Predicting the Outcome of Chess Games based on Historical Data

Diogo R. Ferreira

November 25, 2010

Abstract

This report describes an approach used in the competition "Chess Ratings – Elo versus the Rest of the World" that took place between August and November 2010. Using the Bradley-Terry model as starting point, we postulate that the strength of each player can be approximated by the expected score against a common but unknown reference player, whose strength is never actually computed. This approach provides an iterative procedure for computing the strength of each player based on the strength of its opponents and on the average score against those opponents. To ensure convergence and to better fit the model to the training data, the approach can be tuned with several parameters such as a prior for the average score against opponents, a prior for the average strength of opponents, and a time factor to gradually reduce the weight of past games. In this competition, the approach was tuned in order to minimize a month-aggregated RMSE of game predictions.

1 Introduction

Although the Elo rating system (Elo, 1978) is well-established as the official method of ranking chess players, one may wonder whether it is possible to devise approaches that are better at predicting the outcome of games between ranked players. In recent years, Jeff Sonas[1] has been looking into this problem and in 2010 he set up an online and open competition entitled *Chess Ratings – Elo versus the Rest of the World*[2]. This document describes the approach that finished 4th place (out of 258 teams) in the competition.

The question of whether official rankings are the best predictors for game results has been raised in other sports as well, notably tennis (Scheibehenne and Bröder, 2007). On the other hand, there is a growing and resurging interest in methods of paired comparisons (David, 1969) and their application to sports, including tennis (McHale and Morton, 2010) and chess (Glickman, 1999). In this context, the well-known Bradley-Terry model (Bradley, 1984) is a relatively simple and effective tool to obtain a ranking of players based on paired comparisons, i.e. the result of previous games.

[1] http://www.chessbase.com/newsdetail.asp?newsid=562
[2] http://kaggle.com/chess

Let $P(X > Y)$ denote the probability of a player X winning against an opponent Y. The Bradley-Terry model postulates that this probability can be expressed as,

$$P(X > Y) = \frac{1}{1 + \frac{\gamma_Y}{\gamma_X}} \tag{1}$$

where γ_X is a measure for the strength of a player X. This expression has some desirable properties, namely that when both players have equal strength, the result is 0.5 for both $P(X > Y)$ and $P(Y > X)$; and if Y is much stronger than X, then $P(X > Y)$ tends to zero and $P(Y > X)$ tends to 1.

In chess, there are three possible outcomes for a game between two players X and Y, and they are scored as follows:

- if player X wins, then X is awarded 1 point and Y is awarded 0 points;
- if the game ends in a draw, then both X and Y are awarded 0.5 points;
- if player Y wins, then X is awarded 0 points and Y is awarded 1 point.

Given that, in any case, the score is a value between $[0, 1]$, and given that players of equal strength are more likely to draw than players of uneven strength, one can use Eq.(1) as an approximation for the *expected score* of a game between X and Y. Although the Bradley-Terry model has been extended by other authors to account for the possibility of draws, e.g. (Davidson, 1970), here we will interpret $P(X > Y) = 0.5$ as a prediction of draw.

2 Prediction Model

Estimating the strength γ_X is made difficult by the fact that one must resort to the results of player X against other players whose strength is also unknown. Typically, the parameters $\{\gamma_X, \gamma_Y, \ldots\}$ in the Bradley-Terry model are estimated by maximizing the likelihood of the observed results, with an additional constraint such as $\sum_i \gamma_i = 1$ (Bradley, 1984). However, here we will use a different approach, leading us to a slightly different model.

If X played with Y then from Eq.(1) the strength of X is can be written as,

$$\gamma_X = \gamma_Y \cdot \frac{1}{\frac{1}{P(X > Y)} - 1} \tag{2}$$

where $P(X > Y)$ is now interpreted as the actual (average) score that player X obtained in all games played against Y.

But if Y played with Z then,

$$\gamma_Y = \gamma_Z \cdot \frac{1}{\frac{1}{P(Y > Z)} - 1} \tag{3}$$

and therefore, by substituting (3) into (2),

$$\gamma_X = \gamma_Z \cdot \frac{1}{\frac{1}{P(Y > Z)} - 1} \cdot \frac{1}{\frac{1}{P(X > Y)} - 1} \tag{4}$$

Now, by (1) and (4) it becomes possible to compute $P(X > Z)$ as,

$$P(X > Z) = \frac{1}{1 + \left(\frac{1}{P(Y > Z)} - 1\right) \cdot \left(\frac{1}{P(X > Y)} - 1\right)} \tag{5}$$

Eq.(5) means that if we know the results of games between X and Y and also the results between Y and Z, we can compute the expected result of X against Z (transitivity rule).

