Intrinsic Chess Ratings

Kenneth W. Regan[*]
University at Buffalo

Guy McC. Haworth[†]
University of Reading, UK

Abstract

This paper develops and tests formulas for representing playing strength at chess by the quality of moves played, rather than by the results of games. Intrinsic quality is estimated via evaluations given by computer chess programs run to high depth, ideally so that their playing strength is sufficiently far ahead of the best human players as to be a 'relatively omniscient' guide. Several formulas, each having intrinsic skill parameters s for *sensitivity* and c for *consistency*, are argued theoretically and tested by regression on large sets of tournament games played by humans of varying strength as measured by the internationally standard Elo rating system. This establishes a correspondence between Elo rating and the parameters. A smooth correspondence is shown between statistical results and the century points on the Elo scale, and ratings are shown to have stayed quite constant over time. That is, there has been little or no 'rating inflation'. The theory and empirical results are transferable to other rational-choice settings in which the alternatives have well-defined utilities, but in which complexity and bounded information constrain the perception of the utility values.

Keywords. Computer games, chess, sequential decision making, probabilistic inference, machine learning, data mining, statistics.

1 Introduction

Player strength ratings in chess, and other games including Go, Scrabble®, and Backgammon, are based on the results of games, and are subject to both luck when opponents blunder and to drift in the player pool. This paper aims to rate players intrinsically by the quality of their decisions, as benchmarked by computer programs run to sufficient depth. Our work brings new evidence on controversial questions of import to the chess community, with ramifications for skill assessment in other games:

1. Has there been 'inflation'—or deflation—in the chess Elo rating system over the past forty years?
2. Were the top players of earlier times as strong as the top players of today?
3. Does a faster time control markedly reduce the quality of play?
4. Can recorded games from tournaments where high results by a player are suspected as fraudulent reveal the extent to which luck or collusion played a role?

A previous study (Guid and Bratko 2006) evaluated only the chess world champions, used a relatively low depth (12 ply) of the now sub-elite program CRAFTY (Hyatt 2011), and most importantly for our purposes, evaluated only the played move and Crafty's preferred move, if different. The departure point for our work is that to assess chess skill more accurately, we use the more-informed context provided by the evaluations of all possible moves. At a game turn with L-many legal moves, we can list them $m_0, m_1, \ldots, m_{L-1}$ in order of their evaluations $e(m_0), e(m_1), \ldots, e(m_{L-1})$, and use these as a measure of the moves' utilities. Engines are often set to give the evaluations always from White's point of view, but we define $e(m_i)$ in terms of the player to move, in the standard units of a hundredth of a pawn. Thus the evaluations are listed in non-increasing order.

The *main statistical principle* is that when the move m_0 is given a clear standout evaluation, it is more likely to be selected by a human player than when it is given slight preference over several alternatives. For instance, if $e(m_0) \approx e(m_4) \gg e(m_5)$, then each of the five best moves should have somewhere near 20% probability.

Basically all chess programs proceed by *iterative deepening*, which means that move evaluations are computed for successive base-search depths $d = 1, 2, 3, \ldots$, and the values at depth $d - 1$ are used to structure the search at depth d. Our theory is formally indifferent about whether the evaluations come from a single program or are averaged over a jury of programs. The work reported here used the commercial chess program RYBKA 3 (Rajlich and Kaufman 2007), which was rated the strongest program on the CCRL rating list (Banks and others 2010) between its release in August 2007 and the release of RYBKA 4 in May 2010. RYBKA 3 was run in a mode that evaluates up to 50 moves fully, except for pruning the search for moves evaluated over 4 pawns worse than the optimal move, until it reported the end of the depth-13 iteration. In the vast majority of positions there

[*]Department of CSE, University at Buffalo, Amherst, NY 14260 USA; (716) 645-4738; regan@buffalo.edu

[†]School of Systems Engineering, University of Reading, UK; guy.haworth@bnc.oxon.org

were fewer than 50 un-pruned moves, so capping L at 50 did not affect the context information. A complication of using different chess programs is that the same search depth can represent widely different playing strengths between programs, since one may invest more time in advanced search heuristics and/or (pseudo-)random 'Monte Carlo' simulations of play starting with a given move, and hence take longer to complete the iteration at each depth. There have also been hearsay allegations that RYBKA versions understate the actual depth.

