Statistical Pitfalls and Lessons from a Model of Human Decision-Making at Chess

Kenneth W. Regan[1]
University at Buffalo (SUNY)

Indian Statistical Institute, 2 Aug. 2016
Updated for UB Computational Science Club, 2 Apr. 2018

[1] Joint work with Tamal Tanu Biswas and with grateful acknowledgment to UB's Center for Computational Research (CCR)

Chess History, Ancient and Modern

- Chess, either in Four Army form (Chatur-Angha) or today's White & Black, was known 2,500 years ago on the Subcontinent.

Chess History, Ancient and Modern

- Chess, either in Four Army form (Chatur-Angha) or today's White & Black, was known 2,500 years ago on the Subcontinent.
- Required knowledge for military commanders. Many conquests.

Chess History, Ancient and Modern

- Chess, either in Four Army form (Chatur-Angha) or today's White & Black, was known 2,500 years ago on the Subcontinent.
- Required knowledge for military commanders. Many conquests.
- Final conquest in 1997 by army of...

Chess History, Ancient and Modern

- Chess, either in Four Army form (Chatur-Angha) or today's White & Black, was known 2,500 years ago on the Subcontinent.
- Required knowledge for military commanders. Many conquests.
- Final conquest in 1997 by army of... processors. **Deep Blue**.

Chess History, Ancient and Modern

- Chess, either in Four Army form (Chatur-Angha) or today's White & Black, was known 2,500 years ago on the Subcontinent.
- Required knowledge for military commanders. Many conquests.
- Final conquest in 1997 by army of... processors. **Deep Blue**.
- Later conquered in 2017 by army of...

Chess History, Ancient and Modern

- Chess, either in Four Army form (Chatur-Angha) or today's White & Black, was known 2,500 years ago on the Subcontinent.
- Required knowledge for military commanders. Many conquests.
- Final conquest in 1997 by army of... processors. **Deep Blue**.
- Later conquered in 2017 by army of... nothing: **AlphaZero**.
- Now the army of handheld devices running chess programs (called **engines**) can defeat Carlsen, Anand, Kramnik, Kasparov, anyone.

Chess History, Ancient and Modern

- Chess, either in Four Army form (Chatur-Angha) or today's White & Black, was known 2,500 years ago on the Subcontinent.
- Required knowledge for military commanders. Many conquests.
- Final conquest in 1997 by army of... processors. **Deep Blue**.
- Later conquered in 2017 by army of... nothing: **AlphaZero**.
- Now the army of handheld devices running chess programs (called **engines**) can defeat Carlsen, Anand, Kramnik, Kasparov, anyone.
- Since 2006, real and alleged chess cheating has been a major problem.

Chess History, Ancient and Modern

- Chess, either in Four Army form (Chatur-Angha) or today's White & Black, was known 2,500 years ago on the Subcontinent.
- Required knowledge for military commanders. Many conquests.
- Final conquest in 1997 by army of... processors. **Deep Blue**.
- Later conquered in 2017 by army of... nothing: **AlphaZero**.
- Now the army of handheld devices running chess programs (called **engines**) can defeat Carlsen, Anand, Kramnik, Kasparov, anyone.
- Since 2006, real and alleged chess cheating has been a major problem.
- First person caught and banned: Umakant Sharma, banned 12/2006 for 10 years by the AICF. Has a Wikipedia page,

Chess History, Ancient and Modern

- Chess, either in Four Army form (Chatur-Angha) or today's White & Black, was known 2,500 years ago on the Subcontinent.
- Required knowledge for military commanders. Many conquests.
- Final conquest in 1997 by army of... processors. **Deep Blue**.
- Later conquered in 2017 by army of... nothing: **AlphaZero**.
- Now the army of handheld devices running chess programs (called **engines**) can defeat Carlsen, Anand, Kramnik, Kasparov, anyone.
- Since 2006, real and alleged chess cheating has been a major problem.
- First person caught and banned: Umakant Sharma, banned 12/2006 for 10 years by the AICF. Has a Wikipedia page,
- I advise the World Chess Federation (FIDE) on cases, "too many..."

Chess History, Ancient and Modern

- Chess, either in Four Army form (Chatur-Angha) or today's White & Black, was known 2,500 years ago on the Subcontinent.
- Required knowledge for military commanders. Many conquests.
- Final conquest in 1997 by army of. . . processors. **Deep Blue**.
- Later conquered in 2017 by army of. . . nothing: **AlphaZero**.
- Now the army of handheld devices running chess programs (called **engines**) can defeat Carlsen, Anand, Kramnik, Kasparov, anyone.
- Since 2006, real and alleged chess cheating has been a major problem.
- First person caught and banned: Umakant Sharma, banned 12/2006 for 10 years by the AICF. Has a Wikipedia page,
- I advise the World Chess Federation (FIDE) on cases, "too many..."
- My statistical model has many other uses. My current CSE712 seminar may help to sharpen it.

Elo Rating System

- Named for the Hungarian-American statistician **Arpad Elo**, the system gives every player P a number R_P representing skill.

Elo Rating System

- Named for the Hungarian-American statistician **Arpad Elo**, the system gives every player P a number R_P representing skill.
- Defined by Logistic Curve: expected win % p given by

$$p = \frac{1}{1 + \exp(c\Delta)}$$

where $\Delta = R_P - R_O$ is the difference to your opponent's rating and c is a conversion constant.

Elo Rating System

- Named for the Hungarian-American statistician **Arpad Elo**, the system gives every player P a number R_P representing skill.
- Defined by Logistic Curve: expected win % p given by

$$p = \frac{1}{1 + \exp(c\Delta)}$$

where $\Delta = R_P - R_O$ is the difference to your opponent's rating and c is a conversion constant.
- USCF takes $c = (\ln 10)/400$, so 200-pointse $\approx 75\%$ expectation.

Elo Rating System

- Named for the Hungarian-American statistician **Arpad Elo**, the system gives every player P a number R_P representing skill.
- Defined by Logistic Curve: expected win % p given by

$$p = \frac{1}{1 + \exp(c\Delta)}$$

  where $\Delta = R_P - R_O$ is the difference to your opponent's rating and c is a conversion constant.
- USCF takes $c = (\ln 10)/400$, so 200-pointse $\approx$ 75% expectation.
- **Class Units**: 2000–2200 = Expert, 2200–2400 = Master, 2400–2600 is typical of International/Senior Master and Grandmaster ranks, 2600–2800 = "Super GM,"; Carlsen 2857, 3 others over 2800, Anand 2770.