We are now at a critical point in our approach. Instead of estimating $\{\gamma_X, \gamma_Y, \ldots\}$ by maximizing the likelihood function subject to a constraint, we postulate that the strength of each player can be approximated by taking the expected score against a common *reference player*. This reference player is an abstract entity, whose strength does not need to be computed, and whose single purpose is to bring the strength of all players to comparable terms.

Let Z be the abstract reference player, and let the strength γ_X of a player X be defined as,

$$\gamma_X \triangleq P(X > Z) \tag{6}$$

Then the expected score of a game between a player X and another player Y becomes,

$$P(X > Y) = \frac{1}{1 + \frac{P(Y > Z)}{P(X > Z)}} \tag{7}$$

which a function of the ratio of the expected scores of Y playing with Z and X playing with Z, where Z is the reference player. This is our prediction model.

3 Estimating the Strength of Players

With $P(X > Z)$ as a measure of the strength of player X, Eq.(5) tells us that the strength of X can be computed from the strength of Y (a previous opponent of X) and from the score that X obtained against Y. In practice, if X plays multiple opponents, we make a further simplification by taking Y as the *multiset of opponents* of X, and interpreting $P(X > Y)$ as the *average score* of X against all of its opponents, and $P(Y > Z)$ as the *average strength* of those opponents, i.e.,

$$P(X > Y) = \frac{1}{\mid Y \mid} \sum_{y \in Y} P(X > y) \tag{8}$$

$$P(Y > Z) = \frac{1}{\mid Y \mid} \sum_{y \in Y} P(y > Z) \tag{9}$$

Eq.(5) is then the basis for an iterative procedure to estimate the strength $P(X > Z)$ of every player X, based on the average strength $P(Y > Z)$ of its opponents, and on the average score $P(X > Y)$ obtained against those opponents. Starting from an initial assumption, such that all players have the same strength, at iteration n we can compute the strength of each player X based on the strength of players computed in iteration $n - 1$ as follows:

$$P(X > Z)^{(n)} = \frac{1}{1 + \left(\frac{1}{P(Y > Z)^{(n-1)}} - 1\right) \cdot \left(\frac{1}{P(X > Y)} - 1\right)} \tag{10}$$

where $P(Y > Z)^{(n-1)}$ is computed by using the values of $P(y > Z)$ from the previous iteration.

The convergence of similar iterative procedures for the estimation of the Bradley-Terry model has been established under the assumption that there are no disjoint subsets of players, i.e. that it is not possible to find two subsets of players where no one from the first subset plays anyone from the second subset (Hunter, 2004). In the following, we will add a set of parameters to the model that, as a side effect, will make this assumption hold. In practice, we have observed that the iterative approach of Eq.(10) converges, although convergence can be faster or slower depending on the value of the following parameters:

- The first parameter to be included is a prior in $P(X > Y)$. Imagine that X played a single opponent y, and a single game with this opponent, which X won. Then $P(X > Y) = P(X > y) = 1$ and therefore $P(X > Z) = 1$, which means that X is an unbeatable player, a conclusion that was drawn from a single game. To avoid this, we add a prior $0 < \beta_1 < 1$ with a weight $\beta_0 > 0$ to $P(X > Y)$,

$$P(X > Y) = \frac{\beta_0 \cdot \beta_1 + \sum_{y \in Y} P(X > y)}{\beta_0 + \mid Y \mid} \tag{11}$$

  Here, β_0 represents a number of artificial games, while β_1 represents the average score obtained in those artificial games. This prior may represent a number of draws[3] if $\beta_1 = 0.5$, or it may be a different value adjusted by training the model on a given dataset. With non-zero values for β_0 and β_1, this prior also prevents $P(X > Y)$ from ever reaching zero, which would create a problem in Eq.(10).

- We add another prior to $P(Y > Z)$ to represent an artificial opponent with whom X has obtained the artificial score above.

$$P(Y > Z) = \frac{\beta_2 \cdot \beta_3 + \sum_{y \in Y} P(y > Z)}{\beta_2 + \mid Y \mid} \tag{12}$$

  Here, β_3 represents the strength of the artificial opponent, and β_2 represents a number of artificial games, which may be different from β_0 (in general, $\beta_2 \leq \beta_0$). If $\beta_2 = \beta_0$, then all artificial games β_0 have been played against the artificial opponent with strength β_3. If $\beta_2 < \beta_0$, then β_2 artificial games have been played with the artificial opponent β_3, and $\beta_0 - \beta_2$ artificial games are assumed to have been played against the same opponents Y[4].

- One last parameter to be included is a time factor. In the chess ratings competition, the training data contained the results of 65053 games extending over a period of 100 months (8 years and 4 months identified as months 1-100), and the goal was to predict the results of 7809 games over months 101-105. For this purpose, one needs to calculate the strength

[3] This is similar to the concept of *fake draws* in the Chessmetrics approach (Sonas, 2010).

[4] A similar distinction between fake draws against the same opponents and fake draws against an artificial opponent is used in the Chessmetrics approach (Sonas, 2010).

of players at the end of the training period (month 100). Clearly, recent games will be more important than old games to estimate the strength of players at a given month M. Therefore, we apply a weighting scheme to reduce the weight of games that are further away into the past[5]. While several weighting functions can be considered (such as exponential, normal, linear, etc.) here we apply a simple function in the form,

$$w_g = \frac{1}{1 + \beta_4 \cdot (M - m_g)/100} \tag{13}$$

where w_g is the weight of game g, M is the latest month in the training data, and m_g is the month when the game took place. The parameter β_4 allows us to specify the time weighting as shown in Fig.1. With this weighting scheme, Eqs.(11) and (12) must be adapted to,

$$P(X > Y) = \frac{\beta_0 \cdot \beta_1 + \sum_{g \in G(X,Y)} w_g \cdot P(X > y_g)}{\beta_0 + \sum_{g \in G(X,Y)} w_g} \tag{14}$$

$$P(Y > Z) = \frac{\beta_2 \cdot \beta_3 + \sum_{g \in G(X,Y)} w_g \cdot P(y_g > Z)}{\beta_2 + \sum_{g \in G(X,Y)} w_g} \tag{15}$$

where the sum $\sum_{g \in G(X,Y)}$ is over all games that took place between X and its opponents Y, and y_g is the opponent of X in game g. The quantity w_g is the time weight assigned to game g according to Eq.(13).

The algorithm for estimating the strength of players can now be summarized as follows:

Algorithm 1 Estimates the strength of every player X

1. Set the parameters $\boldsymbol{\beta} = \{\beta_0, \beta_1, \beta_2, \beta_3, \beta_4\}$ or adjust them based on the training data (discussed in section 5).
2. Initialize the strength $P(X > Z)$ of every player X with some value, e.g. 0.5.
3. For every player X, compute the strength $P(X > Z)$ according to Eq.(10), where $P(X > Y)$ and $P(Y > Z)$ are given by Eqs.(14) and (15) respectively, and w_g is given by Eq.(13).
4. Repeat step 3 until the changes in all $P(X > Z)$ are smaller than a certain threshold, e.g. 0.0001%.

4 Predicting Future Games

After the strength of every player has been computed according to Algorithm 1, the expected score for a future game between any two players X and Y is given

[5] A similar weighting scheme is used in the Chessmetrics approach (Sonas, 2010).

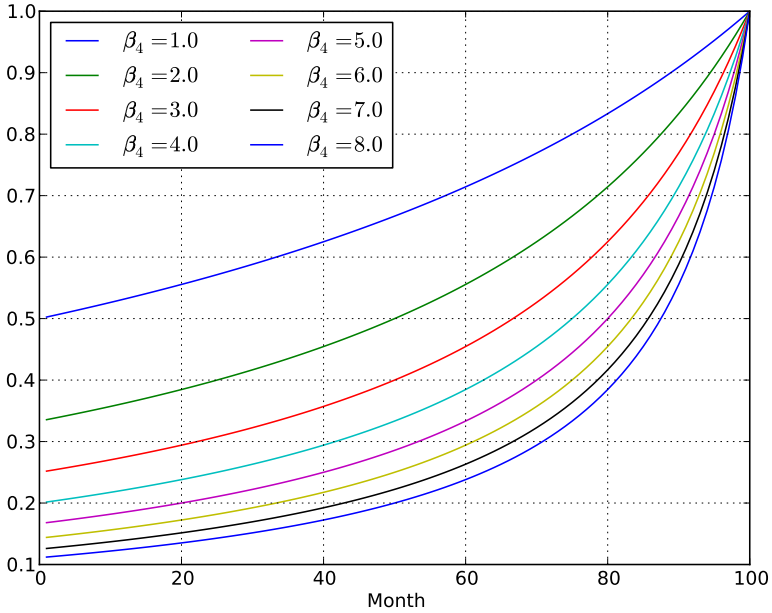

Fig. 1: Time weighting for several values of β_4 with $M = 100$

by Eq.(7). However, to account for the fact that, in chess, white pieces play first and therefore they have a slightly higher probability of winning the game, we apply a correction factor α_w to the result of Eq.(7). In a game between X and Y, if X is playing white then,

$$P^*(X > Y) = \frac{\alpha_w \cdot P(X > Y)}{\alpha_w \cdot P(X > Y) + (1 - \alpha_w) \cdot (1 - P(X > Y))} \tag{16}$$

where α_w is the average score of white in all chess games. When analyzing the training data for the competition, white is found to have an average score of 0.5456473952 over 65053 games, so the use of the above correction factor seems justified. Should α_w be 0.5, which would mean a perfect balance between white and black, Eq.(16) would give $P^*(X > Y) = P(X > Y)$. The advantage of playing white is comparable to the home-field advantage in other sports, a factor which can also be incorporated in Bradley-Terry models (Hunter, 2004).

5 Parameter Tuning

In the chess ratings competition, the error measure has been defined as the root mean square error per player and per month. On a given month, and based on the games that took place in that month, every player is expected to accumulate a certain score (i.e. the sum of all results, in number of points, that X obtained in that month). The error is defined as the difference between the expected score and the actual score accumulated by a player in a month. By averaging

the square of this error across all months and also across all players in each month, one obtains the month-aggregated RMSE used in the competition[6].

Other error measures could be considered, perhaps simpler ones such as the RMSE per game. However, the fact that the month-aggregated RMSE was chosen for the competition suggests that one should try to adjust the parameters $\boldsymbol{\beta} = \{\beta_0, \beta_1, \beta_2, \beta_3, \beta_4\}$ in order to minimize that error measure. Since the goal of the competition was to predict the result of games in months 101-105 based on data available for months 1-100, one idea is to tune $\boldsymbol{\beta}$ using months 1-95 for training and months 96-100 for cross-validation. However, this can lead to overfitting not only because the parameters become tuned for a single cross-validation set, but especially because the month-aggregated RMSE in one test set can be completely different from the month-aggregated RMSE in another test set, since month aggregation depends on the actual games that appear in each test set. Therefore, a better approach is to use several training/cross-validation sets and to minimize an average of the month-aggregated RMSE across those cross-validation sets.

In order to build training/cross-validation sets from months 1-100 that have similar characteristics to the test set of months 101-105 (whose games are to be predicted), one should have the following principles in mind:

- a cross-validation set should be 5 months long;
- a cross-validation set can only include those games for which both players appear in the training set;
- a cross-validation set should only include those games for which both players have at least 12 games in the final 48 months of the training set, since it is known that the test set of months 101-105 was built using players that had played at least 12 fully-rated games in months 53-100, where a fully-rated game is one in which both players have an official FIDE rating at the time of the game (except for newcomers, most players do).

With these principles in mind, on can easily build training/cross-validation sets such as 1-95/96-100, 1-94/95-99, 1-93/94-98, etc. Another strategy is to build non-overlapping datasets such as 1-95/96-100, 1-90/91-95, 1-85/86-90, etc. Yet another strategy is to consider only a limited number of training months. For example, by considering only 48 months for training, one has 48-95/96-100, 47-94/95-99, 46-93/94-98, etc. In any case, $\boldsymbol{\beta}$ should be tuned in order minimize the average of the chosen error measure across the available cross-validation sets. In the present approach, we have used all history back to month 1.

Algorithm 2 describes how the parameters $\boldsymbol{\beta}$ were tuned. Basically, at the beginning we try to change each parameter β_i by 50% of its value. When this does not produce an improvement in the average of the month-aggregated RMSE across the cross-validation sets, we decrease the amount of change to 25% of each parameter value, and then to 12.5%, and so on, until a desired precision is achieved in the values of $\boldsymbol{\beta}$. In general, using somewhere between 6 to 10 training/cross-validation sets, and requiring a precision of 0.0001% in $\boldsymbol{\beta}$, was enough to produce results within the top 10 places in the competition.

[6] For more information on the error measure, please refer to: http://kaggle.com/chess

Algorithm 2 Tuning of $\boldsymbol{\beta}$ using training/cross-validation sets built from months 1-100

1. Initialize the parameters $\boldsymbol{\beta} = \{\beta_0, \beta_1, \beta_2, \beta_3, \beta_4\}$ with some values, e.g. $\boldsymbol{\beta} = \{1.0, 0.5, 1.0, 0.5, 1.0\}$.
2. Initialize $step = 50\%$ and repeat the following sub-steps until $step$ is below a certain threshold, e.g. 0.0001%.
   (a) Run through the parameters $\boldsymbol{\beta}$ in a circular fashion, and for each parameter β_i do the following:
      i. If setting $\beta_i \leftarrow \beta_i + step \cdot \beta_i$ decreases the average of the month-aggregated RMSE across the cross-validation sets, then keep the new value of β_i; otherwise, revert to the previous value.
      ii. If setting $\beta_i \leftarrow \beta_i - step \cdot \beta_i$ decreases the average of the month-aggregated RMSE across the cross-validation sets, then keep the new value of β_i; otherwise, revert to the previous value.
   (b) When it is no longer possible to obtain a decrease in the average of the month-aggregated RMSE by changing any of the parameters $\boldsymbol{\beta}$, reduce the value of $step$ to $step \leftarrow step/2$ and go back to (a).

After experimenting with several training/cross-validation sets, we have chosen the following values for $\boldsymbol{\beta}$,

$$\begin{cases} \beta_0 = 8.389310 \\ \beta_1 = 0.455593 \\ \beta_2 = 1.681413 \\ \beta_3 = 0.012007 \\ \beta_4 = 7.444683 \end{cases} \tag{17}$$

achieving a month-aggregated RMSE of 0.695755, which yielded the fourth place in the final standings[7].

After the competition was over, and the true results for months 101-105 were released, we found that a different choice for these parameters, such as,

$$\begin{cases} \beta_0 = 12.73497 \\ \beta_1 = 0.455120 \\ \beta_2 = 0.656719 \\ \beta_3 = 0.000269 \\ \beta_4 = 8.211413 \end{cases} \tag{18}$$

would have achieved a month-aggregated RMSE of 0.694430, yielding a position above the current first place (0.694770). The question remains as to whether it would have been possible to obtain such parameter values by tuning over some selection of test/cross-validation sets built from months 1-100.

[7] The results of the competition are available at: http://kaggle.com/chess

References

Ralph A. Bradley. Paired comparisons: Some basic procedures and examples. In P.R. Krishnaiah and P.K. Sen, editors, *Nonparametric Methods*, volume 4 of *Handbook of Statistics*, pages 299–326. Elsevier, 1984.

Herbert A. David. *The method of paired comparisons*, volume 12 of *Griffin's Statistical Monographs & Courses*. Griffin & Co., 1969.

Roger R. Davidson. On extending the Bradley-Terry model to accommodate ties in paired comparison experiments. *Journal of the American Statistical Association*, 65(329):317–328, 1970.

Arpad E. Elo. *The Rating Of Chess Players, Past and Present*. Arco Pub., 1978.

Mark E. Glickman. Parameter estimation in large dynamic paired comparison experiments. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 48(3):377–394, 1999.

David R. Hunter. MM algorithms for generalized Bradley-Terry models. *The Annals of Statistics*, 32(1):384–406, 2004.

Ian McHale and Alex Morton. A Bradley-Terry type model for forecasting tennis match results. *International Journal of Forecasting*, In Press, Corrected Proof, 2010. doi: DOI: 10.1016/j.ijforecast.2010.04.004.

Benjamin Scheibehenne and Arndt Bröder. Predicting Wimbledon 2005 tennis results by mere player name recognition. *International Journal of Forecasting*, 23(3):415–426, 2007.

Jeff Sonas. Method used for the Chessmetrics benchmark. Posted on http://kaggle.com/chess in the forum topic "About the Chessmetrics Benchmark", August 2010.