The only game-specific information used by our methods is the sequence of values $e(m_i)$ and the identity of the move m_* that was actually played. This minimal information is common to other games, and can come from any decision-making scenario in which authoritative evaluations of options are available, beyond what is comprehended by agents who are subject to bounded rationality. The general problem is to what extent the probabilities of selection of the various options are correlated with these actual (hindsight) utility values.

2 Background and Previous Literature

The Elo rating system (Elo 1978) computes an expected score E based on the differences $r_P - r_O$ between the rating of a player P and the ratings of various opponents O, and updates r_P according to the difference between the actual score and E. Since only rating differences matter there is no absolute meaning to the numbers produced, but considerable effort has been expended by the World Chess Federation (FIDE) and national federations to keep the ratings stable over time. The term 'master' generally designates a player rated 2200 or above, while grandmasters typically have ratings above 2500, and world champions since Bobby Fischer in 1972 have held ratings around 2800. This paper samples games between players rated within 10 points of a given Elo century mark.

By fitting to itemized skill levels, our paper continues work on *reference fallible agents* in the game context (Reibman and Ballard 1983; Haworth 2003; Haworth and Andrist 2004; Andrist and Haworth 2005). The aim is create stochastic agents whose overall behavior imitates that of actual players. The present work establishes that such a reference-fallible model is well supported by data taken on a far larger scale than previous studies. This paper uses a richer model than (DiFatta, Haworth, and Regan 2009; Haworth, Regan, and DiFatta 2010) which are based on Bayesian inference.

We have already mentioned the Guid-Bratko study. Their 'Average Difference' (AD) statistic is the average of $e(m_0) - e(m_*)$, subject to some adjustments for 'complexity' that we did not employ. We ran not only their selection of world-championship games but also more than 150,000 high-level games representing most of the history of chess, using RYBKA 3 to depth 13 in Single-PV mode, which is the program's normal playing mode. Owing to search-pruning and the program's manner of reporting results, this mode guarantees full evaluation only for the preferred move m_0. Thus when the played move m_* was different, $e(m_*)$ was estimated by using the evaluation $e(m_0')$ of the position that resulted from m_* being played. These runs were used in the work reported here only to determine a scaling adjustment described in Section 3.1 below.

3 Basic Model

Our key abstract concept is a model fallible agent A, which is intended to represent a player P of a given skill level. The agents are stochastic, so the main question is, what is the probability p_i of A choosing move m_i at a given game turn t? We wish to generate values of p_i, based on the values of fitted parameters that define A, that are realistic for the players P that A is modeling. Once the parameters are given, p_i should be a function of the evaluations $e(m_j)$ of the available moves at turn t. Before introducing our parameters, we note three principles about the definition of the agents A.

(a) The choices made by A at different game turns are independent, even for successive game turns.

(b) The context of alternative move choices is used only to define *probability proxies* y_i for the moves m_i. The proxies depend on any agent parameters, here called s and c and defined below, but the manner of converting them into the projected probabilities p_i does not. The latter conversion is the same for all agents, modeling players of all skill levels.

(c) For all agents A, the proxies are given by selecting a curve g_A from a family of like curves according to the parameters for that agent, and computing $y_i = g_A(\delta_i)$. Here δ_i is computed from the difference $e(m_0) - e(m_i)$ but weighted as described below, while g_A is a *continuous, decreasing* function of δ_i, with $g_A(0) = 1$.

Principle (a) is needed in order to regard p_i as a function of information for the given turn t alone. It enables us to multiply probabilities from different turns, and add probabilities to project (for instance) the total number of matches to a computer's first choices over a sequence of moves. It stands in contradistinction to the reality that an actual player's consecutive moves are often part of an overall plan.

Principle (b) allows the presence and values of alternative moves to affect the probability p_i of a given move m_i. The difference from saying that p_i itself depends on $e(m_i)$ and $e(m_0)$ alone, which would ignore the context information of alternative move choices, is that the proxies undergo a separate conversion. This need not be the simple normalization $p_i = y_i/Y$ where $Y = \sum_j y_j$.

In general we consider functions $r(\cdot)$ that determine the following kind of relation between the proxies and the probabilities:

$$y_i = \frac{r(p_i)}{r(p_0)}.$$

Choosing r to be the identity function leads to the simple normalization above. We found, however, that better results are obtained by using $r(p) = 1/\ln(1/p)$ instead. Then the conversion from the proxies into the projected probabilities is obtained by solving the equations

$$p_i = p_0^{1/y_i}; \qquad \sum_i p_i = 1.$$

Principle (c) mandates that inferior moves have smaller proxies, for all agents. It also implies that a revision downward in the authoritative value of a move causes the model to generate a smaller proxy for it, which yields a smaller probability for it, for all agents and regardless of the values of other moves.

3.1 Evaluation Scaling

When one side is far ahead, one might expect that player to care less about small differences in value, and the player who is behind to take risks that a machine sees how to answer at high depth but a human may not. We thought this effect might materialize when the advantage was more than a pawn, but in fact we observed a uniform scaling effect all the way down to zero advantage. From the supplementary Single-PV data on over 150,000 chess games mentioned above, we compiled histograms by plotting the recorded 'raw' AD statistic against $e(m_0)$ in intervals of 0.05 or 0.10, excluding the origin. It was also found that the raw AD for moves where the computer judges the player to be at a disadvantage $-e$ is significantly higher than for an advantage of $+e$.

These observations suggested that to model this large-scale human behavior, the agents' δ_i should not always be $e(m_0) - e(m_i)$, but rather the integral from $e(m_0)$ to $e(m_i)$ of some function defining a non-unit differential. The simple differential $d\mu = \frac{1}{1+x}dx$, with integral $\ln(1+x)$, was found to level the AD histogram very well, in aggregations of all kinds of chess events: round-robin tournaments of every level and historical era, championship matches, open Swiss events, rapid, and blitz. Note that in case of a move judged inferior by b pawns, the integral for the player at a disadvantage goes from $-e$ to $-e-b$, away from the origin, while for the player starting ahead $+e$ the integral goes from e to $e-b$, toward the origin. The former will be scaled down more, thus tending to equalize the resulting *scaled AD* (*ad*) for the $-e$ and $+e$ points of the modified histogram.

The steepness of this scaling also reduced the difference between, say, a 2-pawn blunder and a 5-pawn blunder. This is desirable because both blunders may be equally fatal, hence nearly equally unlikely for a strong player. Moreover, RYBKA 3 was the first program to offer a 'Multi-PV cap' feature whereby it prunes the search when a move is found to be more-than-the-cap worse than the top move. Setting this at 4.00 pawns saved huge amounts of time computing bad moves, and also skirted a known issue with all RYBKA versions when the evaluation exceeds ± 5.09 pawns.

Finally, it was observed that in cases of equal-top moves, the first-listed move was played significantly more often, to similar degree across all training sets. To compensate for this phenomenon (see Section 8 for speculation on cause), a correction of -0.03 on later tied moves was determined and applied in-tandem with the scaling.

We did not try to optimize the scaling, for instance by finding c such that $d\mu' = \frac{1}{c+x}dx$ achieved a better measure of 'level' across some suite of data sets. Optically all c outside $[0.8, 1.2]$ produced inferior results. Rather than compose the integral $\ln(1+x)$ into the curves g_A defining the agents, we regarded the scale correction as a natural feature of the data and the model, common to all agents modeling human players. We speculate that other games will be found to obey a similar scaling law, perhaps implying that human perception of value differences is figuratively judged on log-log paper in proportion to the overall advantage or disadvantage.

4 Parameterizing the Fallible Agents

We regard the following two parameters s, c as core features of agent behavior, with common meanings across the agent space, and across alternate curve choices described below. We introduce them via the inverse-exponential curve used for the results in this paper:

$$y_i = e^{-(\delta_i/s)^c}. \tag{1}$$

The s parameter represents a conversion from the hundredths-of-a-pawn units of δ_i into the dimensionless quantity δ_i/s used in all of our curves. The smaller s, the greater the ratio δ_i/s, thus lowering the proxy and hence the projected probability of the i-th move. Thus s governs an agent's ability to discriminate among moderately inferior move choices, so we call it the *sensitivity*.

The parameter c appears as an exponent of δ/s, directly or with some intervening terms. Intuitively it governs how often a player avoids moves in the range the player discriminates as inferior, and avoids poor moves overall. Hence we regard it as a notion of *consistency*.

Although it is a digression from the results to mention other curve families, the following alternatives, all normalized to make $g(0) = 1$, illustrate the parameters in similar roles:

- Inverse-polynomial curves:

$$\begin{aligned} ipa(\delta) &= \frac{1}{1+(\delta/s)^c}, \qquad \text{or} \\ ipb(\delta) &= \frac{1}{(1+\delta/s)^c}. \end{aligned}$$

- Logistic function-related curves:

$$\begin{aligned} secha(\delta) &= \frac{2}{(e^{(\delta/s)^c} + e^{-(\delta/s)^c})} \qquad \text{or} \\ sechb(\delta) &= \frac{4}{(e^{(\delta/s)^c} + 2 + e^{-(\delta/s)^c})}. \end{aligned}$$

All of these curves were found to give similar results, largely owing to the way they approximate each other near the origin. As stated above, we standardized our present work on the inverse-exponential family.

5 Data Methodology and Experiments

The main data set comprised games in which both players were within 10 Elo rating points of one of the 'milepost' values: 2700, 2600, 2500, ..., run under standard time controls in individual-player round-robin or small-Swiss tournaments. Team events and large-Swiss (meaning more than six times as many players as rounds) tournaments were excluded. Games were taken from three time periods: 2006–2009, 1991–1994, and 1976–1979. These games were evaluated to depth 13 ply in 50-PV mode, using single core threads for each program instance on a 64-bit quad-core PC. Turns 1–8 of every game were skipped, as were turns that were identified as part of a repetition, and positions with greater than three pawns' advantage.

Each such 'milepost data set' had at least 5,000 turns, while the largest set had just over 25,000 turns, so the relative sizes were reasonably consistent. Each set comprised *all* available games meeting the description, from two major commercial game collections (ChessBase 2009; Horvath and others 2010), so as to avoid any bias in game selection.

For 1976–1979 it was possible to find reliable and large-enough game sets only for the 2300 through 2600 mileposts, while 1991–1994 made it possible to add 2700 and 2200. Since FIDE has expanded its rating system to players below 2200 in recent years, for 2006–2009 it was possible to find enough games down to 1600. The ranges around mileposts were expanded from ±10 to ±15 or ±20 for some of the lower sets.

Co-designer GM Larry Kaufman opined that RYBKA 3 at reported depth $d = 14$ would play at 2750 level, stronger in the early phases but weaker in the endgame (Kaufman 2008). By general considerations, and consistent with his estimates for low depth in the same forum thread, this would place depth 13 in the 2650-to-2700 range. This choice enabled the over 6,000 games in the data sets to be analyzed on available hardware over a reasonable timeframe, at 6-8 hours per processor core per game on average.

6 Fitting Methodology

This section describes how we determined the best agent $A = A_{s,c}$ to model the players in each 'milepost' data set. Recall that the analysis of the position at a given game turn t furnishes values δ_i^t for each move m_i (after scaling, and always with $\delta_0^t = 0$). For any fixed values s, c defining the agent, we obtain the proxy values $y_i^t = g_{s,c}(\delta_i^t)$. The proxies are then converted into the probabilities p_i that agent $A_{s,c}$ chooses move m_i. It remains to see how well these probabilities correspond to frequencies of move choice in the data set.

Given the probabilities p_i^t for each game turn t, we can also generate aggregate projections for the agreement mm_0 between the first-listed moves m_0^t and the moves m_*^t that were actually played, and for the average error, by:

$$\begin{aligned} mm_p &= \frac{1}{T}\sum_t p_0^t \\ ad_p &= \frac{1}{T}\sum_t(\sum_i \delta_i^t p_i^t). \end{aligned}$$

We can similarly project the frequency $M_i = \frac{1}{T}\sum_t p_i^t$ of playing the i-th best move, for all i (so $mm_p = M_0$). These projected quantities depend (only) on the machine evaluations of the moves and the parameters s, c which yield the p_i^t. It remains to find the s, c for which these quantities are in best accord with the frequencies $mm_a(i)$ over t such that $m_*^t = m_i^t$, for each $i = 0, 1, 2, \ldots$, and for which ad_p agrees with the actual (scaled) average difference, calculated as $ad_a = \frac{1}{T}\sum_t(e(m_0^t) - e(m_*^t))$. This suggests several fitting ideas:

- Calculate s, c to make $mm_p = mm_a(0)$ and $ad_p = ad_a$, or if that is not possible, to minimize the (squared) distance between the respective sides of each equation.
- Calculate s, c to maximize the projected likelihood $\prod_t p_*^t$, which the same as minimizing $\sum_t \ln(1/p_*^t)$.
- The 'percentiling' method described next, which we actually used.

The trouble observed with the first is that the projections generated for M_i, $i > 0$, were wide of the mark. The second failed to yield even remotely accurate projections for mm_p and ad_p. The third attempts to make the M_i projections for $i \geq 1$ more accurate. First we motivate it.

If all *spread tuples* were the same $\Delta = (0, \delta_1, \ldots, \delta_{N-1})$, or if we had a large-enough set of nearly-equal tuples to form a histogram, fitting the results to a curve would be relatively simple. Let $f_1, f_2, \ldots, f_N, f_{N+1}$ be the observed frequencies of which indexed move in the spread was played, with f_{N+1} standing for "move played not in the top N" and hopefully negligibly small. Then given a curve $g_{s,c}(\delta)$ and (power-of-)metric μ, such as $\mu(x,y) = |x-y|^2$ for least-squares, we could compute the fit score $S_{s,c} =$

$$\mu(f_1, \frac{1}{S}) + \mu(f_2, \frac{g_{s,c}(\delta_2)}{S}) + \cdots + \mu(f_N, \frac{g_{s,c}(\delta_N)}{S}),$$

where $S = 1 + g_{s,c}(\delta_2) + \cdots + g_{s,c}(\delta_N)$. In the case of equal spreads this yields the same best-fit s, c as maximum-likelihood estimation.

When the tuples are different, however, we cannot do this. The idea is to use a fixed grid of percentile points rather than allow the δ_i to set the grid. Given a curve $g_{s,c}(\delta)$, let q additionally stand for a percentile. For each point (q, s, c) in a fine-enough grid, say stepping q by 0.05 from 0 to 1, s by 0.02 from 0 to 0.70, and c by 0.20 from 1 to 5, we iterate through each spread tuple $\Delta_t = (0, \delta_2, \ldots, \delta_N)$. For each i, $1 \leq i \leq N$, compute the probabilities $p_i = g_{s,c}(\delta_i)/S_t$, where $S_t = \sum_i g_{s,c}(\delta_i)$. Let i_t be the index of the played move. Define $p^- = \sum_{j=1}^{i_t - 1} p_j$ and $p^+ = p^- + p_{i_t}$, giving the endpoints of the predicted probability interval of the played move. Then:

- If $p^+ \leq q$, call the tuple 'up'.
- If $p^- \geq q$, call the tuple 'down'.
- If $p^- < q < p^+$, so that the prediction for the played move straddles the q-th percentile, count the tuple as being $|q - p^-|/p_{i_t}$ up, and $|q - p^+|/p_{i_t}$ down.

Finally define $R_{s,c}^q$ to be the percentage of 'up' tuples. Given a μ as above, the score now becomes

$$S_{s,c} = \sum_q \mu(R_{s,c}^q, q).$$

A low score indicates a good fit across a range of percentiles for the curve $g_{s,c}(\delta)$.

To interpret this, consider a case with one clearly-best move given probability $p_0 = 0.9$ that is played. For $q = 0.30$ the tuple will count as one-third up, two-thirds down. It may seem counter-intuitive for a result that confirms a prediction to give an overall 'down' score, but the prediction that is actually tested by our method is not the individual move but the overall frequency of hits above/below a given percentile. Nor is it necessary to estimate the proportion of the cumulative distribution of $g_{s,c}(\delta)$ to the left and right of 0.30 in the spanned range—the straight one-third/two-thirds division is correct. In effect we have converted from the 'δ scale' to the percentile scale, with the effect that instead of plotting data points for a horizontal δ-axis and fitting $g_{s,c}(\delta)$, we fit the derived percentile curve(s) instead.

The quality of fit can be judged in several ways. One is the $S_{s,c}$ score itself, divided by the number of percentile points. Another is the average value of $\mu(M_i, mm_a(i))$ over all move-indices i. The latter speaks the purpose of the percentiling method for fitting the frequencies of playing the ith-best move for all i, in a way that involves no explicit correction for skedasticity. Since the μ used here is squared-distance and we compare percentages, we multiplied by $10,000$ to create the 'Q_{fit}' entries in the tables below.

7 Results

A statistical analyzing program written in C++ carried out the two-dimensional minimization needed to implement the above fitting method. It was found that while s varied from 0.07 to 0.16 and beyond, the c value stayed between 0.430 and 0.545. Accordingly we computed a linear interpolation of the c values for 2006–2009, getting intervals coincidentally highly close to 0.007, yielding the c_{fit} column. With these fixed, one-dimensional fitting of the s-values then gave the final s_{fit} values. The s_{fit} and c_{fit} values defined the agents, for which projected move-match and average-difference statistics (mm_p and ad_p) are shown compared with the actuals (mm_a and ad_a), which together with the above Q_{fit} measure are shown in the following tables.

2006–2009

Elo	s	c	c_{fit}	s_{fit}	mm_p/mm_a	ad_p/ad_a	Q_{fit}
2700	.078	.503	.513	.080	56.2/56.3	.056/.056	.009
2600	.092	.523	.506	.089	55.0/54.2	.063/.064	.041
2500	.092	.491	.499	.093	53.7/53.1	.067/.071	.028
2400	.098	.483	.492	.100	52.3/51.8	.072/.074	.016
2300	.108	.475	.485	.111	51.1/50.3	.084/.088	.044
2200	.123	.490	.478	.120	49.4/48.3	.089/.092	.084
2100	.134	.486	.471	.130	48.2/47.7	.099/.102	.034
2000	.139	.454	.464	.143	46.9/46.1	.110/.115	.065
1900	.159	.474	.457	.153	46.5/45.0	.119/.125	.166
1800	.146	.442	.450	.149	46.4/45.4	.117/.122	.084
1700	.153	.439	.443	.155	45.5/44.5	.123/.131	.065
1600	.165	.431	.436	.168	44.0/42.9	.133/.137	.129

1991–1994

Elo	s	c	c_{fit}	s_{fit}	mm_p/mm_a	ad_p/ad_a	Q_{fit}
2700	.079	.487	.513	.084	55.2/54.9	.058/.060	.043
2600	.092	.533	.506	.087	55.3/54.6	.064/.063	.042
2500	.098	.500	.499	.092	54.3/53.8	.068/.069	.013
2400	.101	.484	.492	.103	52.3/51.9	.077/.079	.016
2300	.116	.480	.485	.117	51.0/50.3	.088/.091	.031
2200	.122	.477	.478	.122	49.7/48.7	.092/.098	.058

1976–1979

2600	.094	.543	.506	.087	53.8/53.0	.062/.061	.038
2500	.094	.512	.499	.091	53.2/52.5	.067/.068	.032
2400	.099	.479	.492	.103	52.3/51.7	.076/.079	.020
2300	.121	.502	.485	.116	50.9/50.0	.088/.090	.070

We conclude that there is a smooth relationship between the actual players' Elo ratings and the intrinsic quality of the move choices as measured by the chess program and the agent fitting. Moreover, the final s_{fit} values obtained are nearly the same for the corresponding entries of all three time periods. Since a lower s indicates higher skill, we conclude that there has been little or no 'inflation' in ratings over time—if anything there has been deflation. This runs counter to conventional wisdom, but is predicted by population models on which rating systems have been based (Glickman 1999).

The results also support a no answer to question 2. In the 1970's there were only two players with ratings over 2700, namely Bobby Fischer and Anatoly Karpov, and there were years as late as 1981 when no one had a rating over 2700 (see (Weeks 2000)). In the past decade there have usually been thirty or more players with such ratings. Thus lack of inflation implies that those players are better than all but Fischer and Karpov were. Extrapolated backwards, this would be consistent with the findings of (Dangauthier et al. 2007), which however (like some recent competitions to improve on the Elo system) are based only on the results of games, not on intrinsic decision-making.

8 Summary and Future Agenda

We have demonstrated that the intrinsic quality of move choice can be ascertained based on benchmarking measures rather than the results of games. We have modeled actual players by a class of stochastic agents employing full information about alternative move choices, and have demonstrated a smooth correlation between their parameters obtained by standard regression techniques and the players' Elo ratings. We have shown that the general belief that the Elo ratings of strong players suffer from inflation is ill-founded.

We have essentially fitted only the means of the respective skill levels. The next task is to obtain projected variances and confidence intervals, and test their correspondence to the training data. This will yield a model of move choice that is capable of testing allegations of collusion during games, thus answering question 4. Testing the curve of skill as a function of time alotted per game, for question 3, will require larger amounts of data.

Another use for our model will be a better simulation of human players of specified skill levels, especially being faithful to their observed tendency to make occasional poor moves. Among other things, this may yield more-realistic settings for human-computer play and practical tournament training. Insofar as our methods involve almost no information specific to chess, they should be transferable to other domains.

To add to the above about the preliminary nature of this work, the first author believes that an important improvement will be to involve the evaluations $e_d(m_i)$ of moves at depths d lower than the reference depth. The extended model computes probability proxies as linear combinations with non-negative weights w_d over the depths, namely:

$$y_i = \sum_d w_d \cdot g_{s,c}(\delta_{i,d}).$$

All other formalism is the same—hence this paper can be viewed as the special case $w_{13} = 1$, all other weights zero. The weights w_d are also agent properties; we would expect stronger players to have higher weights for higher d. Of course the weights would proliferate the number of parameters needing to be fitted.

One effect of using this extended model would be that moves m_i whose strength becomes apparent only at higher depths would have lower y_i values than the present paper computes, especially for agents modeling weaker players. Moves whose evaluations fall would have higher proxies, and avoiding them would be another measure of skill. These effects might cancel in large enough test sets. We also speculate that the equal-top move effect in Section 3.1 may owe to the first move having had higher value at the lowest depths d, which the w_d might correct for naturally, but we have been hindered in investigating this by issues with the reporting of the bottom depths in RYBKA 3. Only recently have open-source engines of comparable strength emerged. Using other programs, going to higher d, and improving the mining of our many gigabytes of data are slated for future work.

Further methodological details, including the incidence of gamescore errors in the test-data sets and the policy on fixing them, and difficulties in the identification of repetitions (amending '0.00' values given by chess programs that are premature according to the rules of chess), will be found in links from the first-author's website, `www.cse.buffalo.edu/~regan/chess/`, along with the data and C++ and Perl code used here.

Acknowledgments We thank the referees for many helpful comments, and we upgraded the description of scaling. Perl scripts for processing the data were partly prepared by University at Buffalo students Tamal Biswas and Daniel Ruszaj. Biswas also helped in identifying gamescore errors and charting tournament data.

References

Andrist, R., and Haworth, G. 2005. Deeper model endgame analysis. *Theoretical Computer Science* 349:158–167.

Banks, G., et al. 2010. CCRL rating lists. http://www.computerchess.org.uk/ccrl/.

ChessBase. 2009. Big2010 Chess Database.

Dangauthier, P.; Herbrich, R.; Minka, T.; and Graepel, T. 2007. TrueSkill through time: Revisiting the history of chess. Microsoft Report 74417, research.microsoft.com/pubs/74417/NIPS2007_0931.pdf. Poster, 2007 Neural Information Processing (NIPS) workshop.

DiFatta, G.; Haworth, G.; and Regan, K. 2009. Skill rating by Bayesian inference. In *Proceedings, 2009 IEEE Symposium on Computational Intelligence and Data Mining (CIDM'09), Nashville, TN, March 30–April 2, 2009*, 89–94.

Elo, A. 1978. *The Rating of Chessplayers, Past and Present*. New York: Arco Pub.

Glickman, M. E. 1999. Parameter estimation in large dynamic paired comparison experiments. *Applied Statistics* 48:377–394.

Guid, M., and Bratko, I. 2006. Computer analysis of world chess champions. *ICGA Journal* 29(2):65–73.

Haworth, G., and Andrist, R. 2004. Model endgame analysis. In *Advances in Computer Games*, volume 135, 65–79. Kluwer Academic Publishers, Norwell MA.

Haworth, G.; Regan, K.; and DiFatta, G. 2010. Performance and prediction: Bayesian modelling of fallible choice in chess. In *Proceedings, 12th ICGA Conference on Advances in Computer Games, Pamplona, Spain, May 11–13, 2009*, volume 6048 of *Lecture Notes in Computer Science*, 99–110. Springer-Verlag.

Haworth, G. 2003. Reference fallible endgame play. *ICGA Journal* 26:81–91.

Horvath, A., et al. 2010. Opening Master 2.0x Full Database. http://www.openingmaster.com/.

Hyatt, R. 2011. Crafty chess engine. http://www.craftychess.com/.

Kaufman, L. 2008. http://rybkaforum.net/cgi-bin/rybkaforum/topic_show.pl?pid=83148#pid83148.

Rajlich, V., and Kaufman, L. 2007. Rybka 3 chess engine. http://www.rybkachess.com.

Reibman, A., and Ballard, B. 1983. Non-minimax strategies for use against fallible opponents. In *proceedings, Third National Conference on Artificial Intelligence (AAAI-83)*.

Weeks, M. 2000. FIDE historical ratings. http://www.markweeks.com/chess/ratings/.