Hierarchical Models for Estimating Individual Ratings from Group Competitions

Joshua Menke, C. Shane Reese, and Tony Martinez

August 31, 2007

Abstract

Providing direct and indirect contributions of more than $18 billion to the nation's gross output in 2004, the computer and video gaming industry is one of the fastest-growing sectors of entertainment. A large part of that market includes team-oriented online games. In fact, according a recent study online gaming is the most popular online entertainment activity in the United States. Players in these games often have a high-level of interest in statistics that help them assess their ability compared to other players. However few models exist that estimate individual player ratings from team competitions. There are models that can be used at the team-level, however the dynamic nature of the teams in the more popular public-style play of these games makes it necessary to build team strengths from player abilities. The following presents a model that describe team abilities in terms of how well the individual players on the teams contribute to their team's winning. In addition, the model presented includes parameters that estimate other characteristics of the games themselves. The model is posed in a hierarchical Bayesian framework. In addition to giving players a better estimate of their skill, this model can also be used to improve current gameplay, and create more enjoyable games in the future. Companies and servers that apply well-developed statistics for assessing their players' abilities are more likely to attract and retain players, leading to greater success in the industry. The model is fit using both *Markov-Chain Monte Carlo* (MCMC) and a recursive updating method. As measured on 4,675 matches, the recursive method results in an accuracy of 73% when used to predict the outcomes of the matches used to estimate player ratings. In addition, it is shown that this method is fast enough to be used in real-time whereas MCMC is not.

1 Introduction

According to one marketing research firm, the computer gaming industry is the fastest-growing sector of entertainment (DFC Intelligence, 2006). Robert Crandall, a senior fellow with the Brookings Institution, and Gregory Sidak, visiting professor of Law at Georgetown University stated in a recent study that "The entertainment software business provided direct and indirect contributions of more than $18 billion to the nation's gross output in 2004" (Crandall and Sidakk, 2006). In fact, "video games represent an $11 billion U.S. entertainment industry, larger than Hollywood box office sales" (DFC Intelligence, 2006). A large portion of this revenue comes from the online gaming sector, which is projected to bring in $2.5 billion in 2006 (Jarret, 2003). 44% of all video and computer gaming players say they play online in addition to offline games (The Entertainment Software Association, 2006). According to Parks Associates (2007), online gaming is the most popular online entertainment activity in the U.S., surpassing both video sharing and social networking. One of the more popular genres in online gaming involves team competitions. This includes games based on popular sports, but more popular are team-based first-person shooter games which include blockbuster titles like Halo 2 and Half-Life 2. Millions of "gamers" can be found playing these games every day.

Most online players prefer having a method of tracking their progress in the games they play, and comparing themselves to other players. If a player feels like he or she can achieve some goal with a given statistic, they are more likely to continue playing. Games and servers that provide better developed statistical methods for comparing a player's skill are more likely to retain a larger player base, and hence be more profitable. Therefore, having good ways of rating players benefits the players, the companies producing the games, and the companies running servers for the games. In addition, if aspects of the game besides the players can be judged statistically, the developers can use this information to improve the quality of newer games. For example, in this paper we show how the fairness of the "field" two teams play on can be analyzed. This can be used to improve the fairness of gameplay.

Most players of online team games play on "public servers". By public we mean that players are free to join, leave, and switch teams as often as they like. By server, we refer to the software that maintains the state for a given game and match and determines when a match is over and who the victors are. Public matches can be highly dynamic in terms of which players are on either team and therefore the ratings of the teams can not be estimated at the group level, but need to be built from individual ratings. Unfortunately, for team-based games, there are few systems in place that estimate individual player ratings from group competitions. The following presents an extension of the model proposed by Bradley and Terry (1952) for

paired-comparisons that estimates individual player ratings based on the winners of team competitions.

There are several types of statistics that can be tracked in these team competitive online games, many of which are interesting at the individual level, but are not consistent indicators of a player's team contribution. The goal of the model in this paper is to estimate how well a player contributes to his or her team winning the match. Therefore, the ratings are estimated only in terms of the wins a player achieves while playing on their given teams. It should be noted that the ratings here are not necessarily strong indicators of how these players would play one-on-one, but rather in team settings.

There are several reasons why ranking and rating players in public, online, team-based competitions can lead to an increase in the number of players that play and continue to play a given game or play on a given server:

- It gives players a measure of their performance and hence motivation to improve themselves. Our experience with running several online servers has shown that players tend to play more often on servers that provide a method of assessing their performance. The servers providing more statistical information are consistently full whereas the others are often near empty.

- It allows server administrators to judge the fairness of gameplay on their servers. Servers known for fair gameplay always attract more players than other servers. This is because the newer players know they have just as much a chance of winning on either team, and older players know the challenge level will remain consistent.

- It can aid players in choosing servers that better fit their abilities. If the difficulty of a server can also be judged, then players can choose the servers that best fit their abilities. The following paper provides a model for comparing the difficulty level of a given server to other servers.

In the end, the games and servers that provide better statistical measures in addition to more challenging and fair gameplay will attract more players.

The game chosen for this paper is called Wolfenstein: Enemy Territory, also known as ET. It is one of the most popular team-based online games played with several hundred thousand unique players playing every day. Its popularity is partly due to the fact that it was made free by its publisher, Activision. In Enemy Territory, each match consists of an Axis side and an Allies side competing on a World War II historically inspired map. Each team has several objectives they need to complete within a certain time limit in order to win. Usually, one team's objective is to accomplish a given goal, while the other team's objective is to prevent them from accomplishing this goal during the time limit. This often unbalanced play-style, combined

with the common coming and going of players on a public server, creates matches that require rich models in order to predict. They can be compared to a soccer game where either team can have as many players as they like so long as the combined number of players on the field is less than some maximum. To complicate matters further, the players are free to switch between teams at will. Furthermore, the soccer field does not have to be symmetrical: the field can be on an incline such that the ball naturally rolls towards one team's goal, and it can also include obstacles on one end that are not on the other. The pool of possible players also does not have a limit, ranging as high as 1,000, and the players are free to leave and join a match at any time. There are no other published models designed to handle this complex of a competition. The following presents a model to estimate individual ratings from team-play and account for both the imbalance in the play-style, and the dynamic nature of the teams.

With the model presented here, we are able to answer several questions, including, "Who are the best players on this server?", "Which maps are the most difficult for the Axis side?", and "Which server is easier to play on for the average player?", in addition to questions like "How likely are the Allies to win the current match on this server?"

While the model in this paper is first fit using MCMC, we also need a method that can estimate the ratings of up to millions of players on thousands of servers in real-time in order to give up-to-date statistics to the players. It takes longer to fit the model with MCMC than the length of a single match in most online games, and therefore MCMC would be inappropriate in a real-time setting. This paper presents a recursive updating method that gives a group generalization of that proposed by Glickman (1999) for quickly estimating the ratings of large amounts of players across large amounts of servers. In order to be practical, any model for ranking individual players from team competitions needs to be less demanding in both its memory requirements and speed. We show that the recursive method we propose is fast enough to fit the data in real-time, whereas MCMC is not.

This paper proceeds as follows. Section 1.1 gives related work, section 2 describes the data analyzed, the model is presented in section 3, section 4 explains the how the model parameters are estimated using MCMC, section 5 presents the less time intensive recursive approximation for fitting the model along with results and comparisons to MCMC, potential applications and uses for real-time estimation of rating players are shown in section 6, and section 7 gives conclusions and directions for future work.

1.1 Related Work

Statistically motivated rating systems are not new. One of the earliest examples is that of Elo (1978) for chess ratings. He used the equivalent of a *Thurstone Case V* model which assumed chess competitors had a rating score that followed a normal distribution. This model also assumed a normal distribution on the difference in player ratings. More recent versions of Elo's chess rating system have adopted a logistic distribution instead of a normal distribution because it has been shown that "weaker players have significantly greater winning chances" (Wikipedia, 2006). The logistic distribution version of Elo's model is equivalent to the model proposed by Bradley and Terry (1952) for paired-comparisons.

The basic model used in this paper is an extension of the commonly used Bradley-Terry model. In this model, two opponents have ability parameters λ_1 and λ_2, and the probability of the first opponent winning is $\lambda_1/(\lambda_1+\lambda_2)$. Several reviews on uses of and extensions to Bradley-Terry models can be found in the literature (Bradley and Terry, 1952; Davidson and Farquhar, 1976; David, 1988; Hunter, 2004). One common extension proposed by Agresti (1988) is to add a parameter to the model for home field advantage. The following uses a similar approach, adding a parameter to model the advantage or disadvantage of playing on the Allies side. In addition, Glickman (1999, 2001) recently extended the Bradley-Terry chess rating model to take into account uncertainty about a player's rating based on the amount of time that has passed since a player has competed.

What makes this work unique is that it presents a new model for estimating individual ratings from team competitions. Few other models have been presented for this situation. One notable model is that of Huang et al. (2006b). They use Bradley and Terry's original formulation of the model, but obtain each team's λ by summing the ratings of its players: $\lambda_1 = \sum_{i \in 1} p_i$, where p_i is the rating of player i. This model fit well in their application which never had uneven amounts of individuals per group. It results in a linear increase in team rating when there are more players on one team than another. Experience has shown us, however, that having more players on one team does not result in a linear, but rather exponential increase in the team's perceived ability to win for our application. Another similarity between our model and that of Huang et al. (2006b) is that we both present a method for weighting individuals by how much they are expected to contribute to their respective teams. Huang et al. (2006b) suggest the weighting be predetermined using expert knowledge about the data, whereas we use a simple exposure model that uses the percent time a player plays on a given team for a given match as that player's "weight" for that match. In a more recent paper, Huang et al. (2006a), use the same basic model as that presented here, however it is posed and fit in a maximum likelihood instead of Bayesian framework.

Another recently proposed model that estimates individual ratings from team competitions is that of Herbrich et al. (2007). Their ratings system is similar to the Thurstone Case V model given by Elo (1978) except instead of comparing two individuals, it compares two groups by summing the ratings of the individuals. Since the likelihood they use is a normal, unlike the model given by Huang et al. (2006b), team ratings increase exponentially when one team has more players than another. The main difference in the model presented here is that we assume a logistic instead of normal distribution on the difference in team strengths. This results in our model being an extension of the Bradley-Terry model, whereas theirs is an extension of the Thurstone Case V model. We made this decision because we felt weaker teams were more likely to win than the normal distribution would predict. In addition, the model presented here infers other game characteristics besides player skills that are also beneficial to both match-making and game design.

The model used in this paper is first fit using MCMC, but MCMC is impractical for online gaming because it takes longer to fit the model than it does to play a match. Therefore, the estimates after a new match would already be inaccurate by the time they were available. A recursive method that generalizes the work in Glickman (1999) to individuals in groups is given for estimating the parameters of the model efficiently enough for real-time use. Other methods for quickly fitting Bradley-Terry models have been proposed by Elo (1978); Hunter (2004); Glickman (1995, 1999, 2001). The method here is similar to that proposed by Elo as given in Elo (1978) and those suggested by Glickman (1999, 2001), however these methods do not naturally extend to estimate individual ratings from groups. Generalizing the method in Glickman (1999) retains the ability to model the uncertainty in a player's rating unlike that in (Elo, 1978). This method is detailed in section 5.3.

In addition to providing ratings for individual players from group competitions, the model proposed by Herbrich et al. (2007) also provides an efficient method for fitting the ratings in real-time. Their method uses a form of normal density filtering which minimizes the Kullback-Liebler divergence between the estimated and actual distributions. The main difference is the method described here fits the mean and variance of a normal distribution at the mode, whereas the method in Herbrich et al. (2007) fits it at the mean. Section 4 shows that the posterior of the player ratings follows a normal distribution, and therefore the quality of estimating at either the mean or mode should be similar.

2 Data

We obtained data by parsing the log files from three Enemy Territory servers yielding 4,675 matches. For each match, the log files list which server the match was played on, which map was played, which side won—Axis or Allies—the length of the match, which players participated, and how long each player spent on both teams. Here is an example of a part of log file entry:

```
Map: fueldump Winner: AXIS Time: 1800000
Name: Player1 GUID DFBB5: Time Axis: 0 Time Allies: 1450200
Name: Player2 GUID EF071: Time Axis: 1549800 Time Allies: 0
```

The first line, or "Map" line gives which map or field the match was played on, the side that won the match, and how long the match lasted in milliseconds. In this match, the Axis side won and the match lasted 1,800,000 milliseconds, or 30 minutes. The remaining "Player" lines give information about each player. First, a name is given, which in this case we have anonymized. These are usually nick names the players choose and often change. In order to track players despite changing names, the next field is used. A player's GUID or *Globally Unique IDentifier* is a 32-digit hexadecimal field that is unique across all servers for Enemy Territory and assigned by a central server for every player everywhere in the world. The GUIDs presented here have been shortened to 5 digits for both privacy and brevity. The player records are stored based on this instead of their names which can change. After the GUID, the last two fields give the amount of time each player spent on the Axis or Allies team in milliseconds. For example, Player1, the first given player, spent 1,450,200 milliseconds, or 24 of the matches' 30 minutes on the Allies side, which lost. Player2, the second player from the top, spent 1,549,800 milliseconds, or almost 26 of the matches' 30 minutes on the winning Axis side. Notice that neither player spent the entire 30 minutes of the match in the game. In addition, here is an example of a player in a different match who spent time on both sides:

```
Name: Player3 GUID 6C875: Time Axis: 552600 Time Allies: 1336200
```

Here, Player3 spends 9 minutes on the Axis team, and then 22 minutes on Allies team. It is not uncommon to see players that do not play the entire match on the same team and players who do not participate for the entire match in online matches like those in Enemy Territory. This happens for several reasons, including players joining the server late in a match, and players switching teams to play with closer acquaintances. Therefore, we present a method for handling this dynamic behavior in section 3.

3 Models

The following shows how we used the given data to estimate player ratings. We present a model for individual player ratings in section 3.1, an extension for handling map-side effects in 3.2, and another extension for server difficulty in section 3.3. Section 3.4 gives the likelihood for predicting whether a given team will win a match given the data above.

3.1 Basic Model

Although the outcome of each competition is measured at the team- or side-level, the fundamental unit of the rating system needs to be built on the abilities of the individual players themselves. The likelihood given in section 3.4 determines a team's rating from the players. Therefore, we need a model for the individual player ratings. We let θ_i represent player i's ability to help their side win a match. There are no existing standards for rating players in these games and so for simplicity we chose a model that places player ratings symmetrically around 0. The better players will have positive player ratings, and the worse players negative ratings. This is in contrast to the ratings in, for example, chess, that have values ranging closer to 1500–2500. Our model for player ability is:

$$\theta_i \sim N(\mu, \sigma_\theta^2). \tag{1}$$

σ_θ^2 is given a prior distribution and $\mu = 0$ without loss of generality. This model could be used without modification with the likelihood in 3.4, however that would make the naive assumption that a player's probability of winning is not affected by the difficulty of the side that player plays on for a given map.

3.2 Accounting for Map-Side Effects

Because maps in Enemy Territory and most other games of this nature are so varied, it is convenient to describe the details of a given match in terms of the combination of map and side that each player played on. Throughout this paper, we refer to a team or a player's "map-side" combination, meaning the map the match took place on, e.g. the map "venice", and the side the player played on—Axis or Allies. Without accounting for a "map-side" effect, the model given above would be naive considering most maps were designed such that one side has a major advantage. Since this imbalance is uniform by design for all players, a more robust model is to assume that the ability of one side to win increases or decreases according to the map being played on. In this model, we let $\Theta_{L,j}$ represent the Allies team's ability to win a match played on map j given the set of players L (using L for Allies and X for Axis). As with player ratings, we choose a model

that places map-side ratings symmetrically around 0:

$$\Theta_{L,j} \equiv \sum_{i \in L} \theta_i + \gamma_j \text{ with } \theta_i \sim N(0, \sigma_\theta^2) \text{ and } \gamma_j \sim N(0, \sigma_\gamma^2). \tag{2}$$

This is similar to fitting the "homefield advantage parameter" proposed by Agresti (1988), giving one γ_j parameter per "field" for the Allies. When γ_j is positive, map j favors the Allies, when negative, the Axis. Although this does not model individual player-map-side interactions, it is assumed that the additive effect will be stronger than the individual interaction effects because most maps have been designed to be uniformly uneven. We are more interested in increasing the predictive power of the ratings than in the interactions themselves, and therefore we chose not to model them at this point. Since we have modeled the "map-side" advantage with respect to the Allies team, we will often also refer to it as the "Allies bias".

With this model, a team's ability on a current map becomes the sum of the player abilities $\sum_{i \in L}$ or $_X \theta_i$, offset by a map-side effect, γ_L or $_{X,j}$. This information is interesting because it suggests that a team of skilled players can still enjoy a reasonable level of challenge against a team of less-skilled players as long as the map-side effect gives the less skilled team an equivalent advantage. In addition, the estimates for γ_j can be used to judge which maps are more balanced between Axis and Allies. Server administrators can use this information to choose which maps to place on their servers to improve gameplay, and map creators can use this information to better balance the gameplay in their maps.

3.3 Server Difficulty

If players can be ranked based on matches they participated in on a given server, it would be beneficial to also be able to compare players participating on different servers, or players that played on several servers. In order to determine how a given server affects a player's rating, the following model adds a server bias, ψ_k, into each player's rating, yielding:

$$\theta_{i,j} \equiv \theta_i + \psi_k \text{ with } \theta_i \sim N(0, \sigma_\theta^2) \text{ and } \psi_k \sim N(0, \sigma_\psi^2). \tag{3}$$

Note that the server difficulty is modeled as an increase instead of a decrease in a player's rating. This means that we are really modeling server "easiness." We did this for simplicity in the model. It means that lower and not higher values of ψ_k denote a more difficult server. With this model, a player's rating depends on their base ability and the server they are playing on. Besides being able to compare players across servers

more effectively, estimating ψ_k also gives us a measure of how difficult each server is. This information can be used by newer players to choose easier servers, or more experienced players can use it to find challenging gameplay. In addition, since the player ratings can now be fit globally over all servers, a player can compare his or herself to all of the players in the game, instead of only those who play on a certain server. The accuracy of the comparison, however, will be affected by how often players move between servers. If enough server "cross-over" occurs, then the ratings should be reasonable enough for comparison.

Another interesting aspect of ψ is since the amount added to each side s is equal to $|L$ or $X|\psi_k$, ψ_k can be interpreted as the rating increase the side expects for each additional player on that side. Servers where having additional players will not make up for the skill of the players are more difficult than those where a few extra players alone can decide the winner. This fact can be used to represent the ability of, for example, the Allies team to win as follows:

$$\Theta_{L,j,k} = \sum_{i \in L} \theta_i + \gamma_j + (|L| - |X|)\psi_k \text{ with } \theta_i \sim N(0, \sigma_\theta^2),\ \gamma_j \sim N(0, \sigma_\gamma^2),\ \text{and },\psi_k \sim N(0, \sigma_\psi^2). \tag{4}$$

If the ability of the Allies team is expressed this way, the Axis team's ability can be expressed without loss of generality as the sum of the player skills on the Axis team:

$$\Theta_X = \sum_{i \in X} \theta_i \text{ with } \theta_i \sim N(0, \sigma_\theta^2). \tag{5}$$

This is because the Allies ability already accounts for both the map-side effect γ_j and the server effect ψ_k.

3.4 Likelihood

The likelihood we use is based on a modified version of the Bradley-Terry paired-comparison model (Bradley and Terry, 1952). Without loss of generality, we choose a likelihood that predicts whether or not the Allies team will win. The Allies's probability of winning a match played on map j for server k is chosen to be proportional to $\lambda_{L,j,k} = exp(\Theta_L)$ using equation 4. Since map and server effects are accounted for in the Allies bias, the Axis' probability of winning is chosen to be proportional to $\lambda_X = exp(\Theta_X)$. Therefore the probability of the Allies winning on map j and server k is $\lambda_{L,j,k}/(\lambda_{L,j,k} + \lambda_X)$. Using equations 4 and 5, this results in the probability of a given side winning increasing exponentially with the size of that side compared to the other. This is appealing because unlike a game like chess where only one move can be made per side despite the number of actual players making the move decisions, in Enemy Territory and other real-time

online team competitions, every player can act simultaneously. Having more players gives a team with more numbers the ability to more effectively defend or accomplish its objectives. It is rare for a team with two or less players than another to win unless that team is very good.

If G is the total number of matches, L_g and X_g represents the Allies and Axis teams in match g respectively, w is 1 for an Allies win and -1 for an Axis win, j the map played on, and k the server played on, this model gives rise to the following likelihood function:

$$P(w|\lambda) = \prod_{g=1}^{G} \lambda_{L_g,j,k}(\lambda_{L_g,j,k} + \lambda_{X_g})^{-1} \tag{6}$$

where

$$\lambda_{L,j,k} = exp(w\Theta_{L,j,k}), \tag{7}$$

and

$$\lambda_X = exp(w\Theta_X) \tag{8}$$

using $\Theta_{L,j,k}$ and Θ_X as given in equations 4 and 5.

One problem with the above likelihood is that in public-style servers players can come, leave, and change teams almost at will. Therefore, a model that does not take into account how much time a player spends on each team is inaccurate. Since the match data we are using does indicate the amount of time the players spent per team, this information can be used to modify equations 4 and 5 as follows:

$$\Theta_{L,j,k} = \sum_{i\in L} \tau_{i,L}\theta_i + \gamma_j + (\sum_{i\in L} \tau_{i,l} - \sum_{i\in X} \tau_{i,X})\psi_k \text{ with } \theta_i \sim N(0,\sigma_\theta^2),\ \gamma_j \sim N(0,\sigma_\gamma^2), \tag{9}$$

$$\text{and },\psi_k \sim N(0,\sigma_\psi^2) \tag{10}$$

$$\Theta_X = \sum_{i\in X} \tau_{i,X}\theta_i \text{ with } \theta_i \sim N(0,\sigma_\theta^2) \tag{11}$$

where $\tau_{i,L}$ ($\tau_{i,X}$) is the percent of the total match time player i spent on the Allies (Axis) team, L (X). This yields a simple approximation to integrating the player abilities over the time of the match to estimate the overall strengths λ per team. Notice that the server effect ψ_k is now weighted by the "effective" number of players on each team throughout the match, instead of the difference in all players that participated in the map.

4 MCMC Analysis strategies

This section will discuss how the priors were chosen, how the model was fit, and how the convergence of the fit was verified, all using MCMC.

4.1 Prior selection

Since the prior means for the parameters are all chosen to be 0 without loss of generality, the remaining priors to choose are on the variances of the player ratings, σ_θ^2, the map-side ratings σ_γ^2, and the server ratings σ_ψ^2. Instead of choosing non-informative priors for these variances, we placed hyperprior distributions on them using inverse gamma distributions, with α and β chosen such that the distributions have means of 1.0 and variances at 1/3. This was done simply to keep most player ratings between -3 and 3. Again, we chose priors this way because there are no current standards for rating players in these types of competitions, and so we decided for simplicity to assume they follow close to a standard normal distribution. We used hyperpriors so we could observe the relative differences in the standard deviations.

4.2 Software

Software to fit the model used in this paper was written in Python specifically for these analyses. It uses MCMC with Gibbs steps to sample from the standard deviations and Metropolis steps to sample from the player rating θs, the map-side effect or Allies-bias γs, and the server difficulty ψs. This leaves a potentially large set of tuning parameters, but following the suggestion of Graves et al. (2003), we chose separate constants for both player- and map-step sizes and divided them by the square root of the number of games a player had played in, or the number of times that map had been played. This resulted in acceptance rates of around 40% for the parameters fit using Metropolis steps.

4.3 Convergence Diagnostics

While it is difficult to guarantee an assessment of the mixing and convergence for MCMC, time series plots and acceptance rate analyses were used and the criteria in the diagnostics proposed by Raftery and Lewis (1996) were met. In addition, section 4.4 discusses the accuracy results of predicting with the estimated parameters.

4.4 Measuring Performance

One way to measure the performance of the given model is by its accuracy in predicting the matches used to estimate the model parameters. Unfortunately this positively biases the results. An unbiased estimate of the model's accuracy would require computationally intensive out-of-sample methods such as leave-one-out cross-validation. These methods would be impossible for real-time applications of the ratings because the amount of time it takes to fit the model using MCMC is longer than the length of the average match. Therefore, out-of-sample results will not be given using MCMC to fit the model, but instead are given in section 5 which provides a more efficient method to both fit and evaluate the given model. That said, the mean accuracy across the MCMC samples achieved near 78% accuracy in predicting the match data. This suggests that the ratings derived for the players are reasonably accurate over the given matches given the fact that "the smallest achievable prediction error could be as big as 50%" (Herbrich et al., 2007) in this type of problem.

4.5 Results

In this section we derive rankings for players based on information from 4,675 matches on three different Enemy Territory servers. Combining the servers allows us to rank the players globally, and allows us to compare the difficulty of the three servers using ψ. In addition, the difficulty of the maps with respect to which side a player chooses are analyzed.

4.5.1 Player and Server Ratings

To determine which were the best players over all three servers, we took advantage of the fact that many players play on more than a single server to fit the server difficulty parameter, ψ, across all three servers simultaneously. The model parameters were estimated using 100,000 iterations of MCMC with 10,000 iterations of burn-in. The results are shown in tables 1 and 2. Player names have been removed since they are not likely to be recognizable to the general public. The players are conservatively ranked by how they play two standard deviations below their posterior means. This has the desired effect of penalizing players whose ratings are less certain. The results meet the goals of giving each player a rating and hence a stronger incentive to continue playing and improving his or her abilities in the game. Although the names have been removed, the resulting rankings are reasonable given player opinions among those who play on the servers.

It is interesting to compare the actual winning percentage with the posterior means and rankings. For example, the 6th-and 8th-ranked players have higher winning percentages than any others in the top 10. In

theory, this suggests that although these player win more often, the matches these players win and lose are easier on average than the matches won and lost by the other players in the top 10. This is actually a benefit of the rating system, it allows us to adjust winning percentage ranking to account for match difficulty in terms of the other players, the map and side being played on, and the difficulty of the server.

The last column in the server table gives the percentage of times a smaller team wins against a larger team on that server. This is relevant recalling that the server difficulty parameter can be interpreted of as being inversely proportional to how often smaller teams win against larger teams. The server results are surprising at first because most players consider the settings on the third-ranked server to make it a more difficult server. However, the fact that the matches on this server are generally smaller because it is the least popular of the servers may contribute to it being easier overall for those players who play on multiple servers. Also, the ratings agree here with the actual win percentages observed when smaller teams play against larger teams. It is not as surprising to see the top-ranked server above the second-ranked server since the top-ranked server is the most popular server and attracts many of the best players. However, both of the top two servers have a similar configuration, and therefore it is reasonable that their respective difficulties are relatively close. Here we see an example where players could use this information to choose a server. The better players could choose to play on the first two servers to enjoy more of a challenge, whereas newer players may be likely to choose the third server because it is easier. In addition, server administrators can use this information to change settings on their servers to increase or decrease the difficulty. The effects of the individual server settings and rules themselves can also be modeled to suggest ways to adjust them.

It is also worth noting that the shape of the resulting marginal posterior distributions for players, servers, and maps were all nearly indistinguishable from a normal distribution with the same mean and variance. An example of this can be seen in figure 1 which shows the posterior distribution for the first server along with a normal distribution with the same mean and variance.

4.5.2 Map-Side Effects or Allies Bias

One of the interesting effects of the model used is that rating parameters are also fit for each map-side combination. This information can be used to judge which maps are more even and which maps have the least balance between Axis and Allies. Table 3 shows the results of fitting the map-side or Allies-bias parameters across the three servers for two of the maps players competed on. In general, the results are in line with the perception of the players. The first map is well known for being biased towards the Allies. Even without accounting for player skill and server difficulty, the Allies still win 72% of the time. The second map

Table 1: Top 10 Ratings and rankings of players along with their posterior means, variances, and win percentages.

Rank	Post. Mean	SD	Actual Win %
1	1.71	0.32	0.76
2	1.63	0.31	0.72
3	1.55	0.29	0.68
4	1.63	0.35	0.72
5	1.49	0.33	0.72
6	1.57	0.38	0.90
7	1.65	0.44	0.69
8	1.66	0.46	0.91
9	1.99	0.63	0.60
10	1.39	0.33	0.74

Table 2: This table shows the servers ranked by difficulty, giving the most difficult first. Shown for each server is the posterior mean and the standard deviation. Recall that smaller not larger values denote more difficult servers. Also shown is the percentage of times a smaller teams defeats a larger team on the server.

Rank	Post. Mean	SD	Win% Smaller Team
1	1.44	0.10	0.22
2	1.63	0.14	0.22
3	2.45	0.30	0.14

is actually symmetrical in its construction, and therefore it is not surprising that it is more fair. The slight bias towards the Allies team despite the map being symmetrical is because the color of the Allies uniforms blends in well with the background of the map, whereas the Axis uniforms stand out more. This makes the Allies harder targets, and therefore biases the map towards the Allies despite its design.

From these findings, it is not unreasonable to conclude that, for example, the first map is imbalanced in favor of the Allies and that the second map is probably a fair map. Server administrators could decide to either not include maps like the first one in the list of maps on their server, or they can take measures to ensure that when this map is played, the Axis team consists of more skilled players. Map makers can use this information to consider whether they should make changes to the first map that would make it more even given equally skilled teams. As for the second map, the slight bias towards Allies despite the symmetrical construction of the map forces map designers to think deeper about what can affect gameplay. Other symmetrical maps have changed background color schemes that favor one uniform color over another in order to correct the imbalance. Games with statistically driven level design are more likely to attract players, as are servers that include more balanced maps in their lineups.

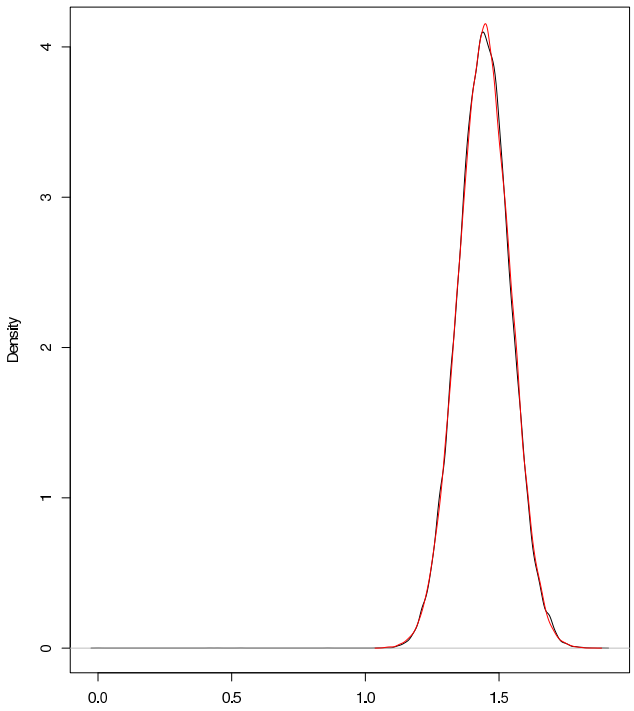

Figure 1: First server's posterior compared to a normal distribution with the same mean and variance. The server's distribution is shown in black and the normal in red.

Table 3: Allies bias examples.

Name	Post. Mean	SD	Actual Allies Win %
map1	1.99	0.24	0.72
map2	0.28	0.22	0.55

5 Real-Time Analysis strategies

5.1 Prior Selection

To simplify the proposed approximation method method, we set the priors on the variances of the player, map-side, and server difficulties to 1.0. Again, this results in ratings between -3 and 3, and we felt this reasonable since there are no established standards for computer game ratings. We did this before observing the hyperprior variances obtained using MCMC, which turned out to have means near 1.0, and therefore we do not expect differences between the estimation methods to result from choice of prior.

5.2 Software

The software used to fit the model using the following approximation was also written in Python specifically for these analyses.

5.3 Efficiently Estimating the Parameters

The method we use to efficiently approximate the marginal posterior of the given model parameters extends the method used in Glickman (1999) to handle three new elements:

1. Estimating the ratings of individuals competing in groups instead of just one-on-one competitions between individuals.

2. Dealing with the fact that players come, go, and change teams commonly throughout a given competition.

3. Additional non-player parameters, including a "map-side" bias and a "server difficulty" bias.

The method in Glickman (1999) can be used to update the marginal posterior ratings of the participating players, the Allies bias for the given map, and the server difficulty after each competition. The goal is to take a given prior mean, $\theta_{i,(t-1)}$ (initialized to 0) and prior variance $\sigma_{i,(t-1)}$ (initialized to 1) where i refers to a given player's rating, a given map's Allies-bias, or a given server's difficulty at time $t-1$ before a competition, and update both to obtain the marginal posterior mean $\theta_{i,t}$ and variance $\sigma_{i,t}$ at a time t after a given competition. Notice that like Glickman, we are not estimating the entire covariance matrix, but only keeping the diagonal elements. This is for mostly to save on both time and space computationally, but we felt it reasonable because the number of times two given players participate in the same match is small

compared to the overall large population of players. Extending the update equations in Glickman (1999) to account for teams, maps, and servers yields:

$$\sigma_{i,t} = (\sigma_{i,t-1}^{-1} + g_{i\notin x}(x^T Dx)^2 P(w)(1-P(w))x_i^2)^{-1} \tag{12}$$

$$\theta_{i,t} = \theta_{i,t-1} + \sigma_{i,t} g_{i\notin x}(x^T Dx)(1-P(w)) * w \tag{13}$$

$$\tag{14}$$

where

$$P(w) = \frac{1}{1+exp(-w g_{i\notin x}(x^T Dx)(\Theta_L - \Theta_X))} \tag{15}$$

$$\tag{16}$$

and

$$g(\sigma^2) = (\sqrt{1+3\sigma^2/\pi^2})^{-1}. \tag{17}$$

Recalling from section 3.4 that $w = 1$ for an Allies win, and -1 for an Axis win, $P(w)$ represents the probability of the given outcome. The form of $P(w)$ is equivalent to that given in 6 except that it also includes $g(\sigma^2)$ which will be explained shortly. x is a column vector with an entry for each parameter participating in the match that took place at time t. For a given player i, a corresponding element in x, x_i, will be set to $\tau_{i,L} - \tau_{i,X}$, or the net time the player spent on the Allies team. There will also be an element set to 1 in x for the Allies bias γ_j for the given map, and finally an element set to $\sum_{i\in L} \tau_{i,L} - \sum_{i\in X} \tau_{i,X}$, or the net advantage in player numbers on the Allies team. This element corresponds to ψ_k for the given server. D represents the diagonalized covariance matrix including only the parameters participating in the given match. The quantity $g(\sigma^2)$ is used to match the first and second moments of the cumulative logistic distribution to that of the cumulative normal or probit distribution in order to approximate a closed form solution to the marginal integral. Notice also that g is subscripted with $i \neq x$, this is because, as in Glickman (1999), when updating the marginal posterior of a given parameter i, that parameter's own variance is not included and therefore the x and D within $g(\sigma^2)$s will not include the mean or variance for parameter i. See Glickman (1999) for the full derivation of the estimation procedure.

5.4 Results

In this section we first discuss the complexity of the approximation method compared to MCMC in section 5.4.1. We then derive rankings for players based on match information on three different Enemy Territory servers in section 5.4.2. The map-side effects are analyzed in section 5.4.3 and section 5.4.4 reports on the accuracy of the approximation method.

5.4.1 Complexity Comparison

The main reason for choosing the method in section 5.3 instead of the MCMC is that MCMC takes longer to fit the data than the length of the average match in Enemy Territory, which is 15 minutes. Here we give a brief overview of the complexity of both MCMC and the approximation method instead of an in-depth analysis because the difference is so pronounced.

The complexity of MCMC for this application is proportional to NPM where N is the number of iterations chosen for the chains, P is the number of players who play over all the matches, and M is the average number of matches per player. The data set used consisted of 4,675 matches, 5,145 players, and 14 matches on average per player. Fitting this data using 100,000 iterations of MCMC took several days on a 2.7 Ghz Core 2 Quad Extreme. This is longer than the average length of a match in Enemy Territory. The length of time it takes to fit the model using MCMC is too long to be used in a real-time application.

The complexity of the approximation method we give in section 5.3 is proportional to the number of players in a given match—in Enemy Territory this never exceeds 64—therefore the update time is at worst a small constant and takes less than 1 second to process on the same Core 2 Quad Extreme machine. Since the complexity of this method scales only linearly with the size of the number of competitors, it can be extended for use in applications that are significantly larger than this one without a major increase in running time. This method can be applied real-time because it can update the model before the next match completes. The comparison may not seem completely fair because we are comparing the update from a single match to fitting the entire data set as is done in MCMC. However, using MCMC would *require* refitting on the entire data set after every match. That said, the amount of time it takes to fit the larger data set used in this paper with the more efficient method is under one minute on the same 2.7 Ghz machine. This is compared to the aforementioned several days using MCMC. Therefore, this method is significantly faster than MCMC and is fast enough to be used in real-time.

Table 4: Ratings, rankings, and comparisons to MCMC for the approximate estimation method.

Rank	MCMC Rank	Post. Mean	MCMC Post. Mean	SD	MCMC SD	Actual Winning %
1	1	1.66	1.71	0.32	0.32	0.76
2	8	1.92	1.66	0.45	0.46	0.74
3	14	2.33	2.00	0.68	0.67	0.91
4	3	1.48	1.55	0.31	0.29	0.68
5	11	1.39	1.39	0.33	0.33	0.74
6	7	1.50	1.57	0.40	0.38	0.69
7	4	1.36	1.63	0.34	0.35	0.72
8	30	1.24	1.06	0.29	0.29	0.62
9	9	1.49	1.65	0.45	0.44	0.60
10	2	1.26	1.63	0.31	0.31	0.72

5.4.2 Player Rankings

In this section we present the top ten rankings of the players for each server as fit by the method given in 5.3. The model was fit using the same 4,675 matches from each of the three servers as used in the MCMC estimation. The approximation method given in the previous section was used to update the participating parameters after each match. By participating parameters we mean those corresponding to the players, map-sides, and server for that match. The results are shown in table 4 with the ranks, means, and standard deviations estimated by MCMC shown side-by-side for comparison. The players are again conservatively ranked by how they play two standard deviations below their posterior means. Once again, it is not uncommon in these tables to see players who have worse winning percentages ranked higher than those with better ones. This suggests that their wins were against more difficult odds—e.g. harder map-side combinations, better players, etc.

It appears that the estimation method performed well with respect to MCMC, especially in approximating the variance. The means show the most difference and lead to some permutations in the ranking, but they are within the expected ranges given by the parameters' variances. The differences could have been caused by the normal–logistic approximation or by the fact the covariance matrix was diagonalized. Overall, however, the estimation method is reasonable given the results.

Table 5 gives results of estimating the server difficulty with the approximation method, along with comparisons to MCMC. The ranking is the same, however the approximation method appears to have overestimated the means. The variances are, as with the players, close to their MCMC estimates. The advantage of using the approximation method, however, is that these server estimates can be updated after each match occurring on each server. Therefore, players can have instant up-to-date information about the

Table 5: This table shows the servers ranked by difficulty, giving the most difficult first. Also shown for each server is the posterior mean, the standard deviation, side-by-side MCMC results, and the percentage of times a smaller team wins on that server.

Rank	Post. Mean	MCMC Post. Mean	SD	MCMC SD	Smaller Team Win %
1	1.96	1.44	0.09	0.10	0.22
2	2.14	1.63	0.14	0.14	0.22
3	2.74	2.45	0.27	0.30	0.14

Table 6: Approximated Allies bias compared to MCMC estimates.