Elo Rating System

- Named for the Hungarian-American statistician **Arpad Elo**, the system gives every player P a number R_P representing skill.
- Defined by Logistic Curve: expected win % p given by

$$p = \frac{1}{1 + \exp(c\Delta)}$$

  where $\Delta = R_P - R_O$ is the difference to your opponent's rating and c is a conversion constant.
- USCF takes $c = (\ln 10)/400$, so 200-pointse $\approx$ 75% expectation.
- **Class Units**: 2000–2200 = Expert, 2200–2400 = Master, 2400–2600 is typical of International/Senior Master and Grandmaster ranks, 2600–2800 = "Super GM,"; Carlsen 2857, 3 others over 2800, Anand 2770. Adult beginner $\approx$ 600, kids $\to$ 100.

Elo Rating System

- Named for the Hungarian-American statistician **Arpad Elo**, the system gives every player P a number R_P representing skill.
- Defined by Logistic Curve: expected win % p given by

$$p = \frac{1}{1 + \exp(c\Delta)}$$

where $\Delta = R_P - R_O$ is the difference to your opponent's rating and c is a conversion constant.

- USCF takes $c = (\ln 10)/400$, so 200-pointse $\approx$ 75% expectation.
- **Class Units**: 2000–2200 = Expert, 2200–2400 = Master, 2400–2600 is typical of International/Senior Master and Grandmaster ranks, 2600–2800 = "Super GM,"; Carlsen 2857, 3 others over 2800, Anand 2770. Adult beginner $\approx$ 600, kids $\to$ 100.
- Komodo 11.1.3 **3414?**, Stockfish 9+ **3447?**, Houdini 6 **3410?**, Fire 6.1 **3298**...

Elo Rating System

- Named for the Hungarian-American statistician **Arpad Elo**, the system gives every player P a number R_P representing skill.
- Defined by Logistic Curve: expected win % p given by

$$p = \frac{1}{1 + \exp(c\Delta)}$$

  where $\Delta = R_P - R_O$ is the difference to your opponent's rating and c is a conversion constant.
- USCF takes $c = (\ln 10)/400$, so 200-pointse $\approx$ 75% expectation.
- **Class Units**: 2000–2200 = Expert, 2200–2400 = Master, 2400–2600 is typical of International/Senior Master and Grandmaster ranks, 2600–2800 = "Super GM,"; Carlsen 2857, 3 others over 2800, Anand 2770. Adult beginner $\approx$ 600, kids $\to$ 100.
- Komodo 11.1.3 **3414?**, Stockfish 9+ **3447?**, Houdini 6 **3410?**, Fire 6.1 **3298**... So computers $\approx$ "Class 14"—a kind of "Moore's Law."

Elo Rating System

- Named for the Hungarian-American statistician **Arpad Elo**, the system gives every player P a number R_P representing skill.
- Defined by Logistic Curve: expected win % p given by

$$p = \frac{1}{1 + \exp(c\Delta)}$$

  where $\Delta = R_P - R_O$ is the difference to your opponent's rating and c is a conversion constant.
- USCF takes $c = (\ln 10)/400$, so 200-pointse $\approx$ 75% expectation.
- **Class Units**: 2000–2200 = Expert, 2200–2400 = Master, 2400–2600 is typical of International/Senior Master and Grandmaster ranks, 2600–2800 = "Super GM,"; Carlsen 2857, 3 others over 2800, Anand 2770. Adult beginner $\approx$ 600, kids $\to$ 100.
- Komodo 11.1.3 **3414?**, Stockfish 9+ **3447?**, Houdini 6 **3410?**, Fire 6.1 **3298**... So computers $\approx$ "Class 14"—a kind of "Moore's Law."
- So **AlphaZero** $>$ **3500**?

Elo Rating System

- Named for the Hungarian-American statistician **Arpad Elo**, the system gives every player P a number R_P representing skill.
- Defined by Logistic Curve: expected win % p given by

$$p = \frac{1}{1 + \exp(c\Delta)}$$

  where $\Delta = R_P - R_O$ is the difference to your opponent's rating and c is a conversion constant.
- USCF takes $c = (\ln 10)/400$, so 200-pointse $\approx$ 75% expectation.
- **Class Units**: 2000–2200 = Expert, 2200–2400 = Master, 2400–2600 is typical of International/Senior Master and Grandmaster ranks, 2600–2800 = "Super GM,"; Carlsen 2857, 3 others over 2800, Anand 2770. Adult beginner $\approx$ 600, kids $\to$ 100.
- Komodo 11.1.3 **3414**?, Stockfish 9+ **3447**?, Houdini 6 **3410**?, Fire 6.1 **3298**... So computers $\approx$ "Class 14"—a kind of "Moore's Law."
- So **AlphaZero > 3500**? Higher than my measures of perfection...

Reducing Chess to Numbers

- Chess engines all work by *incremental search* in rounds of increasing *depth* $d = 1, 2, 3, \ldots$

Reducing Chess to Numbers

- Chess engines all work by *incremental search* in rounds of increasing *depth* $d = 1, 2, 3, \ldots$
- For each round d and legal move m_i the program outputs a value $v_{i,d}$ in units of 0.01 called *centipawns*, figuratively 100ths of a pawn value (roughly P = 1, N = 3, B = 3+, R = 5, Q = 9).

Reducing Chess to Numbers

- Chess engines all work by *incremental search* in rounds of increasing *depth* $d = 1, 2, 3, \ldots$
- For each round d and legal move m_i the program outputs a value $v_{i,d}$ in units of 0.01 called *centipawns*, figuratively 100ths of a pawn value (roughly P = 1, N = 3, B = 3+, R = 5, Q = 9).
- Values by Stockfish 6 in key Kramnik-Anand WC 2008 position:

Move	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
Nd2	103	093	087	093	027	028	000	000	056	-007	039	028	037	020	014	017	000	006	000
Bxd7	048	034	-033	-033	-013	-042	-039	-050	-025	-010	001	000	-009	-027	-018	000	000	000	000
Qg8	114	114	-037	-037	-014	-014	-022	-068	-008	-056	-042	-004	-032	000	-014	-025	-045	-045	-050
...			...			...			...			...			...			...	
Nxd4	-056	-056	-113	-071	-071	-145	-020	-006	077	052	066	040	050	051	-181	-181	-181	-213	-213

Reducing Chess to Numbers

- Chess engines all work by *incremental search* in rounds of increasing *depth* $d = 1, 2, 3, \ldots$
- For each round d and legal move m_i the program outputs a value $v_{i,d}$ in units of 0.01 called *centipawns*, figuratively 100ths of a pawn value (roughly P = 1, N = 3, B = 3+, R = 5, Q = 9).
- Values by Stockfish 6 in key Kramnik-Anand WC 2008 position:

Move	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
Nd2	103	093	087	093	027	028	000	000	056	-007	039	028	037	020	014	017	000	006	000
Bxd7	048	034	-033	-033	-013	-042	-039	-050	-025	-010	001	000	-009	-027	-018	000	000	000	000
Qg8	114	114	-037	-037	-014	-014	-022	-068	-008	-056	-042	-004	-032	000	-014	-025	-045	-045	-050
...			...			...			...			...			...			...	
Nxd4	-056	-056	-113	-071	-071	-145	-020	-006	077	052	066	040	050	051	-181	-181	-181	-213	-213

- Note that two moves have "equal-top value" (EV).

