^V \sim N(\beta' X_i^V, 1), \tag{3.1.8}$$

independently for all games.

There are, apparently, no previous attempts to incorporate serial correlation into Thurstone's Case V Model. There are some related works, however, for chess data. Batchelder and Bershad (1979), Joe and White (1992) and Glickman (1993) estimate parameters quantifying the effect of competing in different eras or in different tournaments. However, individual matches or comparisons within eras or tournaments are still considered to be independent.

3.1.2 Scope of Research

Section 3.2 discusses maximum likelihood estimation of β and ρ using the MCEM Algorithm (Wei and Tanner, 1990), which is the EM Algorithm (Dempster, Laird, and Rubin, 1977) using Monte Carlo evaluation in the E-step. This provides maximum likelihood estimates, but likelihood inference is hampered by the inability to cleanly and quickly calculate the maximized likelihood or the observed information matrix. These obstacles lead to the consideration of two other estimation techniques, which were motivated by analogies to dynamic generalized linear models for binary responses (Cox, 1970, and Zeger and Qaqish, 1988). These methods, described in Sections 3.3 and 3.4, conveniently permit estimation and inference using standard routines for probit and logistic regression. Studies of their operating characteristics, reported in Section 3.5, suggest their usefulness for testing the presence of serial correlation in team performance, and for usual estimation and inference in paired comparisons models, after accounting for such serial correlation, as long as the serial correlation coefficient is not too large. The models are used to analyze data from the 1993 Major League Baseball season in Section 3.6.

3.2 Maximum Likelihood Estimation Using the EM Algorithm

By treating the latent performance variables as missing data, the EM Algorithm can be used to obtain maximum likelihood estimates for both β and ρ. Let W be the $2nx1$ vector $(W^{H'} : W^{V'})'$ and X the $2nxp$ matrix $(X^{H'} : X^{V'})'$. Thus, the home teams' explanatory variables are contained in the first n rows of X and the visiting teams' explanatory variables in the last n rows. For convenience, sort W and X so that all performance variables and explanatory variables for each of the r teams are grouped first by team $(1, ..., r)$ and then sequentially for each team. There will be a block of n_s games for team s ($s = 1, ..., r$).

The "complete data" is the combination of both Y and W while the "observed data" is the vector Y. The EM Algorithm revolves around the function

$$Q(\beta, \rho | \beta^{(t)}, \rho^{(t)}) = E\Big[l_c(\beta, \rho; Y, W) | Y; \beta^{(t)}, \rho^{(t)}\Big], \tag{3.2.1}$$

where $l_c(\beta,\rho;Y,W)$ is the log-likelihood based on the complete data, and $\beta^{(t)}$ and $\rho^{(t)}$ are parameter estimates after the t^{th} iteration. (The β or ρ involved in the conditional expectations are replaced by their current parameter estimates, $\beta^{(t)}$ and $\rho^{(t)}$.) The updated estimates, $\beta^{(t+1)}$and $\rho^{(t+1)}$, are calculated as the β and ρ that maximize (3.2.1). These two steps are repeated until the estimates converge.

Define W_{sk} as the performance of team s in its k^{th} game, and let X_{sk} represent the corresponding explanatory variable vector. Since the conditional distribution of Y given W does not depend on any of the unknown parameters, and since performances for different teams are independent of one another, the complete data log-likelihood may be written as

$$l_c(\beta, \rho; Y, W) = \sum_{s=1}^{r} log\, f(W_{s1}, W_{s2}, ..., W_{sn_s}; \beta, \rho) \tag{3.2.2}$$

which, because of the AR(1) structure, may be conveniently reexpressed as

$$\sum_{s=1}^{r} log\, f(W_{s1}; \beta, \rho) + log\, f(W_{s2} | W_{s1}; \beta, \rho) + ... + log\, f(W_{sn_s} | W_{s(n_s-1)}; \beta, \rho). \tag{3.2.3}$$

The model specified in Section 3.1.1 implies that the conditional distribution of W_{sk} given $W_{s(k-1)}$ is

$$N(\beta' X_{sk} + \rho(W_{s(k-1)} - \beta' X_{s(k-1)}), 1), \tag{3.2.4}$$

where X_{s0} and W_{s0} are defined to be zero. The complete data log-likelihood is therefore,

$$l_c(\beta, \rho; Y, W) = -\tfrac{1}{2}\sum_{s=1}^{r}\sum_{k=1}^{n_s}\Big[W_{sk} - \rho W_{s(k-1)} - \beta'\left(X_{sk} - \rho X_{s(k-1)}\right)\Big]^2. \tag{3.2.5}$$

It follows that the expected value of the complete data log-likelihood given the observed data may be written as

$$\begin{aligned} Q\left(\beta, \rho | \beta^{(r)}, \rho^{(r)}\right) = -\tfrac{1}{2}\Big\{\sum_{s=1}^{r}\sum_{k=1}^{n_s} V_{sk}^{(t)} - 2\rho C_{sk,s(k-1)}^{(t)} + \rho^2 V_{s(k-1)}^{(t)} + \\ \Big[E_{sk}^{(t)} - \rho E_{s(k-1)}^{(t)} - \beta'(X_{sk} - \rho X_{s(k-1)})\Big]^2\Big\} \end{aligned} \tag{3.2.6}$$

where

$$E_{sk}^{(t)} = E(W_{sk}|Y; \beta^{(t)}, \rho^{(t)}), \tag{3.2.7}$$

$$V_{sk}^{(t)} = Var(W_{sk}|Y; \beta^{(t)}, \rho^{(t)}), \tag{3.2.8}$$

and

$$C_{sk,s(k-1)}^{(t)} = Cov(W_{sk}, W_{s(k-1)}|Y; \beta^{(t)}, \rho^{(t)}). \tag{3.2.9}$$

Ignoring, for the moment, how these expectations are found, the M-step of the algorithm updates the parameter estimates as those values of β and ρ that maximize $Q(\beta, \rho|\beta^{(t)}, \rho^{(t)})$. Using the results of Meng and Rubin (1993), this can be accomplished in a two-step process by replacing the M-step with two CM (Conditional Maximization) steps. $Q(\beta, \rho|\beta^{(t)}, \rho^{(t)})$ is first maximized with respect to β while ρ is held fixed at its currently estimated value, followed by maximization with respect to ρ with β held fixed at its newly estimated value:

$$\beta^{(t+1)} = \Big(\sum_{s=1}^{r}\sum_{k=1}^{n_s} X_{sk}^{*(t)} X_{sk}^{*(t)\prime}\Big)^{-1}\Big(\sum_{s=1}^{r}\sum_{k=1}^{n_s} X_{sk}^{*(t)} Z_{sk}^{(t)}\Big), \tag{3.2.10}$$

where

$$X^{*(t)}_{sk} = X_{sk} - \rho^{(t)} X_{s(k-1)} \tag{3.2.11}$$

and

$$Z^{(t)}_{sk} = E^{(t)}_{sk} - \rho^{(t)} E^{(t)}_{s(k-1)}; \tag{3.2.12}$$

and

$$\rho^{(t+1)} = \frac{\sum_{s=1}^{r}\sum_{k=1}^{n_s} C^{(t)}_{sk,s(k-1)} + (E^{(t)}_{sk} - \beta^{(t+1)\prime} X_{sk})(E^{(t)}_{s(k-1)} - \beta^{(t+1)\prime} X_{s(k-1)})}{\sum_{s=1}^{r}\sum_{k=1}^{n_s-1} V^{(t)}_{sk} + (E^{(t)}_{sk} - \beta^{(t+1)\prime} X_{sk})^2}. \tag{3.2.13}$$

These calculations parallel the ordinary least squares computation of regression coefficients and the serial correlation coefficient in an AR(1) model, after filtering.

Because it is not feasible to develop closed form solutions for all of the expectations in 3.2.7-3.2.9, Monte Carlo evaluation is suggested here. This is the MCEM (Monte Carlo EM) approach (Wei and Tanner, 1990). An entire season of performance variables, W^m_{sk}, are simulated according to the AR(1) regression model using the current parameter estimates, for $m = 1, ..., M$ Monte Carlo "seasons." $E^{(t)}_{sk}$ is computed as

$$\frac{1}{M_S}\sum_{S} W^m_{sk}, \tag{3.2.14}$$

where S is the set of simulations for which $Y^m_{s(k-1)} = Y_{s(k-1)}, Y^m_{sk} = Y_{sk}$, and $Y^m_{s(k+1)} = Y_{s(k+1)}$, and M_S is the number of simulations for which this match occurs. That is, $E^{(t)}_{sk}$ is the average value of W^m_{sk} over all simulated "seasons" for which the simulated outcomes of games $s(k-1)$, sk, and $s(k+1)$ match the observed outcomes. $V^{(t)}_{sk}$ is computed as

$$\frac{1}{M_S}\sum_{S}(W^m_{sk})^2 - (E^{(t)}_{sk})^2, \tag{3.2.15}$$

and $C^{(t)}_{sk,s(k-1)}$ is computed as

$$\frac{1}{M_T}\sum_{T} W^m_{sk} W^m_{s(k-1)} - (E^{(t)}_{sk})^2, \tag{3.2.16}$$

where T is the set of simulated seasons for which $Y^m_{s(k-2)} = Y_{s(k-2)}$, $Y^m_{s(k-1)} = Y_{s(k-1)}$, $Y^m_{sk} = Y_{sk}$, and $Y^m_{s(k+1)} = Y_{s(k+1)}$; and M_T is the number of simulated seasons in this set.

3.2.1 Convergence and Initial Estimates

Good initial estimates for β are obtained by fitting the probit regression models mentioned above (under the assumption that $\rho = 0$), where the $Y_i's$ have a probit regression on $X^H_i - X^V_i$ (see Kolsky and Schafer, 1996, for more details). This supplies estimates of β^* in the model

$$Pr(Y_i = 1) = \Phi\left[\beta^{*\prime}(X^H_i - X^V_i)\right]. \tag{3.2.17}$$

Since the model, specified in terms of the underlying performance variables, implies, for $\rho = 0$, that

$$Pr(Y_i = 1) = \Phi\left[\beta'(X^H_i - X^V_i)/\sqrt{2}\right], \tag{3.2.18}$$

the probit regression coefficient should be multiplied by $\sqrt{2}$ when used as initial values for the EM Algorithm. We have used the estimate of ρ from the approximate method described in Section 3.3 as a starting value for the EM Algorithm, but have also obtained adequate results with more ad hoc choices, such as $\rho = 0.5$.

The computational burdens of the MCEM approach to this problem are obvious. After some experimentation, we have used the choice $M = 800$ Monte Carlo simulations in each E-step. Increasing this to 1200 showed no improvement in convergence time for the situations studied. Wei and Tanner (1990) recommended increasing M with higher iterations, but that strategy has not been pursued here. The algorithm converged in 10 to 75 hours, depending on the value of ρ (using an S+ function on a SPARC 20 computer).

Such slow convergence might be tolerable if full likelihood analysis were provided. Unfortunately, the added burden to approximate the maximized likelihood or the observed information matrix (using the SEM Algorithm of Meng and Rubin, 1991) proved to be prohibitive. Improved variations of the EM Algorithm and alternative programming languages may be fruitful. For practical analysis of paired comparisons we believe that some approximations motivated by this approach are more appealing.

3.3 Probit Regression with a Covariate to Account for Serial Correlation

The method proposed here uses estimated expected values of performances and "residuals" based on these to incorporate the effect of the serial correlation through a single additional covariate in a probit regression model. First, it was shown in Kolsky and Schafer (1996) that if W_i^H and W_i^V are independent normally-distributed random variables with means $\beta' X_i^H$ and $\beta' X_i^V$ and variances of 1, then

$$E(W_i^H | W_i^H > W_i^V) = \beta' X_i^H + \frac{1}{\sqrt{2}} \frac{\phi(a)}{\Phi(a)} \tag{3.3.1}$$

and

$$E(W_i^H | W_i^H < W_i^V) = \beta' X_i^H - \frac{1}{\sqrt{2}} \frac{\phi(a)}{1-\Phi(a)} \tag{3.3.2}$$

where $a = [\beta'(X_i^H - X_i^V)]/\sqrt{2}$. Similar expressions hold, of course, for the expectation of W_i^V conditional on whether the home team won or lost game i.

Based on the model in (3.1.1) and (3.1.2), it is evident that

$$W_i^H - W_i^V = \beta'(X_i^H - X_i^V) + \rho\left[\epsilon_{prev}(T_i^H) - \epsilon_{prev}(T_i^V)\right] + \delta_i^H + \delta_i^V . \tag{3.3.3}$$

Using

$$\epsilon_{prev}(T_i^H) = W_{prev}(T_i^H) - \beta'\Big[X_{prev}(T_i^H)\Big] \tag{3.3.4}$$

and

$$\epsilon_{prev}(T_i^V) = W_{prev}(T_i^V) - \beta'\Big[X_{prev}(T_i^V)\Big], \tag{3.3.5}$$

where $W_{prev}(T_i^H)$ and $X_{prev}(T_i^H)$ are the performance and the explanatory variable vector for the home team in its previous game, and $W_{prev}(T_i^V)$ and $X_{prev}(T_i^V)$ are similar expressions for the visiting team, it follows that

$$W_i^H - W_i^V = \beta'(X_i^H - X_i^V) + \rho U_i + \delta_i^H + \delta_i^V, \tag{3.3.6}$$

where

$$U_i = W_{prev}(T_i^H) - \beta'\Big[X_{prev}(T_i^H)\Big] - W_{prev}(T_i^V) + \beta'\Big[X_{prev}(T_i^V)\Big]. \tag{3.3.7}$$

Therefore, since $\delta_i^H + \delta_i^V$ has a $N(0,2)$ distribution, an approximate probit regression model is obtained by taking U_i to be known:

$$\Phi^{-1}\Big[Pr(Y_i = 1)\Big] = \Big[\beta'(X_i^H - X_i^V) + \rho U_i\Big]/\sqrt{2}. \tag{3.3.8}$$

This suggests the following approach: (1) Obtain an initial estimate of β by fitting the probit regression of Y on $(X_i^H - X_i^V)$ and multiplying the estimated coefficient of $(X_i^H - X_i^V)$ by $\sqrt{2}$. (2) Compute the estimated expected performances from (3.3.1) and (3.3.2). (3) Compute U_i with the $W's$ replaced by their estimated expected values. (4) Fit the probit regression of Y on $(X_i^H - X_i^V)$ and U_i. The coefficient of the former, multiplied by $\sqrt{2}$ is the estimate of β, and the coefficient of U_i multiplied by $\sqrt{2}$ is the estimated serial correlation of performances.

This approach is substantially simpler than the MCEM of the previous section. It only involves two probit regression fits, but there is some bookkeeping effort involved in attaching the correct "previous game" to the home and visiting team for each game. Some simulation studies are shown in Section 3.5, which indicate (1) ρ is estimated quite accurately for values of $\rho < 0.9$, (2) less bias is introduced into estimates of the strength parameters then when using models that ignore ρ, and (3) certain covariates, such as home field advantage are estimated fairly well for values of $\rho < 0.9$.

3.4 Probit Regression with a Covariate Based on Previous Outcomes

A simple method that has proved useful in accounting for serial correlation in binary regression models uses ordinary logistic and probit models but with previous responses included as covariates. For a binary sequence, $Y_1, ..., Y_n$, for example, one model is

$$logit(\pi_i) = \alpha_i + \beta Y_{i-1} \tag{3.4.1}$$

(Zeger and Qaqish, 1988; Anderson, 1954; Cox, 1970, 1981). The methods are primarily for "observation-driven" models in which only the observed outcomes are correlated, i.e. there is no unobserved, underlying process contributing to the correlation. These models can then be fit with ordinary routines for generalized linear models. West, Harrison, and Mignon (1985) discuss a generalization of the Kalman filter for an arbitrary link function. Such a procedure addresses a number of weaknesses inherent in the usual generalized linear model, such as the fixed relationship of the parameters across observations, the adequacy of the generalized linear model asymptotic theory, and the failure to account for the sequential procession of observations. Nevertheless, the simplicity of using the previous response as a covariate make the former an attractive alternative.

For the paired comparisons models discussed here, an analogous approach is to include $Y_{prev}(T_i^H) - Y_{prev}(T_i^V)$ as a covariate, where $Y_{prev}(T_i^H)$ is the previous outcome for the home team, regardless of whether that team was the home or visiting in its previous game and $Y_{prev}(T_i^V)$ is defined analogously for the visiting team. T_i^H and T_i^V are defined as in Section 3.1.1. In this form, however, the covariate excludes information about other explanatory variables, which are quite important in the examples considered here. To incorporate such information we include, as a covariate, the quantity

$$V_i = \left[Y_{prev}(T_i^H) - \widehat{\pi}_{prev}(T_i^H)\right] - \left[Y_{prev}(T_i^V) - \widehat{\pi}_{prev}(T_i^V)\right] \qquad (3.4.2)$$

where $\widehat{\pi}_{prev}(T_i^H)$ is the predicted probability, for the home team, of winning the previous game according to the usual Thurstone-Mosteller Model, treating ρ as zero and $\widehat{\pi}_{prev}(T_i^V)$ is analogously defined for the visiting team. The probability that the home team wins, $P(Y_i = 1)$, can then be modeled as

$$\Phi^{-1}(\pi_i) = \beta'(X_i^H - X_i^V) + \theta V_i \qquad (3.4.3)$$

where V_i is the covariate described above, and π_i, β, X_i^H, and X_i^V are defined as in Section 3.2.

The covariate V_i could also be used in logistic regression models to extend the Bradley-Terry Model to account for serial correlation. While logistic regression has a more convenient interpretation, work here will focus on the use of probit regression so that the model can easily be compared to the other models discussed in this paper. Unlike the method of Section 3.3, no estimate of serial correlation is provided here. This approach, however, is the simplest one for drawing inferences about the $\beta's$ in the presence of serial correlation. Furthermore, a test of significance of θ provides an indirect assessment for serial correlation of the performance variables.

3.5 Simulation Studies

Some simulation studies were conducted to investigate the operating characteristics of the methods in Sections 3.3 and 3.4. Because of the overwhelming computational time required for the MCEM Algorithm, they were investigated only for a few simulated conditions. Performance variables were simulated for 14 teams with known strength parameters, α, according to an AR(1) model with a known ρ. A common home field advantage was included into the structure of the simulated performance variables as well. Each team participated in 162 games, with no more than one day off between games, playing each other either 12 or 13 times. The "observed" data, Y, was formed by defining $Y_i = 1$ if $W_i^H > W_i^V$ where W_i^H and W_i^V are the simulated performance values for game i. The regression model is indicated by:

$$W_i^H \sim N(\beta + \sum_{j=2}^{r} \alpha_j T_{ij}^H, \tfrac{1}{1-\rho^2}) \tag{3.5.1}$$

and

$$W_i^V \sim N(\sum_{j=2}^{r} \alpha_j T_{ij}^V, \tfrac{1}{1-\rho^2}) \tag{3.5.2}$$

where β represents the home field advantage and T_{ij}^H and T_{ij}^V are defined as in Section 3.1.1.

Such samples were generated 200 times for each of several values of ρ. Table 3.1 below summarizes some characteristics of the Thurstone-Mosteller (TM) method (ignoring ρ); the method of Section 3.3 which uses the covariate U; and the method of Section 3.4 which uses the covariate V. Average values (over the 200 simulated samples) of the estimates of the home field effect are listed as $\widehat{\beta}_{TM}$, $\widehat{\beta}_U$, and $\widehat{\beta}_V$, respectively. The rows labeled SE contain average values of the standard errors associated with the estimates of home field effect, and the rows labeled SD show the sample standard deviations of the estimates over the 200 values. The accuracy of the standard errors of the estimates can be investigated by the comparison of these two rows. Average values (over the 200 simulated samples) of the estimates for the additional covariates described in Sections 3.3 and 3.4 are listed as $\widehat{\rho}_U$ and $\widehat{\theta}_V$, respectively. The rows labeled SE and SD are defined as above.

Table 3.1: Comparison of 3 Estimates of β (True Value=0.2121)
14 Teams, each playing 162 games

	ρ=0.1	*0.3*	*0.5*	*0.7*	*0.9*
$\widehat{\beta}_{TM}$	0.2147	0.2018	0.1910	0.1593	0.0988
$SE(\widehat{\beta}_{TM})$	0.0404	0.0402	0.0398	0.0392	0.0390
$SD(\widehat{\beta}_{TM})$	0.0405	0.0439	0.0468	0.0554	0.0510
$\widehat{\beta}_U$	0.2152	0.2048	0.2006	0.1772	0.1220
$SE(\widehat{\beta}_U)$	0.0404	0.0405	0.0409	0.0417	0.0451
$SD(\widehat{\beta}_U)$	0.0406	0.0446	0.0478	0.0600	0.0620
$\widehat{\beta}_V$	0.2152	0.2045	0.2000	0.1763	0.1210
$SE(\widehat{\beta}_V)$	0.0404	0.0405	0.0409	0.0417	0.0451
$SD(\widehat{\beta}_V)$	0.0406	0.0445	0.0477	0.0597	0.0616
$\widehat{\rho}_U$	0.0527	0.2027	0.3606	0.5521	0.8576
$SE(\widehat{\rho}_U)$	0.0413	0.0406	0.0405	0.0412	0.0454
$SD(\widehat{\rho}_U)$	0.0383	0.0408	0.0431	0.0439	0.0563
$\widehat{\theta}_V$	0.0616	0.2378	0.4224	0.6432	0.9966
$SE(\widehat{\theta}_V)$	0.0485	0.0477	0.0474	0.0480	0.0527
$SD(\widehat{\theta}_V)$	0.0449	0.0479	0.0503	0.0508	0.0642

Evidence from this study, in which the home field advantage, β, is estimated as well as 13 strength parameters, indicates that if attention is restricted to the covariate there is little reason to use the approximation methods, since the simpler Thurstone-Mosteller methods provide, essentially, the same results. There is some bias in the usual Thurstone-Mosteller estimates due to ignoring ρ and the two approximation methods that account for ρ offer some modest bias reduction. Note that only for $\rho = 0.1$ are the mean standard errors and standard deviations of the covariate approximately equal. For higher values of ρ the standard deviation tends to be at least 20% higher than the mean standard error. Surprisingly, the approximation models seem to be no better in this regard than the usual Thurstone-Mosteller Model. If there is interest in quantifying the amount of correlation, the approximation model using the covariate U_i does an excellent job of estimating the correlation and for both models the mean standard errors of both $\widehat{\rho}$ and $\widehat{\theta}$ are approximately the same as the standard deviations of $\widehat{\rho}$ and $\widehat{\theta}$, except for the most extreme levels of correlation. Tables 3.2 and 3.4 below indicate that while the covariate is not strongly affected by the model used, there are differences between the estimated strength parameters of the three methods.

Table 3.2 summarizes the results from the three methods for one of the strength parameters, α, defined by (3.5.1) and (3.5.2). Average values (over the 200 simulated samples) of the estimates of the strength parameter are listed as $\widehat{\alpha}_{TM}$, $\widehat{\alpha}_U$, and $\widehat{\alpha}_V$, respectively. Again, the rows labeled SE contain average values of the standard errors associated with the estimates of the strength parameter, and the rows labeled SD show the sample standard deviations of the estimates over the 200 values.

Unlike in Table 3.1, none of the methods do a very good job of estimating the standard error of the strength parameter for $\rho > 0.1$. In addition, there is a great deal more bias in these estimates. However, the two approximation methods are an improvement over the usual Thurstone-Mosteller Model in that they offer some bias reduction and maintain constant values of the mean standard error over all values of ρ, whereas the mean standard error for the usual Thurstone-Mosteller Model is decreasing as ρ increases.

Table 3.2: Comparison of 3 Estimates of α (True Value=1.1314)
14 Teams, each playing 162 games

	ρ=0.1	0.3	0.5	0.7	0.9
$\widehat{\alpha}_{TM}$	1.1315	1.0830	0.9871	0.8517	0.5546
$SE(\widehat{\alpha}_{TM})$	0.1508	0.1497	0.1475	0.1447	0.1420
$SD(\widehat{\alpha}_{TM})$	0.1506	0.1795	0.1849	0.2524	0.4024
$\widehat{\alpha}_U$	1.1335	1.1000	1.0361	0.9455	0.6830
$SE(\widehat{\alpha}_U)$	0.1510	0.1510	0.1516	0.1541	0.1650
$SD(\widehat{\alpha}_U)$	0.1512	0.1822	0.1930	0.2722	0.4838
$\widehat{\alpha}_V$	1.1334	1.0988	1.0334	0.9408	0.6774
$SE(\widehat{\alpha}_V)$	0.1510	0.1510	0.1515	0.1539	0.1647
$SD(\widehat{\alpha}_V)$	0.1511	0.1819	0.1923	0.2702	0.4791

A similar study was conducted in which twice as many teams were examined ($r = 27$), but only half as many games were played by each team ($n_i = 82$ for all i). The data was simulated as described above. Results from the simulations are shown in Table 3.3. Estimates in the table are the same as those defined for Table 3.1 above.

The results from this simulation are very similar to those observed for the data contained in Table 3.1, with the exception that the mean standard errors and standard deviations for the home field effects are approximately equal except for the highest levels of correlation.

Due to the amount of time required to calculate approximate maximum likelihood estimates for the parameters from the simulated data described above using the MCEM Algorithm, it was not feasible to include them in any of the above simulation studies. Table 3.4, however, compares all three methods for a few selected values of ρ on a single generated sample. Estimates fit using the usual Thurstone-Mosteller Model, under the assumption that $\rho = 0$, are included for completeness. Data were simulated as described above for the study summarized in Table 3.1.

Table 3.3: Comparison of 3 Estimates of β (True Value=0.2121)
27 Teams, each playing 82 games

	ρ=0.1	0.3	0.5	0.7	0.9
$\widehat{\beta}_{TM}$	0.2170	0.2107	0.1918	0.1597	0.1068
$SE(\widehat{\beta}_{TM})$	0.0414	0.0412	0.0408	0.0402	0.0404
$SD(\widehat{\beta}_{TM})$	0.0434	0.0417	0.0414	0.0380	0.0372
$\widehat{\beta}_U$	0.2173	0.2128	0.1984	0.1718	0.1224
$SE(\widehat{\beta}_U)$	0.0415	0.0414	0.0413	0.0414	0.0427
$SD(\widehat{\beta}_U)$	0.0435	0.0414	0.0414	0.0395	0.0401
$\widehat{\beta}_V$	0.2173	0.2128	0.1985	0.1719	0.1225
$SE(\widehat{\beta}_V)$	0.0415	0.0414	0.0413	0.0414	0.0427
$SD(\widehat{\beta}_V)$	0.0435	0.0413	0.0414	0.0395	0.0402
$\widehat{\rho}_U$	0.0397	0.1875	0.3353	0.5057	0.7055
$SE(\widehat{\rho}_U)$	0.0567	0.0563	0.0562	0.0566	0.0609
$SD(\widehat{\rho}_U)$	0.0544	0.0563	0.0567	0.0598	0.0646
$\widehat{\theta}_V$	0.0470	0.2217	0.3953	0.5920	0.8273
$SE(\widehat{\theta}_V)$	0.0672	0.0666	0.0663	0.0664	0.0715
$SD(\widehat{\theta}_V)$	0.0647	0.0667	0.0667	0.0695	0.0751

For each value of ρ, the true parameter values (used to generate the data) are given, as well as estimates from the Thurstone-Mosteller (TM) method (ignoring ρ), approximate maximum likelihood estimates generated by the MCEM Algorithm, estimates from the methods of Section 3.3 which use the covariate U, and estimates from the method of Section 3.4 using the covariate V, respectively. Standard errors for all parameter estimates are given in parentheses, with the exception of the estimates calculated using the MCEM Algorithm. Estimates and standard errors for the correlation coefficient, or the covariate for V, are given at the bottom of the table.

Table 3.4: Comparison of 4 methods for estimates of β and 13 α's

	$\rho = 0.1$				$\rho = 0.3$			
$\alpha(true)$	$\widehat{\alpha}_{TM}$	$\widehat{\alpha}_{ML}$	$\widehat{\alpha}_{U}$	$\widehat{\alpha}_{V}$	$\widehat{\alpha}_{TM}$	$\widehat{\alpha}_{ML}$	$\widehat{\alpha}_{U}$	$\widehat{\alpha}_{V}$
0.3*	0.33	0.33	0.33	0.33	0.29	0.30	0.29	0.29
	(0.06)		(0.06)	(0.06)	(0.06)		(0.06)	(0.06)
1.2	1.02	1.01	1.02	1.02	1.16	1.23	1.19	1.19
	(0.21)		(0.21)	(0.21)	(0.21)		(0.21)	(0.21)
0.4	0.44	0.42	0.45	0.45	0.31	0.32	0.33	0.33
	(0.21)		(0.21)	(0.21)	(0.21)		(0.21)	(0.21)
1.6	1.74	1.72	1.74	1.74	1.72	1.77	1.74	1.73
	(0.22)		(0.22)	(0.22)	(0.21)		(0.22)	(0.22)
2.1	2.28	2.27	2.28	2.28	2.22	2.29	2.24	2.24
	(0.23)		(0.23)	(0.23)	(0.23)		(0.23)	(0.23)
0.2	0.34	0.34	0.35	0.35	-0.06	-0.04	-0.06	-0.06
	(0.21)		(0.21)	(0.21)	(0.22)		(0.22)	(0.22)
0.9	1.01	1.00	1.01	1.01	0.83	0.88	0.84	0.84
	(0.21)		(0.21)	(0.21)	(0.21)		(0.21)	(0.21)
1.5	1.58	1.60	1.59	1.59	1.65	1.73	1.69	1.69
	(0.21)		(0.21)	(0.21)	(0.21)		(0.22)	(0.22)
1.6	1.83	1.86	1.83	1.83	1.64	1.73	1.68	1.67
	(0.22)		(0.22)	(0.22)	(0.21)		(0.22)	(0.22)
0.6	0.73	0.70	0.73	0.73	0.72	0.73	0.72	0.72
	(0.21)		(0.21)	(0.21)	(0.21)		(0.21)	(0.21)
1.85	2.12	2.10	2.12	2.12	1.69	1.74	1.72	1.71
	(0.22)		(0.22)	(0.22)	(0.21)		(0.22)	(0.22)
0.9	1.03	1.04	1.04	1.04	0.78	0.81	0.79	0.79
	(0.21)		(0.21)	(0.21)	(0.21)		(0.21)	(0.21)
1.0	1.14	1.13	1.15	1.15	1.07	1.12	1.09	1.09
	(0.21)		(0.21)	(0.21)	(0.21)		(0.21)	(0.21)
0.65	0.73	0.71	0.74	0.74	0.84	0.89	0.84	0.84
	(0.21)		(0.21)	(0.21)	(0.21)		(0.21)	(0.21)
ρ		0.08	0.06			0.27	0.27	
			(0.06)				(0.06)	
θ				0.08				0.32
				(0.07)				(0.07)

Table 3.4 Cont'd

	$\rho = 0.4$				$\rho = 0.6$			
$\alpha(true)$	$\widehat{\alpha}_{TM}$	$\widehat{\alpha}_{ML}$	$\widehat{\alpha}_U$	$\widehat{\alpha}_V$	$\widehat{\alpha}_{TM}$	$\widehat{\alpha}_{ML}$	$\widehat{\alpha}_U$	$\widehat{\alpha}_V$
0.30*	0.32	0.34	0.34	0.34	0.23	0.26	0.26	0.25
	(0.06)		(0.06)	(0.06)	(0.06)		(0.06)	(0.06)
1.2	0.91	0.92	0.94	0.94	1.25	1.42	1.30	1.28
	(0.20)		(0.21)	(0.21)	(0.21)		(0.22)	(0.22)
0.4	0.19	0.19	0.22	0.22	0.87	0.97	0.90	0.89
	(0.21)		(0.21)	(0.21)	(0.21)		(0.22)	(0.22)
1.6	1.47	1.54	1.53	1.52	1.76	1.99	1.86	1.84
	(0.21)		(0.21)	(0.21)	(0.22)		(0.23)	(0.23)
2.1	1.85	1.97	1.94	1.93	2.11	2.45	2.23	2.22
	(0.22)		(0.22)	(0.22)	(0.22)		(0.23)	(0.23)
0.2	0.71	0.70	0.74	0.73	0.12	0.05	0.09	0.08
	(0.20)		(0.21)	(0.21)	(0.22)		(0.23)	(0.23)
0.9	0.75	0.76	0.80	0.80	1.37	1.57	1.46	1.44
	(0.20)		(0.21)	(0.21)	(0.21)		(0.22)	(0.22)
1.5	1.63	1.68	1.68	1.68	1.83	2.10	1.90	1.89
	(0.21)		(0.21)	(0.21)	(0.22)		(0.23)	(0.23)
1.6	1.49	1.61	1.57	1.56	1.62	1.87	1.70	1.68
	(0.21)		(0.21)	(0.21)	(0.22)		(0.22)	(0.22)
0.6	0.41	0.41	0.45	0.44	0.81	0.88	0.85	0.84
	(0.21)		(0.21)	(0.21)	(0.21)		(0.22)	(0.22)
1.85	1.38	1.47	1.45	1.44	2.03	2.32	2.13	2.12
	(0.21)		(0.21)	(0.21)	(0.22)		(0.23)	(0.23)
0.9	0.48	0.45	0.49	0.49	1.24	1.34	1.26	1.25
	(0.21)		(0.21)	(0.21)	(0.21)		(0.22)	(0.22)
1.0	0.91	0.95	0.93	0.93	1.42	1.66	1.49	1.47
	(0.20)		(0.21)	(0.21)	(0.21)		(0.22)	(0.22)
0.65	0.42	0.44	0.44	0.44	0.92	1.00	0.94	0.92
	(0.20)		(0.21)	(0.21)	(0.21)		(0.22)	(0.22)
ρ		0.39	0.41			0.58	0.63	
			(0.06)				(0.06)	
θ				0.48				0.75
				(0.07)				(0.07)

Table 3.4 Cont'd

$\rho = 0.75$				
$\alpha(true)$	$\widehat{\alpha}_{TM}$	$\widehat{\alpha}_{ML}$[1]	$\widehat{\alpha}_U$	$\widehat{\alpha}_V$
0.30*	0.29	0.38	0.33	0.35
	(0.06)		(0.06)	(0.06)
1.2	0.96	1.54	0.80	1.10
	(0.20)		(0.22)	(0.22)
0.4	0.69	1.06	0.57	0.79
	(0.20)		(0.21)	(0.22)
1.6	1.62	2.38	1.33	1.84
	(0.21)		(0.22)	(0.23)
2.1	1.49	2.30	1.32	1.79
	(0.21)		(0.22)	(0.23)
0.2	0.35	0.52	0.32	0.43
	(0.20)		(0.22)	(0.22)
0.9	0.28	0.45	0.25	0.38
	(0.20)		(0.22)	(0.22)
1.5	0.97	1.54	0.85	1.17
	(0.20)		(0.22)	(0.22)
1.6	0.98	1.53	0.84	1.12
	(0.20)		(0.22)	(0.22)
0.6	0.09	0.20	0.05	0.10
	(0.20)		(0.22)	(0.22)
1.85	1.28	1.92	1.00	1.42
	(0.20)		(0.22)	(0.22)
0.9	0.72	1.12	0.65	0.89
	(0.20)		(0.21)	(0.22)
1.0	0.84	1.27	0.72	0.92
	(0.20)		(0.21)	(0.22)
0.65	0.34	0.53	0.24	0.38
	(0.20)		(0.21)	(0.22)
ρ		0.77	0.74	
			(0.05)	
θ				1.08
				(0.07)

**Indicates the common home field advantage for all items.*
[1] *These values failed to converge after 135 iterations.*

From the above table, the following can be observed:

- For $\rho \leq 0.6$ standard errors for all three methods differ by less than 5% (lower for lower values of ρ) with the usual Thurstone-Mosteller Model having the lowest standard errors.
- For $\rho = 0.75$ the standard errors from the usual Thurstone-Mosteller Model tend to be at least 10% lower than the standard errors from the methods of Sections 3.3 and 3.4.
- Parameter estimates are similar for all three methods if $\rho \leq 0.3$.
- The two approximation methods give similar results if $\rho \leq 0.6$, while at high levels of ρ the method from Section 3.4 yields much higher estimates than the method of Section 3.3.
- There is some evidence that while the approximation methods tend to underestimate the strength parameters for the stronger teams, estimates from the usual Thurstone-Mosteller Model contain a great deal of bias for all strength parameters with the exception of those corresponding to the poorer teams (estimates closer to zero). In fact, only the method of Section 3.3 consistently provides estimates which remain within a single standard deviation of the estimates calculated using the MCEM Algorithm.

(Note that when comparing Tables 3.1 – 3.3 with Table 3.4, estimates from the approximation models and usual Thurstone-Mosteller Model in the latter table have been multiplied by the $\sqrt{2}$ so that they can be compared on the same scale to the estimates calculated from the MCEM Algorithm.)

From these tables, there seems to be little reason not to use the usual Thurstone-Mosteller Model if inference is to be drawn solely regarding the home field advantage, or if there is evidence that only small amounts of correlation exist between the performance variables. When trying to quantify the amount of correlation or when a moderate to high degree of correlation among the performance variables is present, the approximation model based on the covariate U_i seems to work well.

3.6 Results of MCEM Algorithm Estimation for Major League Baseball Data

Data come from the 1993 Major League Baseball Season in which each of the 14 National League teams play 162 games, playing each other team 12 or 13 times. Data consisted of the observed win-loss outcomes for the home team in each game. If π_i is defined as the probability that the home team wins game i, then the model which estimates separate performance means, a common home field effect, and a common estimate of ρ for each team can be defined by:

$$W_i^H = \sum_{j=2}^{14} \alpha_j X_{ij}^H + \beta + \epsilon_i^H \tag{3.6.1}$$

where

$$\epsilon_i^H = \rho \epsilon_{prev}(T_i^H) + \delta_i^H \tag{3.6.2}$$

and

$$W_i^V = \sum_{j=2}^{14} \alpha_j X_{ij}^V + \epsilon_i^V \tag{3.6.3}$$

where

$$\epsilon_i^V = \rho \epsilon_{prev}(T_i^V) + \delta_i^V \tag{3.6.4}$$

and δ_i^H and δ_i^V are both independent and normally distributed with mean zero and variance one. Then

$$\pi_i = Pr\left[W_i^H - W_i^V > 0\right] \tag{3.6.5}$$

where $X_{ij}^H = 1$ if team j is the home team in the i^{th} game, for $j = 2, ..., 14$, $X_{ij}^V = 1$ if team j is the visiting team in the i^{th} game. The $\alpha's$ represent the individual strength parameters, β estimates the common home field advantage, and ρ the serial correlation. $\epsilon_{prev}(T_i^H)$ and $\epsilon_{prev}(T_i^V)$ are defined as in Section 3.1.1. Estimates from this model are given in Table 3.5 below with standard errors provided in parentheses for each of the estimates.

Table 3.5: Estimates for 1993 Major League Baseball Season

Team	$\widehat{\alpha}_{ML}$	$\widehat{\alpha}_{U}$	$\widehat{\alpha}_{V}$	$\widehat{\alpha}_{TM}$
Home Field Adv.	0.113	0.112	0.112	0.115
		(0.05)	(0.05)	(0.05)
Atlanta (constrained)	0.000	0.000	0.000	0.000
San Francisco	-0.045	-0.021	-0.021	-0.023
		(0.20)	(0.20)	(0.19)
Philadelphia	-0.178	-0.161	-0.161	-0.154
		(0.20)	(0.20)	(0.19)
Montreal	-0.235	-0.208	-0.208	-0.205
		(0.19)	(0.19)	(0.19)
St. Louis	-0.382	-0.352	-0.352	-0.364
		(0.19)	(0.19)	(0.19)
Houston	-0.415	-0.419	-0.418	-0.414
		(0.19)	(0.19)	(0.19)
Chicago	-0.440	-0.424	-0.423	-0.425
		(0.19)	(0.19)	(0.19)
Los Angeles	-0.504	-0.491	-0.490	-0.486
		(0.19)	(0.19)	(0.19)
Pittsburgh	-0.641	-0.617	-0.617	-0.612
		(0.20)	(0.20)	(0.19)
Cincinnati	-0.694	-0.654	-0.654	-0.656
		(0.19)	(0.19)	(0.19)
Colorado	-0.780	-0.784	-0.783	-0.783
		(0.20)	(0.20)	(0.19)
Florida	-0.896	-0.844	-0.844	-0.844
		(0.20)	(0.20)	(0.20)
San Diego	-0.936	-0.905	-0.905	-0.908
		(0.20)	(0.20)	(0.20)
New York	-0.984	-0.956	-0.956	-0.953
		(0.20)	(0.20)	(0.20)
	$\widehat{\rho} =$ -0.147	$\widehat{\rho} =$ -0.156	$\widehat{\theta} =$ -0.177	
		(0.05)	(0.06)	

As was seen in the simulation studies, the estimates for all four methods give similar results. For instance, when using either of the approximation models, the probability that Philadelphia defeats Atlanta in Philadelphia is 0.4826, but is only 0.4235 when the game is played in Atlanta. Likewise the probabilities when using the usual probit regression model are 0.4890 and 0.4246, respectively. Note the significant negative correlation between consecutive performances ($\hat{\rho}$ = -0.156, SE=0.05). This might be due to the effect starting pitching has on game outcomes - due to the increasing number of teams in recent years, good pitchers are more likely to be followed by poor pitchers. This hypothesis could be checked by including, for example, the pitcher's earned run average as a covariate (and seeing whether significant correlation remained after accounting for this measure of pitcher quality).

The actual winning percentages (overall and for home and away games) for each of the teams are given in Table 3.6.

Table 3.6: Team Winning Percentages for 1993 Major League Baseball Season

	Winning Percentage		
Team	*Overall*	*Home*	*Away*
Atlanta	0.642	0.630	0.654
San Francisco	0.636	0.617	0.654
Philadelphia	0.599	0.642	0.556
Montreal	0.580	0.679	0.481
St. Louis	0.537	0.605	0.469
Houston	0.525	0.543	0.506
Chicago	0.519	0.531	0.506
Los Angeles	0.500	0.506	0.494
Pittsburgh	0.463	0.494	0.432
Cincinnati	0.451	0.506	0.395
Colorado	0.414	0.481	0.346
Florida	0.395	0.432	0.358
San Diego	0.377	0.420	0.333
New York	0.364	0.346	0.383

Note that the ranking achieved using the methods discussed above correspond to the ranking based on the overall winning percentages.

3.7 Discussion

The presence of serial correlation among the performance variables is a substantial departure from the usual assumptions used in the analysis of paired comparison experiments. It seems intuitively evident that there may be many situations where there is serial correlation in the "performances" or "merits" of competitors or items that produce paired comparison data. There was significant evidence of serial correlation in the major league baseball example (the first serial correlation coefficient between team performance was estimated to be -0.156 with a standard error of 0.05). In some problems there may be interest in simply testing and estimating such a parameter. The method of Section 3.3 based on the inclusion of a "serial correlation covariate" into a probit regression model is a simple way to accomplish this. Limited simulation studies suggest it is fairly good at this task.

More often the effects of explanatory variables on performance are to be examined. In these cases the method of Section 3.3 (or the simpler method of Section 3.4) can be used as a check of first-order auto correlation. Simulation studies have indicated that, in at least some situations, the usual Thurstone-Mosteller analysis can be misleading in the presence of serial correlation. A test for the significance of ρ (or θ) may provide some assurance that the assumptions behind the usual Thurstone-Mosteller or Bradley-Terry analysis are satisfied. If the estimated ρ is large, then the methods of Sections 3.3 and 3.4 offer a way to make inferences about the regression coefficients while accounting for serial correlation. Evidence from simulation studies, however, suggest that these approximation methods offer only modest bias correction. Nevertheless, these methods provide, at least, a first approach towards the solution of this problem. It is felt that these methods may be most useful when ρ is large enough so that there are worries about the standard analysis, yet not so large that their bias is severe.

The use of the EM Algorithm has some appeal in that the updated estimates at each iteration are very similar to the usual least squares estimates based on filtered variables that are used to account for auto-correlation. The drawback, however, is having to compute the expectations of the underlying performances for the home and visiting team, conditional on which team wins that game *and* conditional on the outcomes of other games (at least the games in the recent past and the near future for each team). The MCEM Algorithm is straightforward, in principle, for this since the Monte Carlo expectations are simply averages over simulated games with similar outcomes, but the computation time has proved to be prohibitive. Because of this, it has not been possible to calculate either the maximized likelihood or the standard errors of the maximum likelihood estimates. (The SEM Algorithm (Meng and Rubin, 1991) may provide standard errors when convergence can be obtained quickly and a small number of parameters are being estimated.) Further approximations, methods for accelerating the EM Algorithm, and faster computers might become available in the future, which may alleviate the computational problems to the extent that approximate likelihood analysis can be carried out.

References:

Anderson, R.L. (1954). The problem of autocorrelated data in regression analysis. *Journal of the American Statistical Association.* **49**, 113-129.

Batchelder, William H., and Bershad, Neil J. (1979). The statistical analysis of a Thurstonian model for rating chess players. *Journal of Mathematical Psychology.* **19**, 39-60.

Bradley, R. A. (1984). Paired comparisons: some basic procedures and examples. *Handbook of Statistics, Vol. 4.* ed. P. R. Krishnaiah and P. K. Sen. Amsterdam: North-Holland, 299-326.

Bradley, R. A. and Terry, M. E. (1952). The rank analysis of incomplete block designs. I. The method of paired comparisons. *Biometrika.* **39**, 324-345.

Critchlow, D. E. and Fligner, M. A. (1991) Paired comparison, triple comparison, and ranking experiments as generalized linear models, and their implementation on GLIM. *Psychometrika.* **56**, 517-533.

Cox, D.R. (1970). *Analysis of Binary Data.* Chapman and Hall. London.

Cox, D.R. (1981). Statistical analysis of time series: some recent developments. *Scandinavian Journal of Statistics.* **8**, 93-115.

David, Herbert A. (1988). *The Method of Paired Comparisons, 2nd ed.* Charles Griffen and Company LTD, London.

Davidson, R. R. and Farquhar, P. H. (1976). A bibliography on the method of paired comparisons. *Biometrics.* **32**, 241-252.

Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977). Maximum likelihood from incomplete data via the EM Algorithm. *Journal of the Royal Statistical Society (B).* **39**, 1-22.

Dillon, William R., Kumar, Ajith and Smith de Borrero, Melinda. (1993). Capturing individual differences in paired comparisons: an extended BTL model incorporating descriptor variables. *Journal of Marketing Research.* **30**, 42-51.

Elo, Arpad E. (1978). *The Rating of Chess Players: Past and Present.* Arco Publishing, Inc., New York.

Glickman, Mark E. (1993). Paired comparison models with time-varying parameters. Unpublished Thesis. Department of Statistics, Harvard University.

Henery, Robert J. (1992). An extension to the Thurstone-Mosteller model for chess. *The Statistician.* **41**, 559-567.

Joe, Harry. (1990). Extended use of paired comparison models, with application to chess rankings. *Applied Statistics.* **39**, 85-93.

Joe, Harry and White, Rick A. (1992). Paired comparison models with time-varying abilities for chess. Technical Report. Department of Statistics, University of British Columbia.

Kolsky, James D. and Schafer, Daniel W. (1996). Thurstone's case III model for paired comparisons. Unpublished.

Meng, X.L. and Rubin, D.B. (1993). Maximum likelihood estimation via the ECM algorithm: A general framework. *Biometrika.* **80**, 267-278.

Mosteller, F. (1951a). Remarks on the method of paired comparisons: I. The least squares solution assuming equal standard deviations and equal correlations. *Psychometrika.* **16**, 3-9.

Mosteller, F. (1951b). Remarks on the method of paired comparisons: II. The effect of an aberrant standard deviation when equal standard deviations and equal correlations are assumed. *Pyschometrika.* **16**, 203-206.

Mosteller, F. (1951c). Remarks on the method of paired comparisons: III. A test of significance for paired comparisons when equal standard deviations and equal correlations are assumed. *Psychometrika.* **16**, 207-218.

Thurstone, L.L. (1927). Psychophysical analysis. *American Journal of Psychology.* **38**, 368-389.

Wei, Greg C.G. and Tanner, Martin A. (1990). A monte carlo implementation of the EM Algorithm and the poor man's data augmentation algorithms. *Journal of the American Statistical Association.* **85**, 699-704.

West, M., Harrison, P.J. and Mignon, H.S. (1985). Dynamic generalized linear models and Bayesian forecasting. *Journal of the American Statistical Association.* **80**, 73-96.

Zeger, Scott L. and Qaqish, Bahjat. (1988). Markov Regression Models for Time Series: A Quasi-Likelihood Approach. *Biometrics.* **44**, 1019-1031.

4. *Conclusions*

4.1 Summary

This thesis clarifies the use of probit and logistic regression models for the analysis of paired comparison data and extends the usual Thurstone-Mosteller Model to include non-constant performance variances and serial correlation among the performance variables.

In Chapter 2 we discuss the use of probit and logistic regression to analyze paired comparison data when the goal is to rank the items after accounting for explanatory variables, or to model the effect of explanatory variables on the ranking. Specifically, we addressed the analysis of unbalanced paired comparison data such as that from the National Basketball Association. Much of the literature focuses on completely balanced designs. We have also included time-dependent explanatory variables, such as which team is the home team or attendance at each game. Previous work in the literature has focused on time-independent variables, which by definition must remain constant over all comparisons. The use of logistic and probit regression also permits response-surface methodology. This was demonstrated with Atkinson's (1972) data in which an optimal level of MSG was determined according to taste preference. Goodness-of-fit tests, such as the drop in deviance test, are immediate consequences of probit and logistic regression as well. Thus, simple methods exist to determine the adequacy of either the Bradley-Terry or Thurstone-Mosteller Models. When minor model inadequacies are found, quasi-likelihood, as demonstrated in Section 2.2.1, can be used to account for problems. This is particularly useful since some excess variation may be incorporated into the data due to such things as judge effects or the decision process not being completely linear. For instance, in the applesauce data of Section 2.2.1, preference may be a function of both taste and texture, rather than a linear function of taste as assumed by both the Bradley-Terry and Thurstone-Mosteller Models.

Chapter 2 also discusses estimation of performance variances for Thurstone's Case III Model. Methodology is examined which permits maximum likelihood estimation and likelihood ratio inference. Estimation of the performance variances using the EM Algorithm provides a solution to a problem that has interested researchers since the 1930's. Since then, several attempts have been made to estimate the performance variances, resulting in several approximate methods. This model also provides a check of the assumptions made in the standard probit model, a method of drawing inference about the differing performance variances, and a method of drawing inference about the regression coefficients in the presence of differing variances.

In Chapter 3 approximate maximum likelihood estimation is presented for paired comparison analysis when performances for a given item are serially correlated. The approximation is due to the need to simulate the conditional expectations at each E-step. Because of computational difficulties, neither the maximized likelihood nor standard errors of the estimates can be calculated. As a result, we present two methods, which can be used with standard statistical packages and some minor programming, to approximate the maximum likelihood estimates. When low levels of correlation are present between performances, all of the models, including the usual Thurstone-Mosteller analysis (ignoring ρ), provide similar results. The approximation models do, however, offer some bias reduction over the usual Thurstone-Mosteller Model, especially for higher levels of correlation, and provide a simple and quick method of analysis when usual methods are suspect due to the presence of serial correlation. Use of the approximation models also provides a check of the assumption of independence usually assumed under standard analysis techniques and provides a method of estimating the regression coefficients after accounting for the serial correlation.

4.2 Further Work

Several issues remain unresolved. Simulation studies indicate that performance variances are very biased when the number of replicate comparisons between items is small. In addition, the EM Algorithm struggles to converge when a large number of parameters are simultaneously estimated. The algorithm is also sensitive to changes in the initial estimates. In some cases changes in these initial estimates caused large changes in the parameter estimates with only small changes in the maximized likelihood.

While neither of the approximation models of Chapter 3 do a great job of estimating parameters for moderate to high levels of correlation, they do represent some improvement over the standard paired comparisons analysis. Methods that accelerate the EM Algorithm, further approximations, or faster computers in the future may make likelihood inference using the MCEM Algorithm more attractive. Individual correlation coefficients and individual performance variances could be estimated using the MCEM Algorithm as well. However, the computational time and amount of data that would be required for this model make this a theoretical exercise rather than a practical application.

The methods here have been limited, primarily, to the Thurstone-Mosteller Model. However, these methods also apply to the Bradley-Terry Model if the normal distribution of the performance variables is replaced by a logistic distribution. Lastly, the issue of ties has been avoided in this thesis. The methods here, however, could be extended to chess data, which has received a great deal of attention in the statistical literature, by viewing the problem in a multinomial framework.

BIBLIOGRAPHY

Abelson, R. M. and Bradley, R. A. (1954). A 2x2 factorial with paired comparisons. *Biometrics*. **10**, 487-502.

Anderson, R.L. (1954). The problem of autocorrelated data in regression analysis. *Journal of the American Statistical Association*. **49**, 113-129.

Atkinson, A. C. (1972). A test of the linear logistic and Bradley-Terry models. *Biometrika*. **59**, 37-42.

Batchelder, William H., and Bershad, Neil J. (1979). The statistical analysis of a Thurstonian model for rating chess players. *Journal of Mathematical Psychology*. **19**, 39-60.

Beaver, R. J. and Gokhale, D. V. (1975). A model to incorporate within-pair order effects in paired comparisons. *Communications in Statistics - Theor. Meth.* **4**, 923-939.

Bradley, R. A. (1984). Paired comparisons: some basic procedures and examples. *Handbook of Statistics, Vol. 4*. ed. P. R. Krishnaiah and P. K. Sen. Amsterdam: North-Holland, 299-326.

Bradley, R. A. and Terry, M. E. (1952). The rank analysis of incomplete block designs. I. The method of paired comparisons. *Biometrika*. **39**, 324-345.

Bradley, R. A. and El-Helbawy, A. T. (1976). Treatment contrasts in paired comparisons: Basic procedures with applications to factorials. *Biometrika*, **63**, 255-262.

Burros, R.H. (1951). The application of the method of paired comparisons to the study of reaction potential. *Psychology Review*. **58**, 60-66.

Burros, R. H. and Gibson, W. A. (1954). A solution for case III of the law of comparative judgment. *Psychometrika*. **19**, 57-64.

Cox, D.R. (1970). *Analysis of Binary Data*. Chapman and Hall. London.

Cox, D.R. (1981). Statistical analysis of time series: some recent developments. *Scandinavian Journal of Statistics*. **8**, 93-115.

Critchlow, D. E. and Fligner, M. A. (1991) Paired comparison, triple comparison, and ranking experiments as generalized linear models, and their implementation on GLIM. *Psychometrika*. **56**, 517-533.

David, Herbert A. (1987). Rankings from unbalanced paired comparison data. *Biometrika*. **74**, 432-436.

David, Herbert A. (1988). *The Method of Paired Comparisons, 2nd ed.* Charles Griffen and Company LTD, London.

Davidson, R. R. (1970). On extending the Bradley-Terry model to accommodate ties in paired comparison experiments. *Journal of the American Statistical Association*. **65**, 317-328.

Davidson, R. R. and Beaver, R. J. (1977). On extending the Bradley-Terry model to incorporate within-pair order effects. *Biometrics*. **33**, 245-254.

Davidson, R.R. and Bradley, R. A. (1970). Multivariate paired comparisons: Some large sample results on estimation and test of equality of preference. In: M.L. Puri, ed., *Nonparametric Techniques in Statistical Inference*. Cambridge, University Press, 111-125.

Davidson, R. R. and Farquhar, P. H. (1976). A bibliography on the method of paired comparisons. *Biometrics*. **32**, 241-252.

Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977). Maximum likelihood from incomplete data via the EM Algorithm. *Journal of the Royal Statistical Society (B)*. **39**, 1-22.

Dillon, William R., Kumar, Ajith and Smith de Borrero, Melinda. (1993). Capturing individual differences in paired comparisons: an extended BTL model incorporating descriptor variables. *Journal of Marketing Research*. **30**, 42-51.

Dykstra, O. (1960). Rank analysis of incomplete block designs: A method of paired comparisons employing unequal repetitions on pairs. *Biometrics*. **16**, 176-188.

Edwards, A.W.F. (1972). *Likelihood: An account of the statistical concept of likelihood and its application to scientific inference*. Cambridge.

Elo, Arpad E. (1978). *The Rating of Chess Players: Past and Present.* Arco Publishing, Inc., New York.

Fienberg, S.E. (1979). Log linear representation for paired comparison models with ties within-pair order effects. *Biometrics*. **35**, 479-481.

Fienberg, S.E. and Larntz, K. (1976). Log linear representation for paired and multiple comparison models. *Biometrika.* **63**, 245-254.

Gibson, W. A. (1953). A least squares solution for case IV of the law of comparative judgment. *Psychometrika.* **18**, 15-21.

Glenn, W. A. and David, H. A. (1960). Ties in paired-comparison experiments using a modified Thurstone-Mosteller method. *Biometrics.* **16**, 86-109.

Glickman, Mark E. (1993). Paired comparison models with time-varying parameters. Unpublished Thesis. Department of Statistics, Harvard University.

Greenberg, M.G. (1965). A modification of Thurstone's law of comparative judgment to accommodate a judgment category of "equal" or "no difference." *Psychological Bulletin.* **64**, 108-112.

Guilford, J. P. (1954). *Psychometric Methods, 2nd ed.* McGraw-Hill, New York.

Harris, W. P. (1957). A revised law of comparative judgment. *Psychometrika.* **22**, 189-198.

Henery, Robert J. (1992). An extension to the Thurstone-Mosteller model for chess. *The Statistician.* **41**, 559-567.

Jackson, J.E. and Fleckenstein M. (1957). An evaluation of some statistical techniques used in the analysis of paired comparison data. *Biometrics.* **13**, 51-64.

Joe, Harry. (1990). Extended use of paired comparison models, with application to chess rankings. *Applied Statistics.* **39**, 85-93.

Joe, Harry and White, Rick A. (1992). Paired comparison models with time-varying abilities for chess. Technical Report. Department of Statistics, University of British Columbia.

Kendall, M.G. (1955). Further Contributions to the theory of paired comparisons. *Biometrics.* **11**, 43-62.

Kendall, M.G. and Babington Smith, B. (1940). On the method of paired comparisons. *Biometrika.* **31**, 324-345.

Kolsky, James D. and Schafer, Daniel W. (1996). Thurstone's case III model for paired comparisons. Unpublished.

Kousgaard, Nils. (1979). A conditional approach to the analysis of data from paired comparison experiments incorporating within-pair order effects. *Scandinavian Journal of Statistics.* **6**, 154-60.

Kousgaard, Nils. (1984). Analysis of a sound field experiment by a model for paired comparisons with explanatory variables. *Scandinavian Journal of Statistics.* **11**, 51-57.

Latta, R. B. (1979). Composition rules for probabilities from paired comparisons. *Annals of Statistics.* **7**, 349-371.

Lewis, Thomas, A. (1982). Finding the observed information matrix when using the EM Algorithm. *Journal of the Royal Statistical Society, Series B.* **44**, 226-233.

Lindsey, J.K. (1989). *The Analysis of Categorical Data Using GLIM (Lecture Notes in Statistics, Volume 56).* Springer-Verlag, New York.

McCormick, E.J. and Bachus, J.A. (1952). Paired comparison Ratings. I. The effect on ratings of reductions in the number of pairs. *Journal Applied Psychology.* **36**, 123-127.

McCullagh, P. and Nelder, J.A. (1989). *Generalized Linear Models, 2nd ed.* Chapman & Hall, London.

Meng, X.L. and Rubin, D.B. (1993). Maximum likelihood estimation via the ECM algorithm: A general framework. *Biometrika.* **80**, 267-278.

Mosteller, F. (1951a). Remarks on the method of paired comparisons: I. The least squares solution assuming equal standard deviations and equal correlations. *Psychometrika.* **16**, 3-9.

Mosteller, F. (1951b). Remarks on the method of paired comparisons: II. The effect of an aberrant standard deviation when equal standard deviations and equal correlations are assumed. *Pyschometrika.* **16**, 203-206.

Mosteller, F. (1951c). Remarks on the method of paired comparisons: III. A test of significance for paired comparisons when equal standard deviations and equal correlations are assumed. *Psychometrika.* **16**, 207-218.

Raghavarao, D. (1971). *Construction and Combinatorial Problem in Design of Experiments.* Wiley, New York.

Rao, P. V. and Kupper, L. L. (1967) Ties in paired-comparison experiments: A generalization of the Bradley-Terry model. *Journal of the American Statistical Association.* **62**, 194-204, Corrigenda **63**, 1550.

Sadasivan, G. (1982). A Thurstone-type model for paired comparisons with unequal numbers of repetitions. *Communications in Statistics - Theor. Meth.* **11**, 821-833.

Sinclair, C.D. (1982). GLIM for preference. *GLIM 82: Proceedings of the international conference on generalized linear models: Lecture notes in Statistics, Volume 14.* ed. R. Gilchrist. Springer-Verlag, New York, 164-178.

The Sporting News: Baseball Guide, 1994 Edition. ed. Craig Carter and Dave Sloan. The Sporting New Publishing Company, St. Louis.

Thurstone, L.L. (1927a). Psychophysical analysis. *American Journal of Psychology.* **38**, 368-389.

Thurstone, L.L. (1927b). The method of paired comparisons for social values. *Journal of Abnormal and Social Psychology.* **21**, 384-400.

Thurstone, L.L. (1932). Stimulus dispersions in the method of constant stimuli. *Journal of Experimental Psychology.* **15**, 284-97.

Torgerson, Warren S. (1960) *Theory and Methods of Scaling.* John Wiley, New York.

Uhrbrock, R.S. and Richardson, M.W. (1933). Item analysis. *Person. J.* **12**, 141-154.

Wei, Greg C.G. and Tanner, Martin A. (1990). A monte carlo implementation of the EM Algorithm and the poor man's data augmentation algorithms. *Journal of the American Statistical Association.* **85**, 699-704.

West, M., Harrison, P.J. and Mignon, H.S. (1985). Dynamic generalized linear models and Bayesian forecasting. *Journal of the American Statistical Association.* **80**, 73-96.

Zeger, Scott L. and Qaqish, Bahjat. (1988). Markov Regression Models for Time Series: A Quasi-Likelihood Approach. *Biometrics.* **44**, 1019-1031.

Zermelo, E. (1929). Die Berechnung der Turnier-Ergebnisse als ein maximumproblem der wahrscheinlichkeitsrechnung. *Math. Zeit.* **29**, 436-460.

APPENDICES

APPENDIX 1 Programs required for the estimation performed in Chapter 2.

```
# FILE NAME = VARSHELL.SPS

# THIS PROGRAM RUNS THE ESTIMATION PROGRAMS USED IN THE FIRST
# ARTICLE WHICH ALLOWED INDIVIDUAL VARIANCE PARAMETERS TO BE
# ESTIMATED. NOTE: IF THE VARIANCE IS ASSUMED TO BE CONSTANT THEN
# THIS FUNCTION DOES NOT NEED TO BE USED. INSTEAD PROBIT OR LOGISTIC
# REGRESSION CAN BE USED. THE VARIABLES ARE DEFINED IN THE
# FUNCTIONS THAT USE THEM. INITIAL AND INITIAL.SIGMA ARE THE INITIAL
# ESTIMATES FOR THE MEANS AND VARIANCES, RESPECTIVELY.


EM.var_ function(team,opp,win,indicator,cov,numt,numc,initial,initial.sigma,
        UNIVARIATE=T,IND=T,MEANS=T)     {


ngames.out_ ngames.sps(team,numt)
ngames_ ngames.out              # calculate the number of games
                                # each team plays.

x.out_ createx.sps(team,numt,numc,cov,indicator,IND)
x_ x.out        # create the design matrix.


alpha_ initial
sigma_ initial.sigma  # initial estimates

iterations_ 0
test_ 10
ldiff_ 10
lik_ 0

while (test > 0.005 | ldiff > 0.01)  {

e.out_ Estep.sps(team,opp,indicator,cov,numt,numc,alpha,sigma,
        UNIVARIATE, MEANS)
PHI_ e.out$PHI
zstar_ e.out$zstar
vstar_ e.out$vstar

m.out_ Mstep.sps(x,team,numt,numc,zstar,vstar,alpha,sigma,MEANS)
alpha_ m.out$alpha
sigma_ m.out$sigma
test_ m.out$test
```

```
lik.new_ (sum(win*log(PHI) + (1-win)*log(1-PHI))/2)
ldiff_ lik.new - lik
lik_ lik.new
iterations_ iterations + 1


print(alpha,5)
print(sigma,5)
print(test,5)
print(lik,5)
print(iterations,2)


                        } # end of while loop


return(alpha,sigma,lik,iterations)

          } # end of function



# FILE NAME : CREATEX.SPS

# THIS PROGRAM CREATES THE DESIGN MATRIX BASED ON THE TEAMS
# INVOLVED AND ANY COVARIATES THAT ARE OBSERVED AT EACH GAME.
# THE FOLLOWING VECTORS MUST BE READ INTO THE FUNCTION:

#  TEAM = VECTOR OF TEAM LABELS FOR EACH GAME
#  NUMT, NUMC = NUMBER OF TEAMS AND COVARIATES IN THE DATA
#  COV = VECTOR OF COVARIATES THAT ARE OBSERVED
#  IND = INDICATOR VARIABLE EQUAL TO "T" IF THE COVARIATE IS THE
#        HOME FIELD ADVANTAGE
#  INDICATOR = HOME FIELD ADVANTAGE

createx.sps_function(team,numt,numc,cov,indicator,IND)  {

x_ matrix(0,length(team),numt-1)

for (i in 1:(numt-1))  {
 for (j in 1:length(team))  {
  if (team[j]==i+1) x[j,i]_1
                    } # end of the "j" loop
                } # end of the "i" loop
```

```
# DESIGN MATRIX FOR A SIMPLE RANKING MODEL


if (numc==1) {

des_cov*indicator
if (IND==T) cov_rep(1,length(team))
x_ cbind(x, des)
        } # end of the "if" loop

if (numc > 1) {

des_ cov
for (i in 1:numc) des[,i]_ indicator*des[,i]
x_ cbind(x, des)

if (IND==T)  cov_cbind(rep(1,length(team)), cov[,2:numc])
          } # end of the "if" loop


return(x)

# note that this will not provide a design matrix if
# parameters are wanted for each individual team for a
# particular covariate.  This function is meant only to provide
# a design matrix for some simple models.  The design matrix
# for more complicated models can easily be formed outside
# the programs here and with some minor modifications the
# new design matrix can be passed into the function.


                        } # end of the function




# FILE NAME = NGAMES.SPS

# CALCULATES THE NUMBER OF GAMES EACH TEAM PLAYS.  THIS COULD ALSO
# BE IMPORTED BY THE USER AS WELL IF THE NUMBER ARE KNOWN.

# THE VARIABLES THAT WILL BE NEEDED ARE:
#  TEAM = LISTING OF THE TEAM INVOLVED IN EACH GAME
#  NUMT = NUMBER OF TEAMS IN THE LEAGUE
```

```
ngames.sps_ function(team, numt)   {

 ngames_ rep(0, numt)

 for (i in 1:numt) {
  for (j in 1:length(team)) {

  if (team[j]==i) ngames[i]_ngames[i] + 1  # count the games that
                                            # team i is involved in.
                  }
            }

return(ngames)

                  } # end of function




# FILE NAME = ESTEP.SPS

# THIS FUNCTION PERFORMS THE EXPECTATION STEP OF THE EM ALGORITHM
# USING THE ESTIMATES FROM THE PREVIOUS ITERATION. THE EXPECTATIONS
# ARE EXACT AND HAVE BEEN DERIVED ELSEWHERE.

# THE VARIABLES THAT ARE TO BE PASSED INTO THE FUNCTION ARE:

#  TEAM, OPP = VECTORS INDICATING THE TWO TEAMS INVOLVED IN EACH
#              GAME.
#  INDICATOR = VECTOR INDICATING THE HOME TEAM IN EACH GAME
#  COV = THE COVARIATES THAT ARE OBSERVED FOR EACH GAME.  THIS COULD
#        BE EITHER A VECTOR OR A MATRIX.
#  NUMT, NUMC = NUMBER OF TEAMS AND THE NUMBER OF COVARIATES.
#  ALPHA, SIGMA = THE CURRENT PARAMETER ESTIMATES.
#  UNIVARIATE=T: INDICATES THAT ONLY A SINGLE COVARIATE IS TO BE
#               ESTIMATED.
#  MEANS=T:  INDICATES THAT ONLY A RANKING MODEL IS DESIRED (NUMC=0)



Estep.sps_ function(team,opp,indicator,cov,numt,numc,alpha,sigma,
        UNIVARIATE,MEANS)  {
```

```
scale2_ numeric(length(team))
scale3_ numeric(length(team))
scale_ numeric(length(team))

factor_ numeric(length(team))
zstar_ numeric(length(team))
zstar.sq_ numeric(length(team))

phi_ numeric(length(team))
PHI_ numeric(length(team))  # initialize the needed vectors.


# The scale vectors are place-holders that represent the portion of
# the mean that comes from the covariates.  These are based on the
# premise that the covariates are measured for both teams.  For instance,
# home field advantage.

if (MEANS==F)   {  # skip the following if only a ranking model is desired.

if (UNIVARIATE==F)   {
  scale2_ rep(0, length(team))
  scale3_ rep(0, length(team))
  for (i in 1:numc)   {
    scale2_ alpha[numt+i]*indicator*cov[,i] + scale2
    scale3_ alpha[numt+i]*(1-indicator)*cov[,i] + scale3
                 } # end of "i" loop
               }  # end of "if" loop

else  {
 scale2_ alpha[numt+numc]*indicator*cov
 scale3_ alpha[numt+numc]*(1-indicator)*cov
    }

# compute the components that will be needed to calculate the
# expectation of w^2 that will be used to calculate the variance.
# The theoretical form has been derived elsewhere.

mu1_ alpha[team] + scale2
mu2_ alpha[opp] + scale3
rem_ mu1 + (mu1*(sigma[opp]/(sigma[team]+sigma[opp])) +
     mu2*(sigma[team]/(sigma[team]+sigma[opp])))
factor_ sigma[team]/sqrt(sigma[team]+sigma[opp])
scale_ (mu1-mu2)/sqrt(sigma[team]+sigma[opp])

phi_ dnorm(scale)
```

```
PHI_ pnorm(scale)

# calculate the expectations of w, w^2, and then the variance of w
# by subtraction. These are the quantities that need to used in the
# complete data log-likelihood that is to be maximized in the M-step.

zstar_ mu1+win*(phi/PHI)*factor - (1-win)*(phi/(1-PHI))*factor
zstar.sq_ mu1^2 + sigma[team] + win*factor*(phi/PHI)*rem -
        (1-win)*factor*(phi/(1-PHI))*rem

vstar_ zstar.sq - (zstar*zstar)
                        } # end of the MEANS loop

else  { # if only a ranking model is desired

  factor_ sigma[team]/sqrt(sigma[team]+sigma[opp])
  r1_ alpha[team]*(sigma[opp]/(sigma[team]+sigma[opp]))
  r2_ alpha[opp]*(sigma[team]/(sigma[team]+sigma[opp]))
  rem_ alpha[team] + r1 + r2
  scale_ (alpha[team] - alpha[opp])/sqrt(sigma[team] + sigma[opp])

  phi_ dnorm(scale)
  PHI_ pnorm(scale)

  zstar_ alpha[team]+win*(phi/PHI)*factor - (1-win)*(phi/(1-PHI))*factor
  zstar.sq_ (alpha[team])^2+sigma[team]+win*(phi/PHI)*factor*rem -
          (1-win)*(phi/(1-PHI))*factor*rem
  vstar_ zstar.sq - (zstar*zstar)

        } # end of else loop


return(PHI,zstar,vstar)

                } # end of function




# FILE NAME = MSTEP.SPS

# THIS FUNCTION PERFORMS THE MAXIMIZATION OF THE EXPECTED LOG-
# LIKELIHOOD.

# THE VARIABLES THAT WILL BE NEEDED ARE THE FOLLOWING:
```

```
# X = THE DESIGN MATRIX
# TEAM = VECTOR INDICATING THE TEAM INVOLVED IN EACH GAME
# NUMT,NUMC = NUMBER OF TEAMS AND NUMBER OF COVARIATES
# ALPHA, SIGMA = ESTIMATES FROM THE PREVIOUS ITERATION
# ZSTAR, VSTAR = THE VECTORS OF THE EXPECTATION AND VARIANCE OF W
#    THAT ARE OUTPUTTED FROM THE FUNCTION ESTEP.SPS


Mstep.sps_ function(x,team,numt,numc,zstar,vstar,alpha,sigma, MEANS)   {

tempv_ numeric(numt-1)

if (MEANS==F)  {

v_ 1/sigma[team]   # the diagonal elements of the inverse of the
                   # variance matrix

tempm1_ matrix(0, numt+numc-1, numt+numc-1)
xvx_ matrix(0, numt+numc-1, numt+numc-1)
tempm2_ matrix(0, numt+numc-1, 1)
xvz_ matrix(0, numt+numc-1, 1)    # create the matrices that will be
                                  # needed for the matrix addition,
                                  # that will be used to avoid having
                                  # to do matrix multiplication on the
                                  # 2000x2000 matrices.

# this performs the addition that will create the two matrices needed
# for weighted regression, the usual inv(X'VX)*X'VW were V is the
# inverse of the variance matrix of W.

for (i in 1:length(team))  {

  tempm1_ v[i]*(x[i,] %*% t(x[i,]))
  xvx_ tempm1 + xvx
  tempm2_ v[i]*(x[i,])*zstar[i]
  xvz_ tempm2 + xvz
                }

temp10_ solve(xvx) %*% xvz   # estimate the updated value of alpha

alpha.new_ c(0,temp10)
compare_ alpha[2:(numt+numc)]
change_ max(abs((temp10 - compare)/compare))
alpha_ alpha.new     # compare to the estimates at the previous iteration.
```

```
# form the components that will be needed for the estimation of the
# variances. The theoretical form of the estimates are derived
# elsewhere.

pred_ x %*% alpha[2:(numt+numc)]
resid_ zstar - pred
es_ resid*resid

# estimate sigma for each of the teams, note the constraint defines
# the variance for the first team to be unity.

  for (i in 1:(numt-1))  {
    ww_ sum(vstar[team==i+1])
    ses_ sum(es[team==i+1])
    tempv[i]_ (ww + ses)/ngames[i+1]
                  }

sigma.new_ c(1,tempv)

change2_ max(abs((sigma.new - sigma)/sigma))
sigma_ sigma.new  # compare the current estimate to the estimate at
                  # the previous iteration.

test_ max(change, change2)

                  } # end of MEANS loop

else  {  # for simple ranking model

temp10_ numeric(numt-1)

  for (i in 1:(numt-1))  {
   temp10[i]_ mean(zstar[team==i+1])
                  }

  alpha.new_ c(0, temp10)
  compare_ alpha[2:numt]
  change_ max(abs((temp10 - compare)/compare))
  alpha_ alpha.new

  pred_ x %*% alpha[2:numt]
  resid_ zstar - pred
  es_ resid*resid

for (i in 1:(numt-1))  {
```

```
  ww_ sum(vstar[team==i+1])
  ses_ sum(es[team==i+1])
  tempv[i]_ (ww + ses)/ngames[i+1]
                  }

sigma.new_ c(1, tempv)
change2_ max(abs((sigma.new - sigma)/sigma))
sigma_ sigma.new

test_ max(change, change2)
              } # end of the else loop


return(alpha, sigma, test)

                         } # end of function
```

APPENDIX 2 Programs required to estimate parameters using the MCEM Algorithm

```
# FILE NAME : SHELL2.SPS

# THIS IS THE SHELL PROGRAM THAT CALLS THE OTHER FUNCTIONS TO
# CALCULATE THE DESIRED ESTIMATES. THE VARIABLES ARE DEFINED IN
# THE FUNCTIONS AS THEY ARE NEEDED.


EM.rho_function(team,opp,win,indicator,cov,numt,n,ngames,game,numc,
      initial,day,IND=F) {

begin.time_ proc.time()[1:2]

alpha_ initial
rho_ 0.5        # provide initial estimates

# create the design matrix

x.out_createx.s2(team,numt,numc,cov,indicator,IND=T)
x_ x.out$x
des_x.out$des

# NOTE: THE DEFUALT FUNCTION HERE DOES NOT CONSTRUCT A DESIGN
# MATRIX THAT ALLOWS COVARIATES TO BE ESTIMATED SPEARATELY
# FOR EACH TEAM. SUCH AN "X" CAN EASILY BE PROVIDED TO THE
# FUNCTION "EM.RHO." THE GOAL HERE IS CREATE A DESIGN MATRIX FOR
# A SIMPLE MODEL.

gt.out_ create.opp(numt,team,opp,day,n,game)
gt_ gt.out

iterations_ 0
test2_ 100

while (test2 > 10 && iterations < 15)  {  # start of the estimation process. Only 15 iterations
                                          #     are run a time due to memory restrictions on the
                                          #     system used in the Department of Statistics at
                                          #     OSU.


zstar_ rep(0, length(team))
zstarsq_ rep(0, length(team))
zstar2_ rep(0, (length(team)-numt))
zstar.l_ rep(0,length(team))
zstar2.l_ rep(0,(length(team)-numt))  # initialize the vectors needed to combine the sets of 200
                                      # samples that are generated in simulate.sps
```

```
for (s in 1:4) {   # create 800 samples

sim.out_simulate.sps(indicator,team,win,numt,numc,n,alpha,rho,gt,des)
# create simulated seasons and the appropriate simulated
# expected values.

zstar.sim_sim.out$zstar
zstarsq.sim_sim.out$zstarsq
zstar2.sim_sim.out$zstar2
zstarl.sim_ sim.out$zstar.l
zstar2l.sim_ sim.out$zstar2.l # expectations and sample sizes for current group of 200

zstar_ zstar + zstar.sim
zstarsq_ zstarsq + zstarsq.sim
zstar2_ zstar2 + zstar2.sim
zstar.l_ zstar.l + zstarl.sim
zstar2.l_ zstar2.l + zstar2l.sim  # updated vector of expectations and sample sizes for combined
                                   # groups of 200 samples

print("Done")
        } # end of the "s" loop

zstar_ zstar/zstar.l
zstarsq_ zstarsq/zstar.l
zstar2_ zstar2/zstar2.l

rb.out_rhobeta.sps(x,zstar,zstarsq,zstar2,numt,numc,n,rho,alpha,iterations)

# current estimates of the paramters

rho_ c(rb.out$rho)
alpha_ c(0,rb.out$alpha)
rel.change_ rb.out$rel.change
abs.change_ rb.out$abs.change
test_ rb.out$test
test2_ rb.out$test2
iterations_ rb.out$iterations

print(alpha,5)
print(rho,5)
print(rel.change,5)
print(abs.change,5)
print(test,5)
print(iterations,2)   # print current estimates

          } # end of the while loop.
```

```
end.time_proc.time()[1:2]
run.time_end.time - begin.time

return(alpha,rho,iterations,run.time)

                } # end of the function




# FILE NAME : CREATEX.SPS

# THIS PROGRAM CREATES THE DESIGN MATRIX BASED ON THE TEAMS
# INVOLVED AND ANY COVARIATES THAT ARE OBSERVED AT EACH GAME.

# THE FOLLOWING VECTORS MUST BE READ INTO THE FUNCTION:

# TEAM = VECTOR OF TEAM LABELS FOR EACH GAME
# NUMT, NUMC = NUMBER OF TEAMS AND COVARIATES IN THE DATA
# COV = VECTOR OF COVARIATES THAT ARE OBSERVED
# IND = INDICATOR VARIABLE EQUAL TO "T" IF THE COVARIATE IS THE
#       HOME FIELD ADVANTAGE
# INDICATOR = HOME FIELD ADVANTAGE

createx.s2_function(team,numt,numc,cov,indicator,IND) {

x_ matrix(0,length(team),numt-1)

for (i in 1:(numt-1)) {
  for (j in 1:length(team)) {
   if (team[j]==i+1) x[j,i]_1
                    } # end of the "j" loop
                } # end of the "i" loop

# DESIGN MATRIX FOR A SIMPLE RANKING MODEL


if (numc==1) {

des_cov*indicator
if (IND==T) cov_rep(1,length(team))
x_ cbind(x, des)
          } # end of the "if" loop

if (numc > 1) {

des_ cov
for (i in 1:numc) des[,i]_ indicator*des[,i]
x_ cbind(x, des)
```

```
if (IND==T)  cov_cbind(rep(1,length(team)), cov[,2:numc])
            }  # end of the "if" loop


return(x,des)

# note that this will not provide a design matrix if
# parameters are wanted for each individual team for a
# particular covariate.


                        }  # end of the function




# FILE NAME : CREATEOP.SPS

# THIS FUNCTION WILL IDENTIFY WHICH SEQUENTIAL GAME IS IDENTIFIED
# WITH THE OPPONENT.  IN OTHER WORDS IT WILL DETERMINE IF TEAM 2'S
# SECOND GAME IS PLAYED AGAINST TEAM 3, WHO IS PLAYING THEIR THIRD
# GAME, ETC.

# THE FOLLOWING VARIABLES NEED TO BE PASSED INTO THE FUNCTION:

#   NUMT = THE NUMBER OF TEAMS IN THE LEAGUE
#   TEAM, OPP = THE VECTORS IDENTIFYING THE TEAM AND OPP FOR EVERY
#                  GAME
#   DAY  = THIS IDENTIFIES THE DAY OF THE SEASON THAT THE PARTICULAR
#                 GAME WAS PLAYED.
#    N =  THE NUMBER OF GAMES IN A SEASON.  FOR NOW WE ASSUME THAT
#         EACH TEAM PLAYS THE SAME NUMBER OF GAMES
#   GAME = A VECTOR FROM 1 TO THE LENGTH OF TEAM.



create.opp_ function(numt,team,opp,day,n,game)    {

gt.temp_ numeric(n)

gt_ 99

for (j in 1:numt) { # examine each of the j team individually

 opp.temp_ opp[team==j]
 day.temp_ day[team==j]

for (i in 1:n)   {
```

```
 gt.temp[i]_ game[team==opp.temp[i] & day==day.temp[i]]
            }  # end of the "i" loop

# locate the performance value of the opponent for team j
# who played on the same day.  Note that day may have to
# be adjusted prior to running the function to account for
# such things as doubleheaders in baseball.

gt_ c(gt,gt.temp)

              }  # end of the "j" loop

gt_ gt[2:length(gt)]

return(gt)
                }  # end of function




# FILE NAME: CREATEW.SPS


# CREATE THE PERFORMANCE VARIABLES FOR THE VECTOR "TEAM" FOR 200
# SAMPLES

# NEED TO PASS THE FOLLOWING VECTORS INTO THE FUNCTION:

#    NGAMES = VECTOR OF THE NUMBER OF GAMES EACH TEAM PLAYS.
#         ASSUME FOR NOW THAT EACH TEAM PLAYS THE SAME
#         NUMBER OF GAMES.
#    RHO = CURRENT ESTIMATE OF THE AUTOCORRELATION
#    ALPHA = CURRENT ESTIMATE OF THE STRENGTH PARAMETERS
#    NUMT = THE NUMBER OF TEAMS IN THE LEAGUE
#    N = THE NUMBER OF GAMES EACH TEAM PLAYS - ASSUMED TO BE
#              CONSTANT.
#    GT = THE VECTOR INDICATING THE GAME OF THE SEASON THE OPPT IS
#             PLAYING.
#    INDICATOR = VECTOR IDENTIFYING THE HOME TEAM IN EACH GAME
#    DES = THE PORTION OF THE DESIGN MATRIX THAT CORRESPONDS TO THE
#              COVARIATES.  I.E. NOT INCLUDING THE INDICATOR VARIABLES
#              IDENTIFYING THE TEAMS INVOLVED IN EACH GAME.


createw.sps_function(indicator,n,rho,alpha,numt,numc,gt,des)  {

cov_rep(1,n)
cov_cbind(cov)  # Assumes just a simple ranking model
wteam_99
```

```
for (j in 1:numt)  {

 if (numc==0) coeff_alpha[j]
 else  {
  cov.t_cbind(cov,des[team==j])
  coeff_ c(alpha[j],alpha[numt+numc])
     }  # This allows a single covariate to be included in the model.

 x_arima.sim(n,model=list(order=c(1,0,0),ar=rho),xreg=cov.t,
           reg.coef=coeff)


 wteam_c(wteam,x)
# creates the simulated season for each of the j teams.
                  }  #  end of "j" loop.

wteam_ wteam[2:length(wteam)]
wopp_ wteam[gt]  # creates the vector of performance variables that
                    # corresponds to the opponent for the ith game against
            # each team.

return(wteam,wopp)

                       } # end of the function




# FILE NAME : SIMULATE.SPS

# THIS IS THE HEART OF THE PROGRAM.  THIS IS THE SIMULATION
# THAT CREATES THE NEEDED EXPECTATIONS.

# THE FOLLOWING VECTORS ARE NEEDED FOR THE PROGRAM:

#   INDICATOR = VECTOR SPECIFYING THE HOME TEAM IN EACH GAME
#   TEAM = VECTOR SPECIFYING THE TEAM INVOLVED IN EACH GAME.
#   WIN = VECTOR OF THE OBSERVED WIN-LOSS OUTCOMES FOR EACH GAME
#    NUMT,NUMC = NUMBER OF TEAMS AND NUMBER OF COVARIATES
#    N    = WE ASSUME FOR NOW THAT EACH TEAM PLAYS THE
#          SAME NUMBER OF GAMES.
#    ALPHA, RHO = THE CURRENT PARAMETER ESTIMATES
#   GT = THE VECTOR INDICATING WHICH GAME OF THE SEASON THE OPPT IS
#              PLAYING IN.
#   DES = THE PORTION OF THE DESIGN MATRIX THAT CORRESPONDS TO THE
#                COVARIATES, I.E. THE INDICATOR VARIABLES IDENTIFYING THE
```

```
#                TEAMS INVOLVED IN EACH GAME ARE NOT INCLUDED.


simulate.sps_function(indicator,team,win,numt,numc,n,alpha,rho,gt,des) {

# begin.time_ proc.time()[1:2]


wteam_matrix(0,length(team),200)
wopp_matrix(0,length(team),200)

zstar.t_ numeric(n)
zstarsq.t_ numeric(n)
zstar2.t_ numeric(n-1)
zstarl.t_ numeric(n)
zstar2l.t_ numeric(n-1)


zstar_ 99
zstarsq_ 99
zstar2_ 99
zstar.l_ 99
zstar2.l_ 99  # initialize needed vectors and matrices.


# create the 200 simulated seasons:
for (i in 1:200) {

w.out_createw.sps(indicator,n,rho,alpha,numt,numc,gt,des)
wteam[,i]_ w.out$wteam
wopp[,i]_ w.out$wopp

            } # end of the "i" loop.


# examine the data for each individual team:
for (j in 1:numt)  {

wtemp_ wteam[(((j-1)*n)+1):(j*n),]
otemp_ wopp[(((j-1)*n)+1):(j*n),]  # selects the portions of these
temp.y_ win[(((j-1)*n)+1):(j*n)]   # two matrices specific to team
                                   # j.

# calculate the simulated expectations and the number of samples, based on the OBSERVED data
# for each w[i].  The "if" portion identifies the pattern of the
# observed data.

for (i in 2:(n-2))  {
```

```
if (temp.y[i-1]==1 & temp.y[i]==1 & temp.y[i+1]==1 & temp.y[i+2]==1) {
zstar.t[i]_sum(wtemp[i,][wtemp[i-1,]>otemp[i-1,] & wtemp[i,]>otemp[i,]
        & wtemp[i+1,]>otemp[i+1,]])
zstarsq.t[i]_sum((wtemp[i,][wtemp[i-1,]>otemp[i-1,] & wtemp[i,]>otemp[i,]
        & wtemp[i+1,]>otemp[i+1,]])*(wtemp[i,][wtemp[i-1,]>otemp[i-1,] & wtemp[i,]>otemp[i,]
        & wtemp[i+1,]>otemp[i+1,]]))
zstar2.t[i]_sum(wtemp[i+1,][wtemp[i-1,]>otemp[i-1,] & wtemp[i,]>otemp[i,]
        & wtemp[i+1,]>otemp[i+1,] & wtemp[i+2,]>otemp[i+2,]]*
    wtemp[i,][wtemp[i-1,]>otemp[i-1,] & wtemp[i,]>otemp[i,]
        & wtemp[i+1,]>otemp[i+1,] & wtemp[i+2,]>otemp[i+2,]])
zstarl.t[i]_length(wtemp[i,][wtemp[i-1,]>otemp[i-1,] & wtemp[i,]>otemp[i,]
        & wtemp[i+1,]>otemp[i+1,]])
zstar2l.t[i]_length(wtemp[i+1,][wtemp[i-1,]>otemp[i-1,] & wtemp[i,]>otemp[i,]
        & wtemp[i+1,]>otemp[i+1,] & wtemp[i+2,]>otemp[i+2,]]*
    wtemp[i,][wtemp[i-1,]>otemp[i-1,] & wtemp[i,]>otemp[i,]
        & wtemp[i+1,]>otemp[i+1,] & wtemp[i+2,]>otemp[i+2,]])
                        }

else if (temp.y[i-1]==1 & temp.y[i]==1 & temp.y[i+1]==1 & temp.y[i+2]==0) {
zstar.t[i]_sum(wtemp[i,][wtemp[i-1,]>otemp[i-1,] & wtemp[i,]>otemp[i,]
        & wtemp[i+1,]>otemp[i+1,]])
zstarsq.t[i]_sum((wtemp[i,][wtemp[i-1,]>otemp[i-1,] & wtemp[i,]>otemp[i,]
        & wtemp[i+1,]>otemp[i+1,]])*(wtemp[i,][wtemp[i-1,]>otemp[i-1,] & wtemp[i,]>otemp[i,]
        & wtemp[i+1,]>otemp[i+1,]]))
zstar2.t[i]_sum(wtemp[i+1,][wtemp[i-1,]>otemp[i-1,] & wtemp[i,]>otemp[i,]
        & wtemp[i+1,]>otemp[i+1,] & wtemp[i+2,]<otemp[i+2,]]*
    wtemp[i,][wtemp[i-1,]>otemp[i-1,] & wtemp[i,]>otemp[i,]
        & wtemp[i+1,]>otemp[i+1,] & wtemp[i+2,]<otemp[i+2,]])
zstarl.t[i]_length(wtemp[i,][wtemp[i-1,]>otemp[i-1,] & wtemp[i,]>otemp[i,]
        & wtemp[i+1,]>otemp[i+1,]])
zstar2l.t[i]_length(wtemp[i+1,][wtemp[i-1,]>otemp[i-1,] & wtemp[i,]>otemp[i,]
        & wtemp[i+1,]>otemp[i+1,] & wtemp[i+2,]<otemp[i+2,]]*
    wtemp[i,][wtemp[i-1,]>otemp[i-1,] & wtemp[i,]>otemp[i,]
        & wtemp[i+1,]>otemp[i+1,] & wtemp[i+2,]<otemp[i+2,]])
                        }
else if (temp.y[i-1]==1 & temp.y[i]==1 & temp.y[i+1]==0 & temp.y[i+2]==1) {
zstar.t[i]_sum(wtemp[i,][wtemp[i-1,]>otemp[i-1,] & wtemp[i,]>otemp[i,]
        & wtemp[i+1,]<otemp[i+1,]])
zstarsq.t[i]_sum((wtemp[i,][wtemp[i-1,]>otemp[i-1,] & wtemp[i,]>otemp[i,]
        & wtemp[i+1,]<otemp[i+1,]])*(wtemp[i,][wtemp[i-1,]>otemp[i-1,] & wtemp[i,]>otemp[i,]
        & wtemp[i+1,]<otemp[i+1,]]))
zstar2.t[i]_sum(wtemp[i+1,][wtemp[i-1,]>otemp[i-1,] & wtemp[i,]>otemp[i,]
        & wtemp[i+1,]<otemp[i+1,] & wtemp[i+2,]>otemp[i+2,]]*
    wtemp[i,][wtemp[i-1,]>otemp[i-1,] & wtemp[i,]>otemp[i,]
        & wtemp[i+1,]<otemp[i+1,] & wtemp[i+2,]>otemp[i+2,]])
zstarl.t[i]_length(wtemp[i,][wtemp[i-1,]>otemp[i-1,] & wtemp[i,]>otemp[i,]
        & wtemp[i+1,]<otemp[i+1,]])
zstar2l.t[i]_length(wtemp[i+1,][wtemp[i-1,]>otemp[i-1,] & wtemp[i,]>otemp[i,]
```

```
            & wtemp[i+1,]<otemp[i+1,] & wtemp[i+2,]>otemp[i+2,]]*
        wtemp[i,][wtemp[i-1,]>otemp[i-1,] & wtemp[i,]>otemp[i,]
            & wtemp[i+1,]<otemp[i+1,] & wtemp[i+2,]>otemp[i+2,]])
                                            }

else if (temp.y[i-1]==1 & temp.y[i]==1 & temp.y[i+1]==0 & temp.y[i+2]==0) {
zstar.t[i]_sum(wtemp[i,][wtemp[i-1,]>otemp[i-1,] & wtemp[i,]>otemp[i,]
            & wtemp[i+1,]<otemp[i+1,]])
zstarsq.t[i]_sum((wtemp[i,][wtemp[i-1,]>otemp[i-1,] & wtemp[i,]>otemp[i,]
          & wtemp[i+1,]<otemp[i+1,]])*(wtemp[i,][wtemp[i-1,]>otemp[i-1,] & wtemp[i,]>otemp[i,]
          & wtemp[i+1,]<otemp[i+1,]]))
zstar2.t[i]_sum(wtemp[i+1,][wtemp[i-1,]>otemp[i-1,] & wtemp[i,]>otemp[i,]
            & wtemp[i+1,]<otemp[i+1,] & wtemp[i+2,]<otemp[i+2,]]*
        wtemp[i,][wtemp[i-1,]>otemp[i-1,] & wtemp[i,]>otemp[i,]
            & wtemp[i+1,]<otemp[i+1,] & wtemp[i+2,]<otemp[i+2,]])
zstarl.t[i]_length(wtemp[i,][wtemp[i-1,]>otemp[i-1,] & wtemp[i,]>otemp[i,]
            & wtemp[i+1,]<otemp[i+1,]])
zstar2l.t[i]_length(wtemp[i+1,][wtemp[i-1,]>otemp[i-1,] & wtemp[i,]>otemp[i,]
            & wtemp[i+1,]<otemp[i+1,] & wtemp[i+2,]<otemp[i+2,]]*
        wtemp[i,][wtemp[i-1,]>otemp[i-1,] & wtemp[i,]>otemp[i,]
            & wtemp[i+1,]<otemp[i+1,] & wtemp[i+2,]<otemp[i+2,]])
                                            }

else if (temp.y[i-1]==1 & temp.y[i]==0 & temp.y[i+1]==1 & temp.y[i+2]==1) {
zstar.t[i]_sum(wtemp[i,][wtemp[i-1,]>otemp[i-1,] & wtemp[i,]<otemp[i,]
            & wtemp[i+1,]>otemp[i+1,]])
zstarsq.t[i]_sum((wtemp[i,][wtemp[i-1,]>otemp[i-1,] & wtemp[i,]<otemp[i,]
          & wtemp[i+1,]>otemp[i+1,]])*(wtemp[i,][wtemp[i-1,]>otemp[i-1,] & wtemp[i,]<otemp[i,]
          & wtemp[i+1,]>otemp[i+1,]]))
zstar2.t[i]_sum(wtemp[i+1,][wtemp[i-1,]>otemp[i-1,] & wtemp[i,]<otemp[i,]
            & wtemp[i+1,]>otemp[i+1,] & wtemp[i+2,]>otemp[i+2,]]*
        wtemp[i,][wtemp[i-1,]>otemp[i-1,] & wtemp[i,]<otemp[i,]
            & wtemp[i+1,]>otemp[i+1,] & wtemp[i+2,]>otemp[i+2,]])
zstarl.t[i]_length(wtemp[i,][wtemp[i-1,]>otemp[i-1,] & wtemp[i,]<otemp[i,]
            & wtemp[i+1,]>otemp[i+1,]])
zstar2l.t[i]_length(wtemp[i+1,][wtemp[i-1,]>otemp[i-1,] & wtemp[i,]<otemp[i,]
            & wtemp[i+1,]>otemp[i+1,] & wtemp[i+2,]>otemp[i+2,]]*
        wtemp[i,][wtemp[i-1,]>otemp[i-1,] & wtemp[i,]<otemp[i,]
            & wtemp[i+1,]>otemp[i+1,] & wtemp[i+2,]>otemp[i+2,]])
                                            }

else if (temp.y[i-1]==1 & temp.y[i]==0 & temp.y[i+1]==1 & temp.y[i+2]==0) {
zstar.t[i]_sum(wtemp[i,][wtemp[i-1,]>otemp[i-1,] & wtemp[i,]<otemp[i,]
            & wtemp[i+1,]>otemp[i+1,]])
zstarsq.t[i]_sum((wtemp[i,][wtemp[i-1,]>otemp[i-1,] & wtemp[i,]<otemp[i,]
          & wtemp[i+1,]>otemp[i+1,]])*(wtemp[i,][wtemp[i-1,]>otemp[i-1,] & wtemp[i,]<otemp[i,]
          & wtemp[i+1,]>otemp[i+1,]]))
zstar2.t[i]_sum(wtemp[i+1,][wtemp[i-1,]>otemp[i-1,] & wtemp[i,]<otemp[i,]
            & wtemp[i+1,]>otemp[i+1,] & wtemp[i+2,]<otemp[i+2,]]*
```

```
    wtemp[i,][wtemp[i-1,]>otemp[i-1,] & wtemp[i,]<otemp[i,]
        & wtemp[i+1,]>otemp[i+1,] & wtemp[i+2,]<otemp[i+2,]])
zstarl.t[i]_length(wtemp[i,][wtemp[i-1,]>otemp[i-1,] & wtemp[i,]<otemp[i,]
        & wtemp[i+1,]>otemp[i+1,]])
zstar2l.t[i]_length(wtemp[i+1,][wtemp[i-1,]>otemp[i-1,] & wtemp[i,]<otemp[i,]
        & wtemp[i+1,]>otemp[i+1,] & wtemp[i+2,]<otemp[i+2,]]*
    wtemp[i,][wtemp[i-1,]>otemp[i-1,] & wtemp[i,]<otemp[i,]
        & wtemp[i+1,]>otemp[i+1,] & wtemp[i+2,]<otemp[i+2,]])
                                        }

else if (temp.y[i-1]==1 & temp.y[i]==0 & temp.y[i+1]==0 & temp.y[i+2]==1) {
zstar.t[i]_sum(wtemp[i,][wtemp[i-1,]>otemp[i-1,] & wtemp[i,]<otemp[i,]
        & wtemp[i+1,]<otemp[i+1,]])
zstarsq.t[i]_sum((wtemp[i,][wtemp[i-1,]>otemp[i-1,] & wtemp[i,]<otemp[i,]
       & wtemp[i+1,]<otemp[i+1,]])*(wtemp[i,][wtemp[i-1,]>otemp[i-1,] & wtemp[i,]<otemp[i,]
       & wtemp[i+1,]<otemp[i+1,]]))
zstar2.t[i]_sum(wtemp[i+1,][wtemp[i-1,]>otemp[i-1,] & wtemp[i,]<otemp[i,]
        & wtemp[i+1,]<otemp[i+1,] & wtemp[i+2,]>otemp[i+2,]]*
    wtemp[i,][wtemp[i-1,]>otemp[i-1,] & wtemp[i,]<otemp[i,]
        & wtemp[i+1,]<otemp[i+1,] & wtemp[i+2,]>otemp[i+2,]])
zstarl.t[i]_length(wtemp[i,][wtemp[i-1,]>otemp[i-1,] & wtemp[i,]<otemp[i,]
        & wtemp[i+1,]<otemp[i+1,]])
zstar2l.t[i]_length(wtemp[i+1,][wtemp[i-1,]>otemp[i-1,] & wtemp[i,]<otemp[i,]
        & wtemp[i+1,]<otemp[i+1,] & wtemp[i+2,]>otemp[i+2,]]*
    wtemp[i,][wtemp[i-1,]>otemp[i-1,] & wtemp[i,]<otemp[i,]
        & wtemp[i+1,]<otemp[i+1,] & wtemp[i+2,]>otemp[i+2,]])
                                        }

else if (temp.y[i-1]==1 & temp.y[i]==0 & temp.y[i+1]==0 & temp.y[i+2]==0) {
zstar.t[i]_sum(wtemp[i,][wtemp[i-1,]>otemp[i-1,] & wtemp[i,]<otemp[i,]
        & wtemp[i+1,]<otemp[i+1,]])
zstarsq.t[i]_sum((wtemp[i,][wtemp[i-1,]>otemp[i-1,] & wtemp[i,]<otemp[i,]
       & wtemp[i+1,]<otemp[i+1,]])*(wtemp[i,][wtemp[i-1,]>otemp[i-1,] & wtemp[i,]<otemp[i,]
       & wtemp[i+1,]<otemp[i+1,]]))
zstar2.t[i]_sum(wtemp[i+1,][wtemp[i-1,]>otemp[i-1,] & wtemp[i,]<otemp[i,]
        & wtemp[i+1,]<otemp[i+1,] & wtemp[i+2,]<otemp[i+2,]]*
    wtemp[i,][wtemp[i-1,]>otemp[i-1,] & wtemp[i,]<otemp[i,]
        & wtemp[i+1,]<otemp[i+1,] & wtemp[i+2,]<otemp[i+2,]])
zstarl.t[i]_length(wtemp[i,][wtemp[i-1,]>otemp[i-1,] & wtemp[i,]<otemp[i,]
        & wtemp[i+1,]<otemp[i+1,]])
zstar2l.t[i]_length(wtemp[i+1,][wtemp[i-1,]>otemp[i-1,] & wtemp[i,]<otemp[i,]
        & wtemp[i+1,]<otemp[i+1,] & wtemp[i+2,]<otemp[i+2,]]*
    wtemp[i,][wtemp[i-1,]>otemp[i-1,] & wtemp[i,]<otemp[i,]
        & wtemp[i+1,]<otemp[i+1,] & wtemp[i+2,]<otemp[i+2,]])
                                        }
else if (temp.y[i-1]==0 & temp.y[i]==1 & temp.y[i+1]==1 & temp.y[i+2]==1) {
zstar.t[i]_sum(wtemp[i,][wtemp[i-1,]<otemp[i-1,] & wtemp[i,]>otemp[i,]
        & wtemp[i+1,]>otemp[i+1,]])
zstarsq.t[i]_sum((wtemp[i,][wtemp[i-1,]<otemp[i-1,] & wtemp[i,]>otemp[i,]
```

```
          & wtemp[i+1,]>otemp[i+1,]])*(wtemp[i,][wtemp[i-1,]<otemp[i-1,] & wtemp[i,]>otemp[i,]
          & wtemp[i+1,]>otemp[i+1,]]))
zstar2.t[i]_sum(wtemp[i+1,][wtemp[i-1,]<otemp[i-1,] & wtemp[i,]>otemp[i,]
           & wtemp[i+1,]>otemp[i+1,] & wtemp[i+2,]>otemp[i+2,]]*
      wtemp[i,][wtemp[i-1,]<otemp[i-1,] & wtemp[i,]>otemp[i,]
           & wtemp[i+1,]>otemp[i+1,] & wtemp[i+2,]>otemp[i+2,]])
zstarl.t[i]_length(wtemp[i,][wtemp[i-1,]<otemp[i-1,] & wtemp[i,]>otemp[i,]
           & wtemp[i+1,]>otemp[i+1,]])
zstar2l.t[i]_length(wtemp[i+1,][wtemp[i-1,]<otemp[i-1,] & wtemp[i,]>otemp[i,]
           & wtemp[i+1,]>otemp[i+1,] & wtemp[i+2,]>otemp[i+2,]]*
      wtemp[i,][wtemp[i-1,]<otemp[i-1,] & wtemp[i,]>otemp[i,]
           & wtemp[i+1,]>otemp[i+1,] & wtemp[i+2,]>otemp[i+2,]])
                                                      }

else if (temp.y[i-1]==0 & temp.y[i]==1 & temp.y[i+1]==1 & temp.y[i+2]==0) {
zstar.t[i]_sum(wtemp[i,][wtemp[i-1,]<otemp[i-1,] & wtemp[i,]>otemp[i,]
           & wtemp[i+1,]>otemp[i+1,]])
zstarsq.t[i]_sum((wtemp[i,][wtemp[i-1,]<otemp[i-1,] & wtemp[i,]>otemp[i,]
          & wtemp[i+1,]>otemp[i+1,]])*(wtemp[i,][wtemp[i-1,]<otemp[i-1,] & wtemp[i,]>otemp[i,]
          & wtemp[i+1,]>otemp[i+1,]]))
zstar2.t[i]_sum(wtemp[i+1,][wtemp[i-1,]<otemp[i-1,] & wtemp[i,]>otemp[i,]
           & wtemp[i+1,]>otemp[i+1,] & wtemp[i+2,]<otemp[i+2,]]*
      wtemp[i,][wtemp[i-1,]<otemp[i-1,] & wtemp[i,]>otemp[i,]
           & wtemp[i+1,]>otemp[i+1,] & wtemp[i+2,]<otemp[i+2,]])
zstarl.t[i]_length(wtemp[i,][wtemp[i-1,]<otemp[i-1,] & wtemp[i,]>otemp[i,]
           & wtemp[i+1,]>otemp[i+1,]])
zstar2l.t[i]_length(wtemp[i+1,][wtemp[i-1,]<otemp[i-1,] & wtemp[i,]>otemp[i,]
           & wtemp[i+1,]>otemp[i+1,] & wtemp[i+2,]<otemp[i+2,]]*
      wtemp[i,][wtemp[i-1,]<otemp[i-1,] & wtemp[i,]>otemp[i,]
           & wtemp[i+1,]>otemp[i+1,] & wtemp[i+2,]<otemp[i+2,]])
                                                      }

else if (temp.y[i-1]==0 & temp.y[i]==1 & temp.y[i+1]==0 & temp.y[i+2]==1) {
zstar.t[i]_sum(wtemp[i,][wtemp[i-1,]<otemp[i-1,] & wtemp[i,]>otemp[i,]
           & wtemp[i+1,]<otemp[i+1,]])
zstarsq.t[i]_sum((wtemp[i,][wtemp[i-1,]<otemp[i-1,] & wtemp[i,]>otemp[i,]
          & wtemp[i+1,]<otemp[i+1,]])*(wtemp[i,][wtemp[i-1,]<otemp[i-1,] & wtemp[i,]>otemp[i,]
          & wtemp[i+1,]<otemp[i+1,]]))
zstar2.t[i]_sum(wtemp[i+1,][wtemp[i-1,]<otemp[i-1,] & wtemp[i,]>otemp[i,]
           & wtemp[i+1,]<otemp[i+1,] & wtemp[i+2,]>otemp[i+2,]]*
      wtemp[i,][wtemp[i-1,]<otemp[i-1,] & wtemp[i,]>otemp[i,]
           & wtemp[i+1,]<otemp[i+1,] & wtemp[i+2,]>otemp[i+2,]])
zstarl.t[i]_length(wtemp[i,][wtemp[i-1,]<otemp[i-1,] & wtemp[i,]>otemp[i,]
           & wtemp[i+1,]<otemp[i+1,]])
zstar2l.t[i]_length(wtemp[i+1,][wtemp[i-1,]<otemp[i-1,] & wtemp[i,]>otemp[i,]
           & wtemp[i+1,]<otemp[i+1,] & wtemp[i+2,]>otemp[i+2,]]*
      wtemp[i,][wtemp[i-1,]<otemp[i-1,] & wtemp[i,]>otemp[i,]
           & wtemp[i+1,]<otemp[i+1,] & wtemp[i+2,]>otemp[i+2,]])
                                                      }
```

```
else if (temp.y[i-1]==0 & temp.y[i]==1 & temp.y[i+1]==0 & temp.y[i+2]==0) {
zstar.t[i]_sum(wtemp[i,][wtemp[i-1,]<otemp[i-1,] & wtemp[i,]>otemp[i,]
      & wtemp[i+1,]<otemp[i+1,]])
zstarsq.t[i]_sum((wtemp[i,][wtemp[i-1,]<otemp[i-1,] & wtemp[i,]>otemp[i,]
      & wtemp[i+1,]<otemp[i+1,]])*(wtemp[i,][wtemp[i-1,]<otemp[i-1,] & wtemp[i,]>otemp[i,]
      & wtemp[i+1,]<otemp[i+1,]]))
zstar2.t[i]_sum(wtemp[i+1,][wtemp[i-1,]<otemp[i-1,] & wtemp[i,]>otemp[i,]
      & wtemp[i+1,]<otemp[i+1,] & wtemp[i+2,]<otemp[i+2,]]*
    wtemp[i,][wtemp[i-1,]<otemp[i-1,] & wtemp[i,]>otemp[i,]
      & wtemp[i+1,]<otemp[i+1,] & wtemp[i+2,]<otemp[i+2,]])
zstarl.t[i]_length(wtemp[i,][wtemp[i-1,]<otemp[i-1,] & wtemp[i,]>otemp[i,]
      & wtemp[i+1,]<otemp[i+1,]])
zstar2l.t[i]_length(wtemp[i+1,][wtemp[i-1,]<otemp[i-1,] & wtemp[i,]>otemp[i,]
      & wtemp[i+1,]<otemp[i+1,] & wtemp[i+2,]<otemp[i+2,]]*
    wtemp[i,][wtemp[i-1,]<otemp[i-1,] & wtemp[i,]>otemp[i,]
      & wtemp[i+1,]<otemp[i+1,] & wtemp[i+2,]<otemp[i+2,]])
                                        }
else if (temp.y[i-1]==0 & temp.y[i]==0 & temp.y[i+1]==1 & temp.y[i+2]==1) {
zstar.t[i]_sum(wtemp[i,][wtemp[i-1,]<otemp[i-1,] & wtemp[i,]<otemp[i,]
      & wtemp[i+1,]>otemp[i+1,]])
zstarsq.t[i]_sum((wtemp[i,][wtemp[i-1,]<otemp[i-1,] & wtemp[i,]<otemp[i,]
      & wtemp[i+1,]>otemp[i+1,]])*(wtemp[i,][wtemp[i-1,]<otemp[i-1,] & wtemp[i,]<otemp[i,]
      & wtemp[i+1,]>otemp[i+1,]]))
zstar2.t[i]_sum(wtemp[i+1,][wtemp[i-1,]<otemp[i-1,] & wtemp[i,]<otemp[i,]
      & wtemp[i+1,]>otemp[i+1,] & wtemp[i+2,]>otemp[i+2,]]*
    wtemp[i,][wtemp[i-1,]<otemp[i-1,] & wtemp[i,]<otemp[i,]
      & wtemp[i+1,]>otemp[i+1,] & wtemp[i+2,]>otemp[i+2,]])
zstarl.t[i]_length(wtemp[i,][wtemp[i-1,]<otemp[i-1,] & wtemp[i,]<otemp[i,]
      & wtemp[i+1,]>otemp[i+1,]])
zstar2l.t[i]_length(wtemp[i+1,][wtemp[i-1,]<otemp[i-1,] & wtemp[i,]<otemp[i,]
      & wtemp[i+1,]>otemp[i+1,] & wtemp[i+2,]>otemp[i+2,]]*
    wtemp[i,][wtemp[i-1,]<otemp[i-1,] & wtemp[i,]<otemp[i,]
      & wtemp[i+1,]>otemp[i+1,] & wtemp[i+2,]>otemp[i+2,]])
                                        }

else if (temp.y[i-1]==0 & temp.y[i]==0 & temp.y[i+1]==1 & temp.y[i+2]==0) {
zstar.t[i]_sum(wtemp[i,][wtemp[i-1,]<otemp[i-1,] & wtemp[i,]<otemp[i,]
      & wtemp[i+1,]>otemp[i+1,]])
zstarsq.t[i]_sum((wtemp[i,][wtemp[i-1,]<otemp[i-1,] & wtemp[i,]<otemp[i,]
      & wtemp[i+1,]>otemp[i+1,]])*(wtemp[i,][wtemp[i-1,]<otemp[i-1,] & wtemp[i,]<otemp[i,]
      & wtemp[i+1,]>otemp[i+1,]]))
zstar2.t[i]_sum(wtemp[i+1,][wtemp[i-1,]<otemp[i-1,] & wtemp[i,]<otemp[i,]
      & wtemp[i+1,]>otemp[i+1,] & wtemp[i+2,]<otemp[i+2,]]*
    wtemp[i,][wtemp[i-1,]<otemp[i-1,] & wtemp[i,]<otemp[i,]
      & wtemp[i+1,]>otemp[i+1,] & wtemp[i+2,]<otemp[i+2,]])
zstarl.t[i]_length(wtemp[i,][wtemp[i-1,]<otemp[i-1,] & wtemp[i,]<otemp[i,]
      & wtemp[i+1,]>otemp[i+1,]])
zstar2l.t[i]_length(wtemp[i+1,][wtemp[i-1,]<otemp[i-1,] & wtemp[i,]<otemp[i,]
```

```
        & wtemp[i+1,]>otemp[i+1,] & wtemp[i+2,]<otemp[i+2,]]*
    wtemp[i,][wtemp[i-1,]<otemp[i-1,] & wtemp[i,]<otemp[i,]
        & wtemp[i+1,]>otemp[i+1,] & wtemp[i+2,]<otemp[i+2,]])
                                        }

else if (temp.y[i-1]==0 & temp.y[i]==0 & temp.y[i+1]==0 & temp.y[i+2]==1) {
zstar.t[i]_sum(wtemp[i,][wtemp[i-1,]<otemp[i-1,] & wtemp[i,]<otemp[i,]
        & wtemp[i+1,]<otemp[i+1,]])
zstarsq.t[i]_sum((wtemp[i,][wtemp[i-1,]<otemp[i-1,] & wtemp[i,]<otemp[i,]
        & wtemp[i+1,]<otemp[i+1,]])*(wtemp[i,][wtemp[i-1,]<otemp[i-1,] & wtemp[i,]<otemp[i,]
        & wtemp[i+1,]<otemp[i+1,]]))
zstar2.t[i]_sum(wtemp[i+1,][wtemp[i-1,]<otemp[i-1,] & wtemp[i,]<otemp[i,]
        & wtemp[i+1,]<otemp[i+1,] & wtemp[i+2,]>otemp[i+2,]]*
    wtemp[i,][wtemp[i-1,]<otemp[i-1,] & wtemp[i,]<otemp[i,]
        & wtemp[i+1,]<otemp[i+1,] & wtemp[i+2,]>otemp[i+2,]])
zstarl.t[i]_length(wtemp[i,][wtemp[i-1,]<otemp[i-1,] & wtemp[i,]<otemp[i,]
        & wtemp[i+1,]<otemp[i+1,]])
zstar2l.t[i]_length(wtemp[i+1,][wtemp[i-1,]<otemp[i-1,] & wtemp[i,]<otemp[i,]
        & wtemp[i+1,]<otemp[i+1,] & wtemp[i+2,]>otemp[i+2,]]*
    wtemp[i,][wtemp[i-1,]<otemp[i-1,] & wtemp[i,]<otemp[i,]
        & wtemp[i+1,]<otemp[i+1,] & wtemp[i+2,]>otemp[i+2,]])
                                        }

else {
zstar.t[i]_sum(wtemp[i,][wtemp[i-1,]<otemp[i-1,] & wtemp[i,]<otemp[i,]
        & wtemp[i+1,]<otemp[i+1,]])
zstarsq.t[i]_sum((wtemp[i,][wtemp[i-1,]<otemp[i-1,] & wtemp[i,]<otemp[i,]
        & wtemp[i+1,]<otemp[i+1,]])*(wtemp[i,][wtemp[i-1,]<otemp[i-1,] & wtemp[i,]<otemp[i,]
        & wtemp[i+1,]<otemp[i+1,]]))
zstar2.t[i]_sum(wtemp[i+1,][wtemp[i-1,]<otemp[i-1,] & wtemp[i,]<otemp[i,]
      & wtemp[i+1,]<otemp[i+1,] & wtemp[i+2,]<otemp[i+2,]]*
    wtemp[i,][wtemp[i-1,]<otemp[i-1,] & wtemp[i,]<otemp[i,]
      & wtemp[i+1,]<otemp[i+1,] & wtemp[i+2,]<otemp[i+2,]])
zstarl.t[i]_length(wtemp[i,][wtemp[i-1,]<otemp[i-1,] & wtemp[i,]<otemp[i,]
        & wtemp[i+1,]<otemp[i+1,]])
zstar2l.t[i]_length(wtemp[i+1,][wtemp[i-1,]<otemp[i-1,] & wtemp[i,]<otemp[i,]
      & wtemp[i+1,]<otemp[i+1,] & wtemp[i+2,]<otemp[i+2,]]*
    wtemp[i,][wtemp[i-1,]<otemp[i-1,] & wtemp[i,]<otemp[i,]
      & wtemp[i+1,]<otemp[i+1,] & wtemp[i+2,]<otemp[i+2,]])
                                        }

                        } # end of the "i" loop

# calculate the expectations associated with the beginning of the season
# and the end of the season, since they will have either no previous game
# or not future games to consider in the expectations

if (temp.y[1]==1 & temp.y[2]==1 & temp.y[3]==1) {
zstar.t[1]_sum(wtemp[1,][wtemp[1,]>otemp[1,] & wtemp[2,]>otemp[2,]])
```

```
zstarsq.t[1]_sum((wtemp[1,][wtemp[1,]>otemp[1,] & wtemp[2,]>otemp[2,]])*
          (wtemp[1,][wtemp[1,]>otemp[1,] & wtemp[2,]>otemp[2,]]))
zstar2.t[1]_sum(wtemp[2,][wtemp[1,]>otemp[1,] & wtemp[2,]>otemp[2,] &
 wtemp[3,]>otemp[3,]]*wtemp[1,][wtemp[1,]>otemp[1,] & wtemp[2,]>otemp[2,] &
 wtemp[3,]>otemp[3,]])
zstarl.t[1]_length(wtemp[1,][wtemp[1,]>otemp[1,] & wtemp[2,]>otemp[2,]])
zstar2l.t[1]_length(wtemp[2,][wtemp[1,]>otemp[1,] & wtemp[2,]>otemp[2,] &
 wtemp[3,]>otemp[3,]]*wtemp[1,][wtemp[1,]>otemp[1,] & wtemp[2,]>otemp[2,] &
 wtemp[3,]>otemp[3,]])
                          }
else if (temp.y[1]==1 & temp.y[2]==1 & temp.y[3]==0) {
zstar.t[1]_sum(wtemp[1,][wtemp[1,]>otemp[1,] & wtemp[2,]>otemp[2,]])
zstarsq.t[1]_sum((wtemp[1,][wtemp[1,]>otemp[1,] & wtemp[2,]>otemp[2,]])*
          (wtemp[1,][wtemp[1,]>otemp[1,] & wtemp[2,]>otemp[2,]]))
zstar2.t[1]_sum(wtemp[2,][wtemp[1,]>otemp[1,] & wtemp[2,]>otemp[2,] &
 wtemp[3,]<otemp[3,]]*wtemp[1,][wtemp[1,]>otemp[1,] & wtemp[2,]>otemp[2,] &
 wtemp[3,]<otemp[3,]])
zstarl.t[1]_length(wtemp[1,][wtemp[1,]>otemp[1,] & wtemp[2,]>otemp[2,]])
zstar2l.t[1]_length(wtemp[2,][wtemp[1,]>otemp[1,] & wtemp[2,]>otemp[2,] &
 wtemp[3,]<otemp[3,]]*wtemp[1,][wtemp[1,]>otemp[1,] & wtemp[2,]>otemp[2,] &
 wtemp[3,]<otemp[3,]])
                  }

else if (temp.y[1]==1 & temp.y[2]==0 & temp.y[3]==1) {
zstar.t[1]_sum(wtemp[1,][wtemp[1,]>otemp[1,] & wtemp[2,]<otemp[2,]])
zstarsq.t[1]_sum((wtemp[1,][wtemp[1,]>otemp[1,] & wtemp[2,]<otemp[2,]])*
          (wtemp[1,][wtemp[1,]>otemp[1,] & wtemp[2,]<otemp[2,]]))
zstar2.t[1]_sum(wtemp[2,][wtemp[1,]>otemp[1,] & wtemp[2,]<otemp[2,] &
 wtemp[3,]>otemp[3,]]*wtemp[1,][wtemp[1,]>otemp[1,] & wtemp[2,]<otemp[2,] &
 wtemp[3,]>otemp[3,]])
zstarl.t[1]_length(wtemp[1,][wtemp[1,]>otemp[1,] & wtemp[2,]<otemp[2,]])
zstar2l.t[1]_length(wtemp[2,][wtemp[1,]>otemp[1,] & wtemp[2,]<otemp[2,] &
 wtemp[3,]>otemp[3,]]*wtemp[1,][wtemp[1,]>otemp[1,] & wtemp[2,]<otemp[2,] &
 wtemp[3,]>otemp[3,]])
                          }
else if (temp.y[1]==1 & temp.y[2]==0 & temp.y[3]==0) {
zstar.t[1]_sum(wtemp[1,][wtemp[1,]>otemp[1,] & wtemp[2,]<otemp[2,]])
zstarsq.t[1]_sum((wtemp[1,][wtemp[1,]>otemp[1,] & wtemp[2,]<otemp[2,]])*
          (wtemp[1,][wtemp[1,]>otemp[1,] & wtemp[2,]<otemp[2,]]))
zstar2.t[1]_sum(wtemp[2,][wtemp[1,]>otemp[1,] & wtemp[2,]<otemp[2,] &
 wtemp[3,]<otemp[3,]]*wtemp[1,][wtemp[1,]>otemp[1,] & wtemp[2,]<otemp[2,] &
 wtemp[3,]<otemp[3,]])
zstarl.t[1]_length(wtemp[1,][wtemp[1,]>otemp[1,] & wtemp[2,]<otemp[2,]])
zstar2l.t[1]_length(wtemp[2,][wtemp[1,]>otemp[1,] & wtemp[2,]<otemp[2,] &
 wtemp[3,]<otemp[3,]]*wtemp[1,][wtemp[1,]>otemp[1,] & wtemp[2,]<otemp[2,] &
 wtemp[3,]<otemp[3,]])
                  }
else if (temp.y[1]==0 & temp.y[2]==1 & temp.y[3]==1) {
zstar.t[1]_sum(wtemp[1,][wtemp[1,]<otemp[1,] & wtemp[2,]>otemp[2,]])
```

```
zstarsq.t[1]_sum((wtemp[1,][wtemp[1,]<otemp[1,] & wtemp[2,]>otemp[2,]])*
            (wtemp[1,][wtemp[1,]<otemp[1,] & wtemp[2,]>otemp[2,]]))
zstar2.t[1]_sum(wtemp[2,][wtemp[1,]<otemp[1,] & wtemp[2,]>otemp[2,] &
 wtemp[3,]>otemp[3,]]*wtemp[1,][wtemp[1,]<otemp[1,] & wtemp[2,]>otemp[2,] &
 wtemp[3,]>otemp[3,]])
zstarl.t[1]_length(wtemp[1,][wtemp[1,]<otemp[1,] & wtemp[2,]>otemp[2,]])
zstar2l.t[1]_length(wtemp[2,][wtemp[1,]<otemp[1,] & wtemp[2,]>otemp[2,] &
 wtemp[3,]>otemp[3,]]*wtemp[1,][wtemp[1,]<otemp[1,] & wtemp[2,]>otemp[2,] &
 wtemp[3,]>otemp[3,]])
                              }

else if (temp.y[1]==0 & temp.y[2]==1 & temp.y[3]==0) {
zstar.t[1]_sum(wtemp[1,][wtemp[1,]<otemp[1,] & wtemp[2,]>otemp[2,]])
zstarsq.t[1]_sum((wtemp[1,][wtemp[1,]<otemp[1,] & wtemp[2,]>otemp[2,]])*
          (wtemp[1,][wtemp[1,]<otemp[1,] & wtemp[2,]>otemp[2,]]))
zstar2.t[1]_sum(wtemp[2,][wtemp[1,]<otemp[1,] & wtemp[2,]>otemp[2,] &
 wtemp[3,]<otemp[3,]]*wtemp[1,][wtemp[1,]<otemp[1,] & wtemp[2,]>otemp[2,] &
 wtemp[3,]<otemp[3,]])
zstarl.t[1]_length(wtemp[1,][wtemp[1,]<otemp[1,] & wtemp[2,]>otemp[2,]])
zstar2l.t[1]_length(wtemp[2,][wtemp[1,]<otemp[1,] & wtemp[2,]>otemp[2,] &
 wtemp[3,]<otemp[3,]]*wtemp[1,][wtemp[1,]<otemp[1,] & wtemp[2,]>otemp[2,] &
 wtemp[3,]<otemp[3,]])
                    }
else if (temp.y[1]==0 & temp.y[2]==0 & temp.y[3]==1) {
zstar.t[1]_sum(wtemp[1,][wtemp[1,]<otemp[1,] & wtemp[2,]<otemp[2,]])
zstarsq.t[1]_sum((wtemp[1,][wtemp[1,]<otemp[1,] & wtemp[2,]<otemp[2,]])*
          (wtemp[1,][wtemp[1,]<otemp[1,] & wtemp[2,]<otemp[2,]]))
zstar2.t[1]_sum(wtemp[2,][wtemp[1,]<otemp[1,] & wtemp[2,]<otemp[2,] &
 wtemp[3,]>otemp[3,]]*wtemp[1,][wtemp[1,]<otemp[1,] & wtemp[2,]<otemp[2,] &
 wtemp[3,]>otemp[3,]])
zstarl.t[1]_length(wtemp[1,][wtemp[1,]<otemp[1,] & wtemp[2,]<otemp[2,]])
zstar2l.t[1]_length(wtemp[2,][wtemp[1,]<otemp[1,] & wtemp[2,]<otemp[2,] &
 wtemp[3,]>otemp[3,]]*wtemp[1,][wtemp[1,]<otemp[1,] & wtemp[2,]<otemp[2,] &
 wtemp[3,]>otemp[3,]])
                              }
else {
zstar.t[1]_sum(wtemp[1,][wtemp[1,]<otemp[1,] & wtemp[2,]<otemp[2,]])
zstarsq.t[1]_sum((wtemp[1,][wtemp[1,]<otemp[1,] & wtemp[2,]<otemp[2,]])*
          (wtemp[1,][wtemp[1,]<otemp[1,] & wtemp[2,]<otemp[2,]]))
zstar2.t[1]_sum(wtemp[2,][wtemp[1,]<otemp[1,] & wtemp[2,]<otemp[2,] &
 wtemp[3,]<otemp[3,]]*wtemp[1,][wtemp[1,]<otemp[1,] & wtemp[2,]<otemp[2,] &
 wtemp[3,]<otemp[3,]])
zstarl.t[1]_length(wtemp[1,][wtemp[1,]<otemp[1,] & wtemp[2,]<otemp[2,]])
zstar2l.t[1]_length(wtemp[2,][wtemp[1,]<otemp[1,] & wtemp[2,]<otemp[2,] &
 wtemp[3,]<otemp[3,]]*wtemp[1,][wtemp[1,]<otemp[1,] & wtemp[2,]<otemp[2,] &
 wtemp[3,]<otemp[3,]])
                    }
```

```
# calculate the expectations dealing with the last games in the season

if (temp.y[n-2]==1 & temp.y[n-1]==1 & temp.y[n]==1) {
zstar.t[n-1]_sum(wtemp[n-1,][wtemp[n-2,]>otemp[n-2,] &
        wtemp[n-1,]>otemp[n-1,] & wtemp[n,]>otemp[n,]])
zstarsq.t[n-1]_sum((wtemp[n-1,][wtemp[n-2,]>otemp[n-2,] &
        wtemp[n-1,]>otemp[n-1,] & wtemp[n,]>otemp[n,]])*
        (wtemp[n-1,][wtemp[n-2,]>otemp[n-2,] &
        wtemp[n-1,]>otemp[n-1,] & wtemp[n,]>otemp[n,]]))
zstar.t[n]_sum(wtemp[n,][wtemp[n-1,]>otemp[n-1,] & wtemp[n,]>otemp[n,]])
zstarsq.t[n]_sum((wtemp[n,][wtemp[n-1,]>otemp[n-1,] & wtemp[n,]>otemp[n,]])*
        (wtemp[n,][wtemp[n-1,]>otemp[n-1,] & wtemp[n,]>otemp[n,]]))
zstar2.t[n-1]_sum(wtemp[n,][wtemp[n-2,]>otemp[n-2,] &
 wtemp[n-1,]>otemp[n-1,] & wtemp[n,]>otemp[n,]]*wtemp[n-1,]
 [wtemp[n-2,]>otemp[n-2,] & wtemp[n-1,]>otemp[n-1,] & wtemp[n,]>otemp[n,]])
zstarl.t[n-1]_length(wtemp[n-1,][wtemp[n-2,]>otemp[n-2,] &
        wtemp[n-1,]>otemp[n-1,] & wtemp[n,]>otemp[n,]])
zstarl.t[n]_length(wtemp[n,][wtemp[n-1,]>otemp[n-1,] & wtemp[n,]>otemp[n,]])
zstar2l.t[n-1]_length(wtemp[n,][wtemp[n-2,]>otemp[n-2,] &
 wtemp[n-1,]>otemp[n-1,] & wtemp[n,]>otemp[n,]]*wtemp[n-1,]
 [wtemp[n-2,]>otemp[n-2,] & wtemp[n-1,]>otemp[n-1,] & wtemp[n,]>otemp[n,]])
                    }

else if (temp.y[n-2]==1 & temp.y[n-1]==1 & temp.y[n]==0) {
zstar.t[n-1]_sum(wtemp[n-1,][wtemp[n-2,]>otemp[n-2,] &
        wtemp[n-1,]>otemp[n-1,] & wtemp[n,]<otemp[n,]])
zstarsq.t[n-1]_sum((wtemp[n-1,][wtemp[n-2,]>otemp[n-2,] &
        wtemp[n-1,]>otemp[n-1,] & wtemp[n,]<otemp[n,]])*
        (wtemp[n-1,][wtemp[n-2,]>otemp[n-2,] &
        wtemp[n-1,]>otemp[n-1,] & wtemp[n,]<otemp[n,]]))
zstar.t[n]_sum(wtemp[n,][wtemp[n-1,]>otemp[n-1,] & wtemp[n,]<otemp[n,]])
zstarsq.t[n]_sum((wtemp[n,][wtemp[n-1,]>otemp[n-1,] & wtemp[n,]<otemp[n,]])*
        (wtemp[n,][wtemp[n-1,]>otemp[n-1,] & wtemp[n,]<otemp[n,]]))
zstar2.t[n-1]_sum(wtemp[n,][wtemp[n-2,]>otemp[n-2,] &
 wtemp[n-1,]>otemp[n-1,] & wtemp[n,]<otemp[n,]]*wtemp[n-1,]
 [wtemp[n-2,]>otemp[n-2,] & wtemp[n-1,]>otemp[n-1,] & wtemp[n,]<otemp[n,]])
zstarl.t[n-1]_length(wtemp[n-1,][wtemp[n-2,]>otemp[n-2,] &
        wtemp[n-1,]>otemp[n-1,] & wtemp[n,]<otemp[n,]])
zstarl.t[n]_length(wtemp[n,][wtemp[n-1,]>otemp[n-1,] & wtemp[n,]<otemp[n,]])
zstar2l.t[n-1]_length(wtemp[n,][wtemp[n-2,]>otemp[n-2,] &
 wtemp[n-1,]>otemp[n-1,] & wtemp[n,]<otemp[n,]]*wtemp[n-1,]
 [wtemp[n-2,]>otemp[n-2,] & wtemp[n-1,]>otemp[n-1,] & wtemp[n,]<otemp[n,]])
                    }

else if (temp.y[n-2]==1 & temp.y[n-1]==0 & temp.y[n]==1) {
zstar.t[n-1]_sum(wtemp[n-1,][wtemp[n-2,]>otemp[n-2,] &
        wtemp[n-1,]<otemp[n-1,] & wtemp[n,]>otemp[n,]])
zstarsq.t[n-1]_sum((wtemp[n-1,][wtemp[n-2,]>otemp[n-2,] &
```

```
        wtemp[n-1,]<otemp[n-1,] & wtemp[n,]>otemp[n,]])*
        (wtemp[n-1,][wtemp[n-2,]>otemp[n-2,] &
        wtemp[n-1,]<otemp[n-1,] & wtemp[n,]>otemp[n,]]))
zstar.t[n]_sum(wtemp[n,][wtemp[n-1,]<otemp[n-1,] & wtemp[n,]>otemp[n,]])
zstarsq.t[n]_sum((wtemp[n,][wtemp[n-1,]<otemp[n-1,] & wtemp[n,]>otemp[n,]])*
      (wtemp[n,][wtemp[n-1,]<otemp[n-1,] & wtemp[n,]>otemp[n,]]))
zstar2.t[n-1]_sum(wtemp[n,][wtemp[n-2,]>otemp[n-2,] &
 wtemp[n-1,]<otemp[n-1,] & wtemp[n,]>otemp[n,]]*wtemp[n-1,]
 [wtemp[n-2,]>otemp[n-2,] & wtemp[n-1,]<otemp[n-1,] & wtemp[n,]>otemp[n,]])
zstarl.t[n-1]_length(wtemp[n-1,][wtemp[n-2,]>otemp[n-2,] &
        wtemp[n-1,]<otemp[n-1,] & wtemp[n,]>otemp[n,]])
zstarl.t[n]_length(wtemp[n,][wtemp[n-1,]<otemp[n-1,] & wtemp[n,]>otemp[n,]])
zstar2l.t[n-1]_length(wtemp[n,][wtemp[n-2,]>otemp[n-2,] &
 wtemp[n-1,]<otemp[n-1,] & wtemp[n,]>otemp[n,]]*wtemp[n-1,]
 [wtemp[n-2,]>otemp[n-2,] & wtemp[n-1,]<otemp[n-1,] & wtemp[n,]>otemp[n,]])
                              }

else if (temp.y[n-2]==1 & temp.y[n-1]==0 & temp.y[n]==0) {
zstar.t[n-1]_sum(wtemp[n-1,][wtemp[n-2,]>otemp[n-2,] &
        wtemp[n-1,]<otemp[n-1,] & wtemp[n,]<otemp[n,]])
zstarsq.t[n-1]_sum((wtemp[n-1,][wtemp[n-2,]>otemp[n-2,] &
        wtemp[n-1,]<otemp[n-1,] & wtemp[n,]<otemp[n,]])*
        (wtemp[n-1,][wtemp[n-2,]>otemp[n-2,] &
        wtemp[n-1,]<otemp[n-1,] & wtemp[n,]<otemp[n,]]))
zstar.t[n]_sum(wtemp[n,][wtemp[n-1,]<otemp[n-1,] & wtemp[n,]<otemp[n,]])
zstarsq.t[n]_sum((wtemp[n,][wtemp[n-1,]<otemp[n-1,] & wtemp[n,]<otemp[n,]])*
        (wtemp[n,][wtemp[n-1,]<otemp[n-1,] & wtemp[n,]<otemp[n,]]))
zstar2.t[n-1]_sum(wtemp[n,][wtemp[n-2,]>otemp[n-2,] &
 wtemp[n-1,]<otemp[n-1,] & wtemp[n,]<otemp[n,]]*wtemp[n-1,]
 [wtemp[n-2,]>otemp[n-2,] & wtemp[n-1,]<otemp[n-1,] & wtemp[n,]<otemp[n,]])
zstarl.t[n-1]_length(wtemp[n-1,][wtemp[n-2,]>otemp[n-2,] &
        wtemp[n-1,]<otemp[n-1,] & wtemp[n,]<otemp[n,]])
zstarl.t[n]_length(wtemp[n,][wtemp[n-1,]<otemp[n-1,] & wtemp[n,]<otemp[n,]])
zstar2l.t[n-1]_length(wtemp[n,][wtemp[n-2,]>otemp[n-2,] &
 wtemp[n-1,]<otemp[n-1,] & wtemp[n,]<otemp[n,]]*wtemp[n-1,]
 [wtemp[n-2,]>otemp[n-2,] & wtemp[n-1,]<otemp[n-1,] & wtemp[n,]<otemp[n,]])
                              }

else if (temp.y[n-2]==0 & temp.y[n-1]==1 & temp.y[n]==1) {
zstar.t[n-1]_sum(wtemp[n-1,][wtemp[n-2,]<otemp[n-2,] &
        wtemp[n-1,]>otemp[n-1,] & wtemp[n,]>otemp[n,]])
zstarsq.t[n-1]_sum((wtemp[n-1,][wtemp[n-2,]<otemp[n-2,] &
        wtemp[n-1,]>otemp[n-1,] & wtemp[n,]>otemp[n,]])*
        (wtemp[n-1,][wtemp[n-2,]<otemp[n-2,] &
        wtemp[n-1,]>otemp[n-1,] & wtemp[n,]>otemp[n,]]))
zstar.t[n]_sum(wtemp[n,][wtemp[n-1,]>otemp[n-1,] & wtemp[n,]>otemp[n,]])
zstarsq.t[n]_sum((wtemp[n,][wtemp[n-1,]>otemp[n-1,] & wtemp[n,]>otemp[n,]])*
        (wtemp[n,][wtemp[n-1,]>otemp[n-1,] & wtemp[n,]>otemp[n,]]))
zstar2.t[n-1]_sum(wtemp[n,][wtemp[n-2,]<otemp[n-2,] &
```

```
wtemp[n-1,]>otemp[n-1,] & wtemp[n,]>otemp[n,]]*wtemp[n-1,]
 [wtemp[n-2,]<otemp[n-2,] & wtemp[n-1,]>otemp[n-1,] & wtemp[n,]>otemp[n,]])
zstarl.t[n-1]_length(wtemp[n-1,][wtemp[n-2,]<otemp[n-2,] &
          wtemp[n-1,]>otemp[n-1,] & wtemp[n,]>otemp[n,]])
zstarl.t[n]_length(wtemp[n,][wtemp[n-1,]>otemp[n-1,] & wtemp[n,]>otemp[n,]])
zstar2l.t[n-1]_length(wtemp[n,][wtemp[n-2,]<otemp[n-2,] &
 wtemp[n-1,]>otemp[n-1,] & wtemp[n,]>otemp[n,]]*wtemp[n-1,]
 [wtemp[n-2,]<otemp[n-2,] & wtemp[n-1,]>otemp[n-1,] & wtemp[n,]>otemp[n,]])
                                   }

else if (temp.y[n-2]==0 & temp.y[n-1]==1 & temp.y[n]==0) {
zstar.t[n-1]_sum(wtemp[n-1,][wtemp[n-2,]<otemp[n-2,] &
          wtemp[n-1,]>otemp[n-1,] & wtemp[n,]<otemp[n,]])
zstarsq.t[n-1]_sum((wtemp[n-1,][wtemp[n-2,]<otemp[n-2,] &
          wtemp[n-1,]>otemp[n-1,] & wtemp[n,]<otemp[n,]])*
           (wtemp[n-1,][wtemp[n-2,]<otemp[n-2,] &
          wtemp[n-1,]>otemp[n-1,] & wtemp[n,]<otemp[n,]]))
zstar.t[n]_sum(wtemp[n,][wtemp[n-1,]>otemp[n-1,] & wtemp[n,]<otemp[n,]])
zstarsq.t[n]_sum((wtemp[n,][wtemp[n-1,]>otemp[n-1,] & wtemp[n,]<otemp[n,]])*
           (wtemp[n,][wtemp[n-1,]>otemp[n-1,] & wtemp[n,]<otemp[n,]]))
zstar2.t[n-1]_sum(wtemp[n,][wtemp[n-2,]<otemp[n-2,] &
 wtemp[n-1,]>otemp[n-1,] & wtemp[n,]<otemp[n,]]*wtemp[n-1,]
 [wtemp[n-2,]<otemp[n-2,] & wtemp[n-1,]>otemp[n-1,] & wtemp[n,]<otemp[n,]])
zstarl.t[n-1]_length(wtemp[n-1,][wtemp[n-2,]<otemp[n-2,] &
          wtemp[n-1,]>otemp[n-1,] & wtemp[n,]<otemp[n,]])
zstarl.t[n]_length(wtemp[n,][wtemp[n-1,]>otemp[n-1,] & wtemp[n,]<otemp[n,]])
zstar2l.t[n-1]_length(wtemp[n,][wtemp[n-2,]<otemp[n-2,] &
 wtemp[n-1,]>otemp[n-1,] & wtemp[n,]<otemp[n,]]*wtemp[n-1,]
 [wtemp[n-2,]<otemp[n-2,] & wtemp[n-1,]>otemp[n-1,] & wtemp[n,]<otemp[n,]])
                                   }

else if (temp.y[n-2]==0 & temp.y[n-1]==0 & temp.y[n]==1) {
zstar.t[n-1]_sum(wtemp[n-1,][wtemp[n-2,]<otemp[n-2,] &
          wtemp[n-1,]<otemp[n-1,] & wtemp[n,]>otemp[n,]])
zstarsq.t[n-1]_sum((wtemp[n-1,][wtemp[n-2,]<otemp[n-2,] &
          wtemp[n-1,]<otemp[n-1,] & wtemp[n,]>otemp[n,]])*
          (wtemp[n-1,][wtemp[n-2,]<otemp[n-2,] &
          wtemp[n-1,]<otemp[n-1,] & wtemp[n,]>otemp[n,]]))
zstar.t[n]_sum(wtemp[n,][wtemp[n-1,]<otemp[n-1,] & wtemp[n,]>otemp[n,]])
zstarsq.t[n]_sum((wtemp[n,][wtemp[n-1,]<otemp[n-1,] & wtemp[n,]>otemp[n,]])*
           (wtemp[n,][wtemp[n-1,]<otemp[n-1,] & wtemp[n,]>otemp[n,]]))
zstar2.t[n-1]_sum(wtemp[n,][wtemp[n-2,]<otemp[n-2,] &
 wtemp[n-1,]<otemp[n-1,] & wtemp[n,]>otemp[n,]]*wtemp[n-1,]
 [wtemp[n-2,]<otemp[n-2,] & wtemp[n-1,]<otemp[n-1,] & wtemp[n,]>otemp[n,]])
zstarl.t[n-1]_length(wtemp[n-1,][wtemp[n-2,]<otemp[n-2,] &
          wtemp[n-1,]<otemp[n-1,] & wtemp[n,]>otemp[n,]])
zstarl.t[n]_length(wtemp[n,][wtemp[n-1,]<otemp[n-1,] & wtemp[n,]>otemp[n,]])
zstar2l.t[n-1]_length(wtemp[n,][wtemp[n-2,]<otemp[n-2,] &
 wtemp[n-1,]<otemp[n-1,] & wtemp[n,]>otemp[n,]]*wtemp[n-1,]
```

```
 [wtemp[n-2,]<otemp[n-2,] & wtemp[n-1,]<otemp[n-1,] & wtemp[n,]>otemp[n,]])
                                        }

else {
zstar.t[n-1]_sum(wtemp[n-1,][wtemp[n-2,]<otemp[n-2,] &
          wtemp[n-1,]<otemp[n-1,] & wtemp[n,]<otemp[n,]])
zstarsq.t[n-1]_sum((wtemp[n-1,][wtemp[n-2,]<otemp[n-2,] &
          wtemp[n-1,]<otemp[n-1,] & wtemp[n,]<otemp[n,]])*
          (wtemp[n-1,][wtemp[n-2,]<otemp[n-2,] &
          wtemp[n-1,]<otemp[n-1,] & wtemp[n,]<otemp[n,]]))
zstar.t[n]_sum(wtemp[n,][wtemp[n-1,]<otemp[n-1,] & wtemp[n,]<otemp[n,]])
zstarsq.t[n]_sum((wtemp[n,][wtemp[n-1,]<otemp[n-1,] & wtemp[n,]<otemp[n,]])*
          (wtemp[n,][wtemp[n-1,]<otemp[n-1,] & wtemp[n,]<otemp[n,]]))
zstar2.t[n-1]_sum(wtemp[n,][wtemp[n-2,]<otemp[n-2,] &
 wtemp[n-1,]<otemp[n-1,] & wtemp[n,]<otemp[n,]]*wtemp[n-1,]
 [wtemp[n-2,]<otemp[n-2,] & wtemp[n-1,]<otemp[n-1,] & wtemp[n,]<otemp[n,]])
zstarl.t[n-1]_length(wtemp[n-1,][wtemp[n-2,]<otemp[n-2,] &
          wtemp[n-1,]<otemp[n-1,] & wtemp[n,]<otemp[n,]])
zstarl.t[n]_length(wtemp[n,][wtemp[n-1,]<otemp[n-1,] & wtemp[n,]<otemp[n,]])
zstar2l.t[n-1]_length(wtemp[n,][wtemp[n-2,]<otemp[n-2,] &
 wtemp[n-1,]<otemp[n-1,] & wtemp[n,]<otemp[n,]]*wtemp[n-1,]
 [wtemp[n-2,]<otemp[n-2,] & wtemp[n-1,]<otemp[n-1,] & wtemp[n,]<otemp[n,]])
        }

# replace all cases in which the observed data pattern was not simulated
# with either the team mean or the team mean squared

zstar.t[zstar.t==0]_alpha[j]
zstarsq.t[zstarsq.t==0]_alpha[j]*alpha[j]
zstar2.t[zstar2.t==0]_alpha[j]*alpha[j]
zstarl.t[zstarl.t==0]_ 1
zstar2l.t[zstar2l.t==0]_ 1


# build the final vector during the "j" loop
zstar_ c(zstar,zstar.t)
zstarsq_ c(zstarsq,zstarsq.t)
zstar2_ c(zstar2,zstar2.t)
zstar.l_ c(zstar.l, zstarl.t)
zstar2.l_ c(zstar2.l, zstar2l.t)


                          } # end of the "j" loop.

# remove the dummy variable in the first cell
zstar_ zstar[2:length(zstar)]
zstarsq_ zstarsq[2:length(zstarsq)]
zstar2_ zstar2[2:length(zstar2)]
zstar.l_ zstar.l[2:length(zstar.l)]
```

```
zstar2.l_ zstar2.l[2:length(zstar2.l)]


# end.time_ proc.time()[1:2]
# run.time_ end.time - begin.time


return(zstar,zstarsq,zstar2,zstar.l,zstar2.l)

                        } # end of function



# FILE NAME:  RHOBETA2.SPS

# THIS PORTION OF THE PROGRAMS ESTIMATES THE PARAMETERS BASED ON
# THE SIMULATED EXPECTATIONS.
# THE FOLLOWING VECTORS WILL BE NEEDED:

#  X = THE DESIGN MATRIX
#  ZSTAR, ZSTARSQ, ZSTAR2 = SIMULATED EXPECTATIONS
#  NUMT, NUMC = THE NUMBER OF TEAMS IN THE LEAGUE AND THE NUMBER
#            OF COVARIATES BEING ESTIMATED.
#   N = THE NUMBER OF GAMES EACH TEAM PLAYED - ASSUMED TO BE EQUAL!
#   RHO = THE CURRENT ESTIMATE OF THE AUTOCORRELATION
#   ALPHA = THE CURRENT PARAMETER ESTIMATES OF THE COVARIATES.



rhobeta.sps_function(x,zstar,zstarsq,zstar2,numt,numc,n,rho,alpha,iterations)   {

change_ numeric(numt+numc-1)


rho1_ 0
rho2_ 0



alpha_alpha[2:(numt+numc)] # drop the constrain, alpha[1]=0


x.new_ rep(0,(numt+numc-1))
rx_ rho*x  # transformed design matrix

z.new_0
rz_ rho*zstar  # transformed expectations

# analyze the data for each team individually
```

```
for (j in 1:numt)  {

# subset the full data matrices to the parts specific to team j
  x.temp_x[(((j-1)*n)+1):(j*n),]
  rx.temp_rx[(((j-1)*n)+1):(j*n),]

  x.temp2_x.temp[2:n,]
  x.temp.n1_x.temp[1:(n-1),]
  rx.temp2_rx.temp[1:(n-1),]
  x.rx_x.temp2 - rx.temp2
  x.new_ rbind(x.new,x.rx)
# creates the transformed covariates for i=2 to n.

# defines the portions of zstar specific to team j that are
# to be used in the estimation of alpha.  Does the same for
# zstarsq and zstar2 for the estimation of rho.

  z.temp_zstar[(((j-1)*n)+1):(j*n)]
  rz.temp_rz[(((j-1)*n)+1):(j*n)]
  zsq.temp_zstarsq[(((j-1)*n)+1):(j*n)]
  z2.temp_zstar2[(((j-1)*(n-1))+1):(j*(n-1))]

  z.temp2_z.temp[2:n]
  z.temp.n1_ z.temp[1:(n-1)]
  rz.temp2_rz.temp[1:(n-1)]


  z.rz_ z.temp2 - rz.temp2
  z.new_ c(z.new, z.rz)

#  these create the transformed variables that are used in the usual
#  ordinary regression routine for i=2 to n.

# estimation of RHO for team j (assumes that RHO is constant for all teams)

  x1_ sum(z2.temp)  #sum of zstar2
  x2_ sum(z.temp2 * (x.temp.n1 %*% alpha)) #sum of zstar*alpha'x[i-1]
  x3_ sum(z.temp.n1 * (x.temp2 %*% alpha)) #sum of zstar[i-1]*alpha'x[i]
  x4_ sum((x.temp2 %*% alpha) * (x.temp.n1 %*% alpha)) #sum of alpha'x[i]*
                                          # alpha'x[i-1]

  rho1_rho1 + (x1-x2-x3+x4)  # numerator of RHO



  x5_ sum(zsq.temp[2:(n-1)])  #sum of zstarsq
  x6_ sum(2 * z.temp[2:(n-1)] * (x.temp[2:(n-1),] %*% alpha))
                     #sum of 2*zstar*alpha'x
  x7_ sum((x.temp[2:(n-1),] %*% alpha)^2)  #sum of (alpha'x)^2
```

```
 rho2_ rho2 + (x5-x6+x7) #summation of denominator from i=2 to N-1

            } # end of the "j" loop.

x.new_x.new[2:length(x.new[,1]),]
z.new_z.new[2:length(z.new)] #drop the dummy variable in the vector.

# ESTIMATE ALPHA AND RHO

temp10_ solve(t(x.new) %*% x.new) %*% t(x.new) %*% z.new
rho10_ rho1/rho2

# test for convergence


test_rep(10,numt+numc-1)

alpha.new_c(temp10)
compare_alpha
compare2_rho
rel.change1_abs((temp10 - compare)/compare)
rel.change2_abs((rho10 - compare2)/compare2)
rel.change_ c(rel.change1, rel.change2)
abs.change1_ abs(temp10 - compare)
abs.change2_ abs(rho10 - compare2)
abs.change_ c(abs.change1, abs.change2)

for (i in 1:(numt+numc)) {
  if (rel.change[i] < 0.015) test[i]_ 0
  else {
   if (abs.change[i] < 0.005) test[i]_ 0
   else test[i] _ 100
     } # end of "else" loop
                 } # end of "for" loop

alpha_alpha.new
rho_ rho10

test2_ max(test)
iterations_iterations+1


return(rho,alpha,test,test2,rel.change,abs.change,iterations)

                   } # end of function
```

APPENDIX 3 Creation of the additional variables needed for the approximation models

```
# FILE NAME : ALT2.SPS

# THIS PROGRAM CREATES THE COMPONENTS NEEDED TO FIT TWO
# APPROXIMATIONS TO THE SERIAL CORRELATION MODEL. THESE
# APPROXIMATIONS DO NOT USE THE EM ALGORITHM. INSTEAD THEY USE
# GLM'S.

# THE FOLLOWING VARIABLES ARE NEEDED BY THE FUNCTION:

#   TEAM, OPP = VECTORS SPECIFYING THE TEAM AND OPPONENT IN EACH
#         GAME PLAYED THAT SEASON.
#   WIN = VECTOR INDICATING IF TEAM WON OR LOST THE GAME
#   INDICATOR = BINARY VARIABLE INDICATING IF TEAM WAS AT HOME OR ON
#       THE ROAD.
#   COV = VECTOR OR MATRIX OF COVARIATES.
#   NUMT, NUMC = NUMBER OF TEAMS IN THE LEAGUE AND THE NUMBER OF
#              COVARIATES
#   ALPHA = PARAMETER ESTIMATES FROM A CASE V MODEL
#   GT = VECTOR INDICATING NUMBER OF GAMES THE OPP HAS PLAYED WHEN
#       FACING TEAM.
#   N = NUMBER OF GAMES EACH TEAM PLAYS
#   GAME.SEQ = VECTOR IDENTIFYING THE GAME NUMBER FOR EACH OF THE
#           TEAMS. I.E. (1,2,...,162,1,2,....162,.....,1,...,162)
#   MEANS=F: DO NOT LOOK AT RANKING MODEL
#   UNIVARIATE=T: ALL COVARIATES HAVE A SINGLE PARAMETER
#   IND=T: THE HOME FIELD IS USED AS A COVARIATE
#   COVARIATE=T: COVARIATES ARE TO BE INCLUDED IN THE MODEL

# NOTE: IF IND=T THEN THE FIRST COLUMN OF COV MUST BE ALL ONES.


altuv.sps_ function(team,opp,win,indicator,cov,numt,numc,alpha,gt,n,
        game.seq, MEANS=F, UNIVARIATE=T, IND=T, COVARIATE=T)  {

 x.out_ createx.sps(team,numt,numc,cov,indicator,IND)
 x_ x.out  # create the design matrix for the model

#CREATE EXPECTATIONS BASED ON THE PROBIT ESTIMATES OF ALPHA

scale2_ numeric(length(team))
scale3_ numeric(length(team))
scale_ numeric(length(team))
zstar_ numeric(length(team))
phi_ numeric(length(team))
PHI_ numeric(length(team)) # initialize needed variables
```

```
if (MEANS==F)  {  # Thus, covariates are to be included in the model

 if (UNIVARIATE==F)  {
# multiple covariates are to be included in the model.  Individual
# parameters/team can be included by properly specifying the matrix
# of covariates, COV.

 scale2_ rep(0,length(team))
 scale3_ rep(0,length(team))

# compute the proper components of alpha[i] - alpha[j].

   for (i in 1:numc)   {
     scale2_ alpha[numt+i]*indicator*cov[,i] + scale2
     scale3_ alpha[numt+i]*(1-indicator)*cov[,i] + scale3
                   }  # end of the for loop
                }   # end of the UNIVARIATE loop


# single covariate in the model.  Again due to duplicity only the values
# associated with the home team are included.
   else  {
     scale2_ alpha[numt+numc]*indicator*cov
     scale3_ alpha[numt+numc]*(1-indicator)*cov
        }  # end of else statement

scale_ (alpha[team] + scale2 - alpha[opp] - scale3)/sqrt(2)
phi_ dnorm(scale)
PHI_ pnorm(scale)

# EXPECTATIONS REQUIRED BY THE EM ALOGORITHM
zstar_ (alpha[team]+scale2)+win*phi/(PHI*sqrt(2))-(1-win)*phi/
       ((1-PHI)*sqrt(2))

       }  # end of MEANS loop

else    {  # No covariates are to be included in the model.  Simple
        # ranking model

   scale_ (alpha[team] - alpha[opp])/sqrt(2)
   phi_ dnorm(scale)
   PHI_ pnorm(scale)
   zstar_ alpha[team]+win*phi/(PHI*sqrt(2))-(1-win)*phi/((1-PHI)*sqrt(2))
     }  # end of the else statement

# CREATE THE VARIABLES U, V THAT ARE TO BE USED IN THE PROBIT MODEL
# TO APPROXIMATE THE EFFECT OF A LAG-1 DEPENDENCE ON THE GAME
# OUTCOMES.
```

```
alpha_ alpha[2:(numt+numc)] # eliminates the constraints. Must be altered if
                            # other constraints, such as home field advantage
              # for team one=0, are used.
zstar.h_ 99
win.h_ 99
zstar.vt_ numeric(n)
win.vt_ numeric(n)
prob.vt_ numeric(n)
zstar.v_ 99
win.v_ 99
prob.h_ 99
prob.v_ 99
x.h_ rep(0,(numt+numc-1))
x.v_ rep(0,(numt+numc-1)) # initialize needed variables


for (j in 1:numt)   { # work with a single team at a time

 ind.t_ indicator[team==j] # home/away variable for team j
 zstar.ht_ zstar[team==j]
 zstar.ht_ c(0,zstar.ht[1:(n-1)]) #create lag-1 expectation for home team
 win.ht_ win[team==j]
 win.ht_ c(0,win.ht[1:(n-1)])  #create lag-1 outcomes for home team
 prob.ht_ PHI[team==j]
 prob.ht_ c(0,prob.ht[1:(n-1)]) #create lag-1 pi's for home team.
 x.ht_ x[(((j-1)*n)+1):(j*n),]
# formation of lag-1 vectors of the expectations, observed outcomes, and
# fitted probabilities for team j. Also the portion of the design matrix
# specific to team j.

 gt.t_ gt[(((j-1)*n)+1):(j*n)] # game of the season that team j's
                               # opponent was playing in.


# account for the first game of the season. Use the constraint
# that pi[0]=w[0]=x[0]=0.

 if (ind.t[1]==1) {

        x.h_ rbind(x.h, rep(0,(numt+numc-1)))

# account for the opponents of team j's first game
 if (ind.t[1]==1 & game.seq[gt.t[1]]==1)    {

    x.v_rbind(x.v,rep(0,(numt+numc-1)))

    zstar.vt[1]_ 0
    win.vt[1]_ 0
```

```
prob.vt[1]_ 0
          } # end of the "if" loop


# accounts for the possibility that team j's first game is actually
# their opponents second (or third) game.  Note that one may get faulty numbers
# if IND.T[1]==0 but their values will be discarded later anyway.

    else   {
     x.v_ rbind(x.v, x[((gt.t[1])-1),])
     zstar.vt[1]_ zstar[(gt.t[1])-1]
     win.vt[1]_ win[(gt.t[1])-1]
     prob.vt[1]_ PHI[(gt.t[1])-1]
         } # end of else loop

              } # end of the original IF loop

# form the lag-1 vectors corresponding to the opponents expected values,
# observed outcomes, and predicted probability of winning
 for (i in 2:n)   {

   zstar.vt[i]_ zstar[(gt.t[i])-1]
   win.vt[i]_ win[(gt.t[i])-1]
   prob.vt[i]_ PHI[(gt.t[i])-1]

# form the new matrix based on the explanatory variables for both the
# team and their opponents previous game.
   if (ind.t[i]==1)  {

    x.vt_ x[((gt.t[i])-1),]
    x.v_ rbind(x.v,x.vt)
    x.ht2_ x.ht[i-1,]
    x.h_ rbind(x.h, x.ht2)

              } # end of the "if" loop

            } # end of the "i" loop


zstar.v_ c(zstar.v, zstar.vt)
win.v_ c(win.v, win.vt)
prob.h_ c(prob.h, prob.ht)
prob.v_ c(prob.v, prob.vt)

zstar.h_ c(zstar.h, zstar.ht)
win.h_ c(win.h, win.ht)
# build the final vectors that include the results from all j teams.

         } # end of the "j" loop
```

```
# remove the dummy first value from each of the vectors

zstar.v_ zstar.v[2:length(zstar.v)]
win.v_ win.v[2:length(win.v)]
zstar.h_ zstar.h[2:length(zstar.h)]
win.h_ win.h[2:length(win.h)]
prob.h_ prob.h[2:length(prob.h)]
prob.v_ prob.v[2:length(prob.v)]
x.h_ x.h[2:(length(team[indicator==1])+1),]
x.v_ x.v[2:(length(team[indicator==1])+1),]

# use only those values that correspond to the next game being a home game,
# which is all that is required when using probit regression.

zstar.v_ zstar.v[indicator==1]
win.v_ win.v[indicator==1]
zstar.h_ zstar.h[indicator==1]
win.h_ win.h[indicator==1]
prob.h_ prob.h[indicator==1]
prob.v_ prob.v[indicator==1]


# FORM THE VARIABLES U AND V THAT WILL BE USED IN THE PROBIT
# APPROXIMATION MODELS.

x.new_ x.h - x.v
u_ zstar.h - zstar.v - (x.new %*% alpha)
v_ (win.h - prob.h) - (win.v - prob.v)

return(u,v)

                    } # end of the function
```