Name	Post. Mean	MCMC Post. Mean	SD	MCMC SD	Actual Allies Win %
map1	1.93	1.99	0.25	0.24	0.72
map2	0.59	0.28	0.24	0.22	0.55

difficulty of the servers they are choosing from. The method we use to estimate these parameters is efficient enough to keep up with millions of players and thousands of servers, and therefore it can be applied to any one of the popular games that are already out or scheduled for release in the near future.

5.4.3 Map-Side Effects or Allies Bias

This section compares the results of approximating the Allies bias parameter γ to the estimates given by MCMC for the same maps. Table 6 shows the results and comparisons with MCMC. The approximations for the first map are close in both mean and variance, whereas the second in variance. The mean of the second map is just over twice the MCMC estimate, but still reasonably near 0 given the map is symmetrical in design. Again, the advantage here is these approximations are close enough to be used in making judgements about the maps, and available in near real-time after each match. Because the method we use to estimate the map ratings is efficient enough to run real-time, server administrators can use the map rating information to judge the fairness of a current match in progress, and make changes to improve balance. More on this is discussed in section 6.

5.4.4 Measuring Performance

Here we attempt to obtain a less-biased measure of the performance of the model whereas in section 4.4 the accuracy given is biased. Like section 4.4, the model's accuracy is tested against predicting the matches used to estimate the model parameters. In order to remain unbiased, the prediction measurement for a given match is taken *before* updating the model based on that match. This is not completely unbiased, and in

fact results in a slight negative bias because the model parameters used for prediction will not have been approximated well for the first group of matches. However this also represents how the model is updated and used in the real-world. The results show the accuracy to be 72.5%. Given the complexity of these diverse online matches, an unbiased, first-pass accuracy of over 70% is significant. Especially considering the aforementioned fact that the smallest error may be as large as 50% on a given match. In addition, 70% is comparable to the results obtained by Herbrich et al. (2007) on an application with less complexity than the one used here.

Another, more ad hoc performance measure is to simply verify that the values of the parameters and resulting rankings follow the intuitive sense of those familiar with the servers, maps, and players. Acting as those who have experience with these maps, servers, and players, we agree that, in general, the rankings fit.

6 Applications

This section describes potential applications for the suggested model and parameter estimation method.

6.1 Ranking the Players

Our first application is the obvious one, using the ratings to rank the players on the servers and across the servers. Players tend to prefer servers that give them a ranking they can work to improve. As stated, MCMC is too computationally intensive to be used real-time in situations involving as little as thousands of players on only three servers, let alone millions of players and matches. This is because the amount of time it takes MCMC to fit the data is usually longer than the length of a single match.

The approximation method we used, however, is efficient enough to provide real-time rankings and ratings to players across multiple servers. These ratings and rankings can be updated quickly after every match, and made available to the players for comparisons. For example, for the servers that we run, players can go to web pages to see listings of their ratings and rankings and compare them to other players on the servers. In addition, we provide methods for players to view their ratings and rankings with in-game commands. The players tend to appreciate these features. For example, one of our servers provides these commands and is almost consistently full at 25-32 players, whereas another server where we do not is almost always empty. Having the ability to query a player's rating information in-game is probably not the only factor contributing to the other server's lower popularity, but it is worth noting.

Because we model server difficulty as well as player and map-side ratings, players can compare themselves

across servers. This assumes there are enough players that play on more than one server to make comparisons meaningful. For example, if *none* of the players who play on the third server ever play on any other server, their ratings and their server's difficulty rating can not be correctly compared to other players and servers.

Since the writing of this paper, a system that posts world-wide rankings to players who play Enemy Territory has been added to the game and made publicly available Joshua E. Menke (2007). To date, over 200 server administrators have enabled this feature in their game, resulting in player ratings based on several hundred thousand players and matches. It also includes Allies bias ratings on several hundred different maps. The running accuracy of this global system in predicting the winner of a new match is currently at 74%. Player ratings are periodically removed for inactivity, so the current player count posted only represents active players, and not the total. The number of matches is, however, accurate (currently 354,967).

6.2 Choosing Servers

Since we fit a server "difficulty" parameter with ψ per server, we can use that parameter to rank the servers in order of difficulty. The parameter estimation method we used is efficient enough to update ψ for a large amount of servers in real-time after each match. Games that have this information available real-time can make it available to players who are trying to choose which servers to play on. Newer players can choose the easier servers, and veterans can choose servers that will give them more of a challenge. Also, because player ratings are fit taking into account the difficulty of the servers they play on, the servers can be also be listed in order of the average player ratings of the players *currently* playing. This measure gives an even better estimate of the current state of the server because it represents skill of the players in the current match, instead of the average difficulty. Players can sort the servers by the average rating of the players on them, and then choose servers with ratings to suit the desired challenge level. As a more advanced option, games can be designed to automatically choose servers that best fit a given player. This can ensure that players always have a positive experience in playing these complex team-based games.

Besides a more consistent and balanced gameplay experience, giving players the ability to choose servers based on difficulty gives them another incentive to continue playing. Players will start on easier servers with the goal of improving their skills to a point they can comfortably play on harder servers. This natural improvement path will encourage players to return again and again to a game to see if they are good enough to play on harder and harder servers. One of the most popular online game types is the MMORPG or Massively Multiplayer Online Role-Playing games. The most popular of these games, World of Warcraft, boasts a player base of more than 8 million (Woodcock, 2006). The draw to the MMORPG is that the

game is designed with explicit character development paths that give players incentives to continue playing. However, these paths can be very time-consuming to pursue, especially for the more casual gamers that play in games like Enemy Territory. The server rating system can supply a character development path for the casual gamer based solely on the ratings system. This can enlarge the audience for first-person shooters and encourage players to buy games with this feature, and continue playing them.

6.3 Balancing Teams

At the server level, administrators can use real-time player rating information to balance the teams currently playing. To make this easier, the game can use the approximated marginal posterior ratings of the players to give posterior prediction estimates on which team is likely to win. If a team is very unlikely to win the match based on this real-time information, administrators can move better players to the disadvantaged team, and / or move worse players to the favored team. This can also be done automatically. If a team falls below a specified probability of winning, say 30%, then the game can automatically move players to bring the probability as close to 50% as possible. A computer can easily try all possible moves involving one player and in such a way greedily optimize the probabilities around 50%. Servers that employ automatic team balancing will enjoy a consistently balanced level of gameplay and attract more players. Games that have this option available may be more popular than those that do not.

In addition to making the game more enjoyable for its players, balancing teams in this manner can improve the efficiency of rating the players. As Herbrich et al. (2007) observed, "the [team balancing] process can be viewed as a process of sequential experimental design" (Fedorov, 1972). This is appealing and "since the quality of a match is determined by the unpredictability of its outcome, the goals of [balancing teams] and finding the most informative matches are aligned!" (Herbrich et al., 2007). If the data used to fit the model comes from matches where a form of team balancing has been employed, less data may be needed to achieve a more accurate fit of the model.

One important question when choosing a limit for automatic team balancing is whether or not the predictions are precise around that threshold. It is important that teams with, for example, a 70% chance of winning actually do win 70% of the time. We can report that, in our experiments, we found that when the prediction for a team winning was above 70%, that team did in fact win 70% of the time. Therefore, 70% seems to be a good value as a cut-off if more than a 70% winning chance is considered unfair.

7 Conclusion and Future Work

We have presented a new model for estimating individual ratings in team competitions. These ratings allow players to effectively track their ability to help the teams they play on win. Games and servers that provide well-developed methods for tracking their players' abilities are more likely to attract players and therefore be more profitable. The model presented is also able to account for both the dynamic nature of the teams in public, online team-competitions, and the imbalance commonly associated with the levels or maps designed for these competitions. In addition, we have estimated parameters that allow us to analyze the fairness of the maps in terms of the sides playing. This information can be use to create maps that are more fair and therefore more enjoyable for the players, or the information can be used by server administrators to choose different maps for their respective servers.

We have also fit a parameter that allows us to estimate the difficulty of each of a set of servers. This information can be used to allow players to select servers that better match their abilities. Players are more likely to continue playing a game if they can ensure matches will not be too easy or too difficult. The model developed in this paper can be directly applied to modern-day online team-competitions with results that will likely improve gameplay and attract players.

In addition, we have presented an efficient method for estimating the parameters of this model so that they can be used to provide real-time information about individual player rankings, server difficulty, and the fairness of the current match on a given server. MCMC is not efficient enough to do this real-time. We have given applications that can use this information, including suggesting games allow players to choose which servers to play on based on server difficulty or the average rating of the players currently playing on a given server. We have also suggested that the information be used to automatically balance current matches by moving players between teams. Games and servers that provide well-developed methods for both tracking their players' abilities and ensuring balanced gameplay are more likely to attract players and therefore be more profitable.

Future work will first evaluate the model that the estimation method is used to fit. For instance, instead of assuming player ratings are stationary, we could model time-varying changes in player abilities. Glickman (2001) suggested a model to do this for rating and ranking chess players.

It would also be interesting to model the effect of the number of players on how hard a given map-side combination is. There may be maps that are easier for a given side when there are fewer players, but become harder as the total number of players increases. In this model, γ_j would be the result of a regression fit instead of a single, learned constant. Being able to analyze the effects of the number of players on a

maps fairness will allow server administrators to choose maps more appropriate for the number of players commonly on their servers, again leading to increases in server usage.

In addition to improving the model, the estimation method could also be improved. One way to do this would be to retain the entire covariance matrix, especially if the number of parameters is not prohibitively large. This may narrow the differences between the estimates of the means given by both approximation method and MCMC. To save time computationally, only those parts of the covariance matrix related to the given match would be used in an update. For applications where the number of parameters is too large, modeling assumptions could be made that only tracked covariances between pre-determined parameters. This would be chosen based on prior knowledge of the application.

Finally, we will investigate non-parametric approaches to modeling player ratings and match-making. One simple approach can be seen by viewing it as a method for fitting a simple *artificial neural network* (ANN) as shown in Menke and Martinez (2007). This ANN model can be extended to multiple layers, thus yielding a generalized non-parametric and non-linear model. In order to preserve individual ratings, the non-linear portion of the ANN can be incorporated independently from the original. This non-parametric model may detect and predict arbitrary higher-level interactions and non-linear dependencies not accounted for by the given model.

References

Agresti, A. (1988), *Categorical Data Analysis*, Oxford University Press.

Bradley, R. and Terry, M. (1952), "The Rank Analysis of Incomplete Block Designs I: The Method of Paired Comparisions," *Biometrika*, 39, 324–345.

Crandall, R. W. and Sidakk, J. G. (2006), "Video Games: Serious Business for America's Economy," .

David, H. A. (1988), *The method of paired comparisons*, Oxford University Press.

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

DFC Intelligence (2006), "Online Game Market," .

Elo, A. E. (1978), *The rating of chess players: Past and Present*, Arco Publishing, New York.

Fedorov, V. (1972), *Theory of Optimal Experiments*, Academic Press, New York.

Glickman, M. E. (1995), "A Comprehensive Guide to Chess Ratings," *American Chess Journal*, 3, 59–102, submitted.

— (1999), "Parameter Estimation in Large Dynamic Paired Comparison Experiments," *Applied Statistics*, 48, 377–394.

— (2001), "Dynamic paired comparison models with stochastic variances," *Journal of Applied Statistics*, 28, 673.

Graves, T., Reese, C., and Fitzgerald, M. (2003), "Hierarchical Models for Permutations: Analysis of Auto Racing Results," *Journal of the American Statistical Association*, 98, 282–291.

Herbrich, R., Minka, T., and Graepel, T. (2007), "TrueSkill(TM): A Bayesian Skill Rating System," in *Advances in Neural Information Processing Systems 19*, eds. Schölkopf, B., Platt, J., and Hoffman, T., Cambridge, MA: MIT Press, pp. 569–576.

Huang, T.-K., Lin, C.-J., and Weng, R. C. (2006a), "Ranking individuals by group comparisons," in *ICML '06: Proceedings of the 23rd international conference on Machine learning*, New York, NY, USA: ACM Press, pp. 425–432.

Huang, T.-K., Weng, R. C., and Lin, C.-J. (2006b), "Generalized Bradley-Terry Models and Multi-Class Probability Estimates," *Journal of Machine Learning Research*, 7(Jan), 85–115.

Hunter, D. R. (2004), "MM algorithms for generalized Bradley-Terry models," *The Annals of Statistics*, 32, 386–408.

Jarret, A. (2003), "IGDA Online Games White Paper 2nd Edition – March 2003," .

Joshua E. Menke (2007), "etpub Global Stats Site," .

Menke, J. E. and Martinez, T. R. (2007), "A Bradley-Terry Artificial Neural Network Model for Individual Ratings in Group Competitions," *To appear in Neural Computing and Applications*.

Parks Associates (2007), "Gaming Remains the Most Popular Online Entertainment Activity," .

Raftery, A. and Lewis, S. (1996), *Implementing MCMC*, Chapman and Hall, pp. 115–130.

The Entertainment Software Association (2006), "Essential Facts about the Computer and Video Game Industry," .

Wikipedia (2006), "Elo rating system," .

Woodcock, B. S. (2006), "MMOG Active Subscriptions 20.0 120,000+," .