Reducing Chess to Numbers

- Chess engines all work by *incremental search* in rounds of increasing *depth* $d = 1, 2, 3, \ldots$
- For each round d and legal move m_i the program outputs a value $v_{i,d}$ in units of 0.01 called *centipawns*, figuratively 100ths of a pawn value (roughly P = 1, N = 3, B = 3+, R = 5, Q = 9).
- Values by Stockfish 6 in key Kramnik-Anand WC 2008 position:

Move	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
Nd2	103	093	087	093	027	028	000	000	056	-007	039	028	037	020	014	017	000	006	000
Bxd7	048	034	-033	-033	-013	-042	-039	-050	-025	-010	001	000	-009	-027	-018	000	000	000	000
Qg8	114	114	-037	-037	-014	-014	-022	-068	-008	-056	-042	-004	-032	000	-014	-025	-045	-045	-050
...			...			...			...			...			...			...	
Nxd4	-056	-056	-113	-071	-071	-145	-020	-006	077	052	066	040	050	051	-181	-181	-181	-213	-213

- Note that two moves have "equal-top value" (EV).
- This happens in 8–10% of positions.

Reducing Chess to Numbers

- Chess engines all work by *incremental search* in rounds of increasing *depth* $d = 1, 2, 3, \ldots$
- For each round d and legal move m_i the program outputs a value $v_{i,d}$ in units of 0.01 called *centipawns*, figuratively 100ths of a pawn value (roughly P = 1, N = 3, B = 3+, R = 5, Q = 9).
- Values by Stockfish 6 in key Kramnik-Anand WC 2008 position:

Move	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
Nd2	103	093	087	093	027	028	000	000	056	-007	039	028	037	020	014	017	000	006	000
Bxd7	048	034	-033	-033	-013	-042	-039	-050	-025	-010	001	000	-009	-027	-018	000	000	000	000
Qg8	114	114	-037	-037	-014	-014	-022	-068	-008	-056	-042	-004	-032	000	-014	-025	-045	-045	-050
...			...			...			...			...			...			...	
Nxd4	-056	-056	-113	-071	-071	-145	-020	-006	077	052	066	040	050	051	-181	-181	-181	-213	-213

- Note that two moves have "equal-top value" (EV).
- This happens in 8–10% of positions.
- *These values are* (currently) *the only chess-specific inputs.*

A Predictive Analytic Model

1. Domain: A set T of decision-making situations t.
   Chess game turns

A Predictive Analytic Model

1. Domain: A set T of decision-making situations t.
   Chess game turns
2. Inputs: Values v_i for every option at turn t.
   Computer values of moves m_i

A Predictive Analytic Model

1. Domain: A set T of decision-making situations t.
   Chess game turns
2. Inputs: Values v_i for every option at turn t.
   Computer values of moves m_i
3. Parameters: $s, c, \ldots$ denoting skills and levels.
   Trained correspondence $P(s, c, \ldots) \longleftrightarrow$ Elo rating E

A Predictive Analytic Model

1. Domain: A set T of decision-making situations t.
   Chess game turns
2. Inputs: Values v_i for every option at turn t.
   Computer values of moves m_i
3. Parameters: $s, c, \ldots$ denoting skills and levels.
   Trained correspondence $P(s, c, \ldots) \longleftrightarrow$ Elo rating E
4. Main Output: Probabilities p_i $(= p_{t,i})$ for $P(s, c, \ldots)$ to select option i (at turn t).

A Predictive Analytic Model

1. Domain: A set T of decision-making situations t.
   Chess game turns
2. Inputs: Values v_i for every option at turn t.
   Computer values of moves m_i
3. Parameters: $s, c, \dots$ denoting skills and levels.
   Trained correspondence $P(s, c, \dots) \longleftrightarrow$ Elo rating E
4. Main Output: Probabilities p_i $(= p_{t,i})$ for $P(s, c, \dots)$ to select option i (at turn t).
5. Derived Outputs:
   - **MM%, EV%, AE** and other aggregate statistics.
   - Projected confidence intervals for them—via Multinomial Bernoulli Trials plus an adjustment for correlation between consecutive turns.
   - **Intrinsic Performance Ratings** (IPRs) for the players.

How the Model Operates

- Given $s, c, \ldots$ and each legal move m_i with value v_i (at top depth), the model computes a *proxy value*

$$u_i = g_{s,c}(\delta(v_1, v_i)),$$

where $\delta(v_1, v_i)$ scales down the raw difference $v_1 - v_i$ in relation to the overall position value v_1, and $g = g_{s,c}$ is a family of curves giving $g(0) = 1$, $g(z) \to 0$.

How the Model Operates

- Given $s, c, \ldots$ and each legal move m_i with value v_i (at top depth), the model computes a *proxy value*

$$u_i = g_{s,c}(\delta(v_1, v_i)),$$

  where $\delta(v_1, v_i)$ scales down the raw difference $v_1 - v_i$ in relation to the overall position value v_1, and $g = g_{s,c}$ is a family of curves giving $g(0) = 1$, $g(z) \to 0$.
- Intuitively, $1 - u_i$ is the "perceived inferiority" of the move m_i.

How the Model Operates

- Given $s, c, \ldots$ and each legal move m_i with value v_i (at top depth), the model computes a *proxy value*

$$u_i = g_{s,c}(\delta(v_1, v_i)),$$

where $\delta(v_1, v_i)$ scales down the raw difference $v_1 - v_i$ in relation to the overall position value v_1, and $g = g_{s,c}$ is a family of curves giving $g(0) = 1$, $g(z) \to 0$.
- Intuitively, $1 - u_i$ is the "perceived inferiority" of the move m_i.
- Besides g, the model picks a function $h(p_i)$ on probabilities.
- Could be $h(p) = p$ (bad), log (good enough?), $H(p_i)$, *logit*...

How the Model Operates

- Given $s, c, \ldots$ and each legal move m_i with value v_i (at top depth), the model computes a *proxy value*
$$u_i = g_{s,c}(\delta(v_1, v_i)),$$
where $\delta(v_1, v_i)$ scales down the raw difference $v_1 - v_i$ in relation to the overall position value v_1, and $g = g_{s,c}$ is a family of curves giving $g(0) = 1$, $g(z) \to 0$.
- Intuitively, $1 - u_i$ is the "perceived inferiority" of the move m_i.
- Besides g, the model picks a function $h(p_i)$ on probabilities.
- Could be $h(p) = p$ (bad), log (good enough?), $H(p_i)$, *logit*. . .
- The **Original Main Equation:**
$$\frac{h(p_i)}{h(p_1)} = u_i$$

How the Model Operates

- Given $s, c, \dots$ and each legal move m_i with value v_i (at top depth), the model computes a *proxy value*

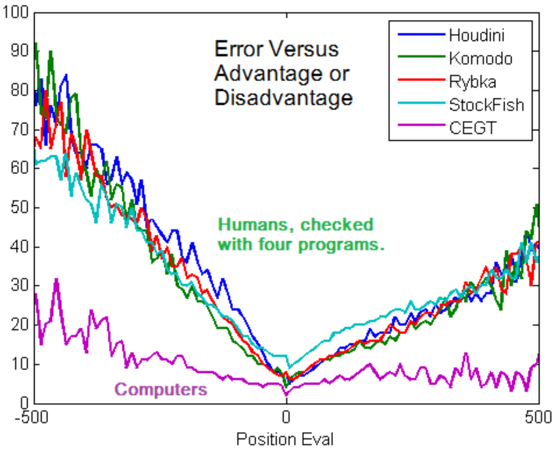
$$u_i = g_{s,c}(\delta(v_1, v_i)),$$

  where $\delta(v_1, v_i)$ scales down the raw difference $v_1 - v_i$ in relation to the overall position value v_1, and $g = g_{s,c}$ is a family of curves giving $g(0) = 1$, $g(z) \to 0$.
- Intuitively, $1 - u_i$ is the "perceived inferiority" of the move m_i.
- Besides g, the model picks a function $h(p_i)$ on probabilities.
- Could be $h(p) = p$ (bad), log (good enough?), $H(p_i)$, *logit*...
- The **Original Main Equation:**

$$\frac{h(p_i)}{h(p_1)} = u_i = \exp(-\left(\frac{\delta(v_1, v_i)}{s}\right)^c).$$

- *Any such value-based model entails* $v_1 = v_2 \implies p_1 = p_2$.

Why the Scaling?



Scaling $\delta(u, v) = \int_{x=u}^{x=v} \frac{1}{1+Cx}\,dx$ (for $x > 0$) levels out differences.

Five Expectations—and Curveballs/Googlies:

1. Equal values yield equal behavior.

Five Expectations—and Curveballs/Googlies:

1. Equal values yield equal behavior.
2. Unbiased data-gathering yields unbiased data.

Five Expectations—and Curveballs/Googlies:

1. Equal values yield equal behavior.
2. Unbiased data-gathering yields unbiased data.
3. Biases that are obvious will show up in the data.

Five Expectations—and Curveballs/Googlies:

1. Equal values yield equal behavior.
2. Unbiased data-gathering yields unbiased data.
3. Biases that are obvious will show up in the data.
4. If Y is a continuous function of X, then a small change in X produces a small change in Y.

Five Expectations—and Curveballs/Googlies:

1. Equal values yield equal behavior.
2. Unbiased data-gathering yields unbiased data.
3. Biases that are obvious will show up in the data.
4. If Y is a continuous function of X, then a small change in X produces a small change in Y.
5. Factors whose insignificance you demonstrated will stay insignificant when you have 10x–100x data.

Five Expectations—and Curveballs/Googlies:

1. Equal values yield equal behavior.
2. Unbiased data-gathering yields unbiased data.
3. Biases that are obvious will show up in the data.
4. If Y is a continuous function of X, then a small change in X produces a small change in Y.
5. Factors whose insignificance you demonstrated will stay insignificant when you have 10x–100x data.
6. *OK, 1.5:* Secondary aspects of standard library routines called by your data-gathering engines won't disturb the above expectations.

Five Expectations—and Curveballs/Googlies:

1. Equal values yield equal behavior.
2. Unbiased data-gathering yields unbiased data.
3. Biases that are obvious will show up in the data.
4. If Y is a continuous function of X, then a small change in X produces a small change in Y.
5. Factors whose insignificance you demonstrated will stay insignificant when you have 10x–100x data.
6. *OK, 1.5:* Secondary aspects of standard library routines called by your data-gathering engines won't disturb the above expectations.

Googlies: *Data points have histories*,

Five Expectations—and Curveballs/Googlies:

1. Equal values yield equal behavior.
2. Unbiased data-gathering yields unbiased data.
3. Biases that are obvious will show up in the data.
4. If Y is a continuous function of X, then a small change in X produces a small change in Y.
5. Factors whose insignificance you demonstrated will stay insignificant when you have 10x–100x data.
6. *OK, 1.5:* Secondary aspects of standard library routines called by your data-gathering engines won't disturb the above expectations.

Googlies: *Data points have histories*, *notionally* unbiased/ continuous/... need not imply *factually* unbiased/ continuous/...,

Five Expectations—and Curveballs/Googlies:

1. Equal values yield equal behavior.
2. Unbiased data-gathering yields unbiased data.
3. Biases that are obvious will show up in the data.
4. If Y is a continuous function of X, then a small change in X produces a small change in Y.
5. Factors whose insignificance you demonstrated will stay insignificant when you have 10x–100x data.
6. *OK, 1.5:* Secondary aspects of standard library routines called by your data-gathering engines won't disturb the above expectations.

Googlies: *Data points have histories*, *notionally* unbiased/ continuous/... need not imply *factually* unbiased/ continuous/..., and *zero-sigma* results can be artifacts too.

X and Y and Z

- $X =$ *values of chess moves.*

X and Y and Z

- X = *values of chess moves.*
- Y = *performance indicators* of (human) players:

X and Y and Z

- X = *values of chess moves.*
- Y = *performance indicators* of (human) players:
  - **MM%** = how often the player chose the move listed first by the engine in value order.
  - **EV%** = how often the player chose the first move or one of equal value, as happens in 8–10% of positions.
  - **ASD** = the average scaled difference in value between the player's chosen move m_i and the engine's first move m_1.

X and Y and Z

- X = *values of chess moves.*
- Y = *performance indicators* of (human) players:
  - **MM%** = how often the player chose the move listed first by the engine in value order.
  - **EV%** = how often the player chose the first move or one of equal value, as happens in 8–10% of positions.
  - **ASD** = the average scaled difference in value between the player's chosen move m_i and the engine's first move m_1.
- Z = Elo rating

X and Y and Z

- X = *values of chess moves.*
- Y = *performance indicators* of (human) players:
  - **MM%** = how often the player chose the move listed first by the engine in value order.
  - **EV%** = how often the player chose the first move or one of equal value, as happens in 8–10% of positions.
  - **ASD** = the average scaled difference in value between the player's chosen move m_i and the engine's first move m_1.
- Z = Elo rating
- The 2-parameter model is fitted simply by setting the projected MM% and ASD equal to the sample means.

X and Y and Z

- X = *values of chess moves.*
- Y = *performance indicators* of (human) players:
  - **MM%** = how often the player chose the move listed first by the engine in value order.
  - **EV%** = how often the player chose the first move or one of equal value, as happens in 8–10% of positions.
  - **ASD** = the average scaled difference in value between the player's chosen move m_i and the engine's first move m_1.
- Z = Elo rating
- The 2-parameter model is fitted simply by setting the projected MM% and ASD equal to the sample means.
- Resulting EV estimator is biased "conservatively" (against false positives).

The Data: Old and New

- **Old:** Over 6 million moves of **Multi-PV** data: $>$ 500 GB.

The Data: Old and New

- **Old:** Over 6 million moves of **Multi-PV** data: $>$ 500 GB.
- Over 120 million moves of **Single-PV** data: $>$ 200 GB

The Data: Old and New

- **Old:** Over 6 million moves of **Multi-PV** data: $>$ 500 GB.
- Over 120 million moves of **Single-PV** data: $>$ 200 GB
- $=$ 350 million pages of text data at 2k/page.

The Data: Old and New

- **Old:** Over 6 million moves of **Multi-PV** data: $>$ 500 GB.
- Over 120 million moves of **Single-PV** data: $>$ 200 GB
- $=$ 350 million pages of text data at 2k/page.
- All taken on two quad-core home-style PC's plus a laptop using the GUI. This involved **retaining hashed move values** between game turns—which is the normal playing mode and only GUI option.

The Data: Old and New

- **Old:** Over 6 million moves of **Multi-PV** data: $>$ 500 GB.
- Over 120 million moves of **Single-PV** data: $>$ 200 GB
- $=$ 350 million pages of text data at 2k/page.
- All taken on two quad-core home-style PC's plus a laptop using the GUI. This involved **retaining hashed move values** between game turns—which is the normal playing mode and only GUI option.
- **New—using CCR:** Every published high-level game since 2014 in **Single-PV** mode.

The Data: Old and New

- **Old:** Over 6 million moves of **Multi-PV** data: > 500 GB.
- Over 120 million moves of **Single-PV** data: > 200 GB
- $=$ 350 million pages of text data at 2k/page.
- All taken on two quad-core home-style PC's plus a laptop using the GUI. This involved **retaining hashed move values** between game turns—which is the normal playing mode and only GUI option.
- **New—using CCR:** Every published high-level game since 2014 in **Single-PV** mode.
- **Master training sets** of 1.15 million moves by players of Elo ratings 1025, 1050, 1075, 1100, ... (stepping by 25) ..., 2750, 2775, 2800, all in **Multi-PV mode**.

The Data: Old and New

- **Old:** Over 6 million moves of **Multi-PV** data: $>$ 500 GB.
- Over 120 million moves of **Single-PV** data: $>$ 200 GB
- $=$ 350 million pages of text data at 2k/page.
- All taken on two quad-core home-style PC's plus a laptop using the GUI. This involved **retaining hashed move values** between game turns—which is the normal playing mode and only GUI option.
- **New—using CCR:** Every published high-level game since 2014 in **Single-PV** mode.
- **Master training sets** of 1.15 million moves by players of Elo ratings 1025, 1050, 1075, 1100, . . . (stepping by 25) . . . , 2750, 2775, 2800, all in **Multi-PV mode**.
- Taken with Komodo 10 and Stockfish 7, *all years since 1971*.

First Googly: An "ESP Test"

- In 8%–10% of positions, engine gives the top two moves the same value.

First Googly: An "ESP Test"

- In 8%–10% of positions, engine gives the top two moves the same value.
- Even more often, *some* pair of moves in the top 10 (say) will end up tied. Conditioned on one of them having been played, let us invite humans to guess **which move is listed first by the program**.

First Googly: An "ESP Test"

- In 8%–10% of positions, engine gives the top two moves the same value.
- Even more often, *some* pair of moves in the top 10 (say) will end up tied. Conditioned on one of them having been played, let us invite humans to guess **which move is listed first by the program**.
- The values are identical to the engine: it would not matter to the quality of the output which one the engine listed first. The values give no human reason to prefer one over the other.

First Googly: An "ESP Test"

- In 8%–10% of positions, engine gives the top two moves the same value.
- Even more often, *some* pair of moves in the top 10 (say) will end up tied. Conditioned on one of them having been played, let us invite humans to guess **which move is listed first by the program**.
- The values are identical to the engine: it would not matter to the quality of the output which one the engine listed first. The values give no human reason to prefer one over the other.
- So this is a kind of ESP test.

First Googly: An "ESP Test"

- In 8%–10% of positions, engine gives the top two moves the same value.
- Even more often, *some* pair of moves in the top 10 (say) will end up tied. Conditioned on one of them having been played, let us invite humans to guess **which move is listed first by the program**.
- The values are identical to the engine: it would not matter to the quality of the output which one the engine listed first. The values give no human reason to prefer one over the other.
- So this is a kind of ESP test. *How well do humans perform on it?*

First Googly: An "ESP Test"

- In 8%–10% of positions, engine gives the top two moves the same value.
- Even more often, *some* pair of moves in the top 10 (say) will end up tied. Conditioned on one of them having been played, let us invite humans to guess **which move is listed first by the program**.
- The values are identical to the engine: it would not matter to the quality of the output which one the engine listed first. The values give no human reason to prefer one over the other.
- So this is a kind of ESP test. *How well do humans perform on it?*
- PEAR—Princeton Engineering Anomalies Research—notorious ESP project.

First Googly: An "ESP Test"

- In 8%–10% of positions, engine gives the top two moves the same value.
- Even more often, *some* pair of moves in the top 10 (say) will end up tied. Conditioned on one of them having been played, let us invite humans to guess **which move is listed first by the program**.
- The values are identical to the engine: it would not matter to the quality of the output which one the engine listed first. The values give no human reason to prefer one over the other.
- So this is a kind of ESP test. *How well do humans perform on it?*
- PEAR—Princeton Engineering Anomalies Research—notorious ESP project.
- PEAR did 10,000s–100,000s of trials, trying to judge significance of deviations like 50.1% or even 50.01%.

First Googly: An "ESP Test"

- In 8%–10% of positions, engine gives the top two moves the same value.
- Even more often, *some* pair of moves in the top 10 (say) will end up tied. Conditioned on one of them having been played, let us invite humans to guess **which move is listed first by the program**.
- The values are identical to the engine: it would not matter to the quality of the output which one the engine listed first. The values give no human reason to prefer one over the other.
- So this is a kind of ESP test. *How well do humans perform on it?*
- PEAR—Princeton Engineering Anomalies Research—notorious ESP project.
- PEAR did 10,000s–100,000s of trials, trying to judge significance of deviations like 50.1% or even 50.01%.
- How about *my* ESP test??

Sensitivity—Plotting Y against X

Conditioned on one of the top two moves being played, if their values (old: Rybka 3, depth 13; new: Stockfish and Komodo, depths 19+) differ by...:

1. **0.01**, the higher move is played 53–55% of the time.

Sensitivity—Plotting Y against X

Conditioned on one of the top two moves being played, if their values (old: Rybka 3, depth 13; new: Stockfish and Komodo, depths 19+) differ by...:

1. **0.01**, the higher move is played 53–55% of the time.
2. **0.02**, the higher move is played 58–59% of the time.

Sensitivity—Plotting Y against X

Conditioned on one of the top two moves being played, if their values (old: Rybka 3, depth 13; new: Stockfish and Komodo, depths 19+) differ by...:

1. **0.01**, the higher move is played 53–55% of the time.
2. **0.02**, the higher move is played 58–59% of the time.
3. **0.03**, the higher move is played 60–61% of the time.

Sensitivity—Plotting Y against X

Conditioned on one of the top two moves being played, if their values (old: Rybka 3, depth 13; new: Stockfish and Komodo, depths 19+) differ by...:

1. **0.01**, the higher move is played 53–55% of the time.
2. **0.02**, the higher move is played 58–59% of the time.
3. **0.03**, the higher move is played 60–61% of the time.
4. **0.00**, the higher move is played

Sensitivity—Plotting Y against X

Conditioned on one of the top two moves being played, if their values (old: Rybka 3, depth 13; new: Stockfish and Komodo, depths 19+) differ by...:

1. **0.01**, the higher move is played 53–55% of the time.
2. **0.02**, the higher move is played 58–59% of the time.
3. **0.03**, the higher move is played 60–61% of the time.
4. **0.00**, the higher move is played 55–59% of the time.

Sensitivity—Plotting Y against X

Conditioned on one of the top two moves being played, if their values (old: Rybka 3, depth 13; new: Stockfish and Komodo, depths 19+) differ by...:

1. **0.01**, the higher move is played 53–55% of the time.
2. **0.02**, the higher move is played 58–59% of the time.
3. **0.03**, the higher move is played 60–61% of the time.
4. **0.00**, the higher move is played 55–59% of the time.

- Last is not a typo—see post "**When is a Law Natural?**"

Sensitivity—Plotting Y against X

Conditioned on one of the top two moves being played, if their values (old: Rybka 3, depth 13; new: Stockfish and Komodo, depths 19+) differ by...:

1. **0.01**, the higher move is played 53–55% of the time.
2. **0.02**, the higher move is played 58–59% of the time.
3. **0.03**, the higher move is played 60–61% of the time.
4. **0.00**, the higher move is played 55–59% of the time.

- Last is not a typo—see post "**When is a Law Natural?**"
- Similar 58%-42% split seen for any pair of tied moves, all Elo over 2000, down to 55%–45% for Elo 1050. What can explain it?

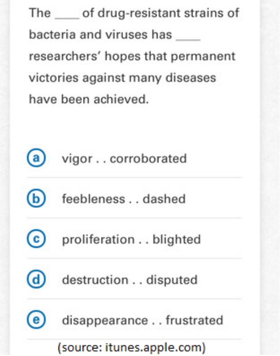
Sensitivity—Plotting Y against X

Conditioned on one of the top two moves being played, if their values (old: Rybka 3, depth 13; new: Stockfish and Komodo, depths 19+) differ by...:

1. **0.01**, the higher move is played 53–55% of the time.
2. **0.02**, the higher move is played 58–59% of the time.
3. **0.03**, the higher move is played 60–61% of the time.
4. **0.00**, the higher move is played 55–59% of the time.

- Last is not a typo—see post "**When is a Law Natural?**"
- Similar 58%-42% split seen for any pair of tied moves, all Elo over 2000, down to 55%–45% for Elo 1050. What can explain it?
- Relation to slime molds and other "semi-Brownian" systems?

History and “Swing” over Increasing Depths



The ____ of drug-resistant strains of bacteria and viruses has ____ researchers’ hopes that permanent victories against many diseases have been achieved.

(a) vigor . . corroborated

(b) feebleness . . dashed

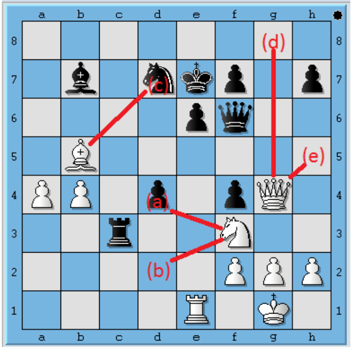
(c) proliferation . . blighted

(d) destruction . . disputed

(e) disappearance . . frustrated

(source: itunes.apple.com)

=



Move	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
Nd2	103	093	087	093	027	028	000	000	056	-007	039	028	037	020	014	017	000	006	000
Bxd7	048	034	-033	-033	-013	-042	-039	-050	-025	-010	001	000	-009	-027	-018	000	000	000	000
Qg8	114	114	-037	-037	-014	-014	-022	-068	-008	-056	-042	-004	-032	000	-014	-025	-045	-045	-050
...			...			...			...			...			...			...	
Nxd4	-056	-056	-113	-071	-071	-145	-020	-006	077	052	066	040	050	051	-181	-181	-181	-213	-213

Measuring "Swing" and Complexity and Difficulty

- Non-Parapsychological Explanation:

Measuring "Swing" and Complexity and Difficulty

- Non-Parapsychological Explanation: *Stable* Library Sorting.

Measuring "Swing" and Complexity and Difficulty

- Non-Parapsychological Explanation: *Stable* Library Sorting.
- Chess engines sort moves from last depth to schedule next round of search.

Measuring “Swing” and Complexity and Difficulty

- Non-Parapsychological Explanation: *Stable* Library Sorting.
- Chess engines sort moves from last depth to schedule next round of search.
- Stable $\rightarrow$ lower move jumps to 1st only with *strictly higher* value.

Measuring "Swing" and Complexity and Difficulty

- Non-Parapsychological Explanation: *Stable* Library Sorting.
- Chess engines sort moves from last depth to schedule next round of search.
- Stable $\rightarrow$ lower move jumps to 1st only with *strictly higher* value.
- Lead moves tend to have been higher at lower depths. Lower move "swings up."

Measuring "Swing" and Complexity and Difficulty

- Non-Parapsychological Explanation: *Stable* Library Sorting.
- Chess engines sort moves from last depth to schedule next round of search.
- Stable $\rightarrow$ lower move jumps to 1st only with *strictly higher* value.
- Lead moves tend to have been higher at lower depths. Lower move "swings up."
- Formulate numerical measure of swing "up" and "down" (a trap).

Measuring "Swing" and Complexity and Difficulty

- Non-Parapsychological Explanation: *Stable* Library Sorting.
- Chess engines sort moves from last depth to schedule next round of search.
- Stable $\rightarrow$ lower move jumps to 1st only with *strictly higher* value.
- Lead moves tend to have been higher at lower depths. Lower move "swings up."
- Formulate numerical measure of swing "up" and "down" (a trap).
- When best move swings up **4.0–5.0** versus **0.0–1.0**, players rated 2700+ find it only **30%** versus **70%**.

Measuring "Swing" and Complexity and Difficulty

- Non-Parapsychological Explanation: *Stable* Library Sorting.
- Chess engines sort moves from last depth to schedule next round of search.
- Stable $\rightarrow$ lower move jumps to 1st only with *strictly higher* value.
- Lead moves tend to have been higher at lower depths. Lower move "swings up."
- Formulate numerical measure of swing "up" and "down" (a trap).
- When best move swings up **4.0–5.0** versus **0.0–1.0**, players rated 2700+ find it only **30%** versus **70%**.
- **Huge differences** $\Longrightarrow$ corrections to the **main equation**.

Measuring "Swing" and Complexity and Difficulty

- Non-Parapsychological Explanation: *Stable* Library Sorting.
- Chess engines sort moves from last depth to schedule next round of search.
- Stable $\rightarrow$ lower move jumps to 1st only with *strictly higher* value.
- Lead moves tend to have been higher at lower depths. Lower move "swings up."
- Formulate numerical measure of swing "up" and "down" (a trap).
- When best move swings up **4.0–5.0** versus **0.0–1.0**, players rated 2700+ find it only **30%** versus **70%**.
- **Huge differences** $\Longrightarrow$ corrections to the **main equation**.
- Will also separate *performance* and *prediction* in the model.

The New Model—as of today!

- My old idea was to extend the main equation to a weighted linear combinationover depths governed by a "peak depth" parameter d:

$$\frac{h(p_i)}{h(p_1)} = 1 - x_i$$

The New Model—as of today!

- My old idea was to extend the main equation to a weighted linear combinationover depths governed by a "peak depth" parameter d:

$$\frac{h(p_i)}{h(p_1)} = 1 - x_i = u_i = \sum_{j=1}^{D} w_j \exp(-\left(\frac{\delta(v_{1,j}, v_{i,j})}{s}\right)^c),$$

The New Model—as of today!

- My old idea was to extend the main equation to a weighted linear combinationover depths governed by a "peak depth" parameter d:

$$\frac{h(p_i)}{h(p_1)} = 1 - x_i = u_i = \sum_{j=1}^{D} w_j \exp(-\left(\frac{\delta(v_{1,j}, v_{i,j})}{s}\right)^c),$$

- Led to horrible fitting landscape, many local minima. . .

The New Model—as of today!

- My old idea was to extend the main equation to a weighted linear combinationover depths governed by a "peak depth" parameter d:

$$\frac{h(p_i)}{h(p_1)} = 1 - x_i = u_i = \sum_{j=1}^{D} w_j \exp(-\left(\frac{\delta(v_{1,j}, v_{i,j})}{s}\right)^c),$$

- Led to horrible fitting landscape, many local minima...
- Simpler idea advocated by my student Tamal Biswas: first define some concrete measure of the "swing" of move m_i, *viz.*

$$sw(i) = \frac{1}{D}\sum_{j=1}^{D}(\delta_{i,j} - \delta_{i,D}).$$

The New Model—as of today!

- My old idea was to extend the main equation to a weighted linear combinationover depths governed by a "peak depth" parameter d:

$$\frac{h(p_i)}{h(p_1)} = 1 - x_i = u_i = \sum_{j=1}^{D} w_j \exp(-\left(\frac{\delta(v_{1,j}, v_{i,j})}{s}\right)^c),$$

- Led to horrible fitting landscape, many local minima. . .
- Simpler idea advocated by my student Tamal Biswas: first define some concrete measure of the "swing" of move m_i, *viz.*

$$sw(i) = \frac{1}{D} \sum_{j=1}^{D} (\delta_{i,j} - \delta_{i,D}).$$

- Then introduce a new parameter h (for nautical "heave") and fit:

$$\frac{h(p_i)}{h(p_1)} = 1 - x_i$$

The New Model—as of today!

- My old idea was to extend the main equation to a weighted linear combinationover depths governed by a "peak depth" parameter d:

$$\frac{h(p_i)}{h(p_1)} = 1 - x_i = u_i = \sum_{j=1}^{D} w_j \exp(-\left(\frac{\delta(v_{1,j}, v_{i,j})}{s}\right)^c),$$

- Led to horrible fitting landscape, many local minima...
- Simpler idea advocated by my student Tamal Biswas: first define some concrete measure of the "swing" of move m_i, *viz.*

$$sw(i) = \frac{1}{D}\sum_{j=1}^{D}(\delta_{i,j} - \delta_{i,D}).$$

- Then introduce a new parameter h (for nautical "heave") and fit:

$$\frac{h(p_i)}{h(p_1)} = 1 - x_i = \exp(-\left(\frac{\delta(v_1, v_i) + h \cdot sw(i)}{s}\right)^c).$$

How the Model is Fitted

How the Model is Fitted

- Given s, c, h, compute proxy values $u_i = g_{s,c,h}(v_1, v_i)$.

How the Model is Fitted

- Given s, c, h, compute proxy values $u_i = g_{s,c,h}(v_1, v_i)$.
- Solve for $p_1, \ldots, p_i, \ldots$ subject to $\sum_i p_i = 1$ such that

$$\frac{h(p_i)}{h(p_1)} = u_i; \qquad \text{specific choice:} \qquad \frac{\log(1/p_1)}{\log(1/p_i)} = u_i.$$

How the Model is Fitted

- Given s, c, h, compute proxy values $u_i = g_{s,c,h}(v_1, v_i)$.
- Solve for $p_1, \dots, p_i, \dots$ subject to $\sum_i p_i = 1$ such that

$$\frac{h(p_i)}{h(p_1)} = u_i; \qquad \text{specific choice:} \qquad \frac{\log(1/p_1)}{\log(1/p_i)} = u_i.$$

- This gives $P_{s,c,h} : p_i = p_1^{1/u_i}$ for each i.

How the Model is Fitted

- Given s, c, h, compute proxy values $u_i = g_{s,c,h}(v_1, v_i)$.
- Solve for $p_1, \ldots, p_i, \ldots$ subject to $\sum_i p_i = 1$ such that

$$\frac{h(p_i)}{h(p_1)} = u_i; \qquad \text{specific choice:} \qquad \frac{\log(1/p_1)}{\log(1/p_i)} = u_i.$$

- This gives $P_{s,c,h} : p_i = p_1^{1/u_i}$ for each i.
- No closed form? Hence inner regression to find $\{p_i\}$ that we will *memoize*.

How the Model is Fitted

- Given s, c, h, compute proxy values $u_i = g_{s,c,h}(v_1, v_i)$.
- Solve for $p_1, \ldots, p_i, \ldots$ subject to $\sum_i p_i = 1$ such that

$$\frac{h(p_i)}{h(p_1)} = u_i; \qquad \text{specific choice:} \qquad \frac{\log(1/p_1)}{\log(1/p_i)} = u_i.$$

- This gives $P_{s,c,h} : p_i = p_1^{1/u_i}$ for each i.
- No closed form? Hence inner regression to find $\{p_i\}$ that we will *memoize*.
- Outer regression applies $P_{s,c,h}$ to generate projected MM%, EV%, ASD.

How the Model is Fitted

- Given s, c, h, compute proxy values $u_i = g_{s,c,h}(v_1, v_i)$.
- Solve for $p_1, \dots, p_i, \dots$ subject to $\sum_i p_i = 1$ such that

$$\frac{h(p_i)}{h(p_1)} = u_i; \qquad \text{specific choice:} \qquad \frac{\log(1/p_1)}{\log(1/p_i)} = u_i.$$

- This gives $P_{s,c,h} : p_i = p_1^{1/u_i}$ for each i.
- No closed form? Hence inner regression to find $\{p_i\}$ that we will *memoize*.
- Outer regression applies $P_{s,c,h}$ to generate projected MM%, EV%, ASD.
- Regress over s, c, h to fit to sample means. Expensive!

How the Model is Fitted

- Given s, c, h, compute proxy values $u_i = g_{s,c,h}(v_1, v_i)$.
- Solve for $p_1, \dots, p_i, \dots$ subject to $\sum_i p_i = 1$ such that

$$\frac{h(p_i)}{h(p_1)} = u_i; \qquad \text{specific choice:} \qquad \frac{\log(1/p_1)}{\log(1/p_i)} = u_i.$$

- This gives $P_{s,c,h} : p_i = p_1^{1/u_i}$ for each i.
- No closed form? Hence inner regression to find $\{p_i\}$ that we will *memoize*.
- Outer regression applies $P_{s,c,h}$ to generate projected MM%, EV%, ASD.
- Regress over s, c, h to fit to sample means. Expensive!
- But appears to work well: the 2nd-best, 3rd-best, 4th-best move frequencies fall into place all down the line.

How the Model is Fitted

- Given s, c, h, compute proxy values $u_i = g_{s,c,h}(v_1, v_i)$.
- Solve for $p_1, \dots, p_i, \dots$ subject to $\sum_i p_i = 1$ such that

$$\frac{h(p_i)}{h(p_1)} = u_i; \qquad \text{specific choice:} \qquad \frac{\log(1/p_1)}{\log(1/p_i)} = u_i.$$

- This gives $P_{s,c,h} : p_i = p_1^{1/u_i}$ for each i.
- No closed form? Hence inner regression to find $\{p_i\}$ that we will *memoize*.
- Outer regression applies $P_{s,c,h}$ to generate projected MM%, EV%, ASD.
- Regress over s, c, h to fit to sample means. Expensive!
- But appears to work well: the 2nd-best, 3rd-best, 4th-best move frequencies fall into place all down the line.
- Another "natural law"? At least indicates model is basically right...

Second Googly

- **Single-PV** = normal playing (and cheating?) mode.

Second Googly

- **Single-PV** = normal playing (and cheating?) mode.
- **Multi-PV** values needed for main model equation.

Second Googly

- **Single-PV** = normal playing (and cheating?) mode.
- **Multi-PV** values needed for main model equation.
- Does difference matter for **MM%, EV%, ASD**?

Second Googly

- **Single-PV** = normal playing (and cheating?) mode.
- **Multi-PV** values needed for main model equation.
- Does difference matter for **MM%**, **EV%**, **ASD**?
- *Value* of first move seems unaffected. However (plotting Y vs. Z):

Second Googly

- **Single-PV** = normal playing (and cheating?) mode.
- **Multi-PV** values needed for main model equation.
- Does difference matter for **MM%, EV%, ASD**?
- *Value* of first move seems unaffected. However (plotting Y vs. Z):

> Human players of all rating levels have 2–3% higher MM% and EV% to the Single-PV mode.

Second Googly

- **Single-PV** = normal playing (and cheating?) mode.
- **Multi-PV** values needed for main model equation.
- Does difference matter for **MM%, EV%, ASD**?
- *Value* of first move seems unaffected. However (plotting Y vs. Z):

> Human players of all rating levels have 2–3% higher MM% and EV% to the Single-PV mode.

Thus my model is a biased predictor of MM% in Single-PV mode. Bias avoided by conducting test entirely in Multi-PV mode (arguably conservative). Why might this happen?

Second Googly

- **Single-PV** = normal playing (and cheating?) mode.
- **Multi-PV** values needed for main model equation.
- Does difference matter for **MM%, EV%, ASD**?
- *Value* of first move seems unaffected. However (plotting Y vs. Z):

> Human players of all rating levels have 2–3% higher MM% and EV% to the Single-PV mode.

Thus my model is a biased predictor of MM% in Single-PV mode. Bias avoided by conducting test entirely in Multi-PV mode (arguably conservative). Why might this happen?

> Single-PV mode maximally retards "late-blooming" moves from jumping ahead in the stable sort.

Third Googly: No Such Thing As Being "In Form"?

Third Googly: No Such Thing As Being “In Form”?

- I routinely “screen” 5,000+ games per week in SinglePV mode.

Third Googly: No Such Thing As Being "In Form"?

- I routinely "screen" 5,000+ games per week in SinglePV mode.
- Not my full model, just a simple "Raw Outlier Index" (ROI) from each player's MM%, ASD, and rating.

Third Googly: No Such Thing As Being "In Form"?

- I routinely "screen" 5,000+ games per week in SinglePV mode.
- Not my full model, just a simple "Raw Outlier Index" (ROI) from each player's MM%, ASD, and rating.
- Large "Open" tournaments have hundreds of players in a "Swiss System" (not knockout) format.

Third Googly: No Such Thing As Being "In Form"?

- I routinely "screen" 5,000+ games per week in SinglePV mode.
- Not my full model, just a simple "Raw Outlier Index" (ROI) from each player's MM%, ASD, and rating.
- Large "Open" tournaments have hundreds of players in a "Swiss System" (not knockout) format.
- The top 10-20 or so games are on auto-recording boards that can broadcast moves.

Third Googly: No Such Thing As Being "In Form"?

- I routinely "screen" 5,000+ games per week in SinglePV mode.
- Not my full model, just a simple "Raw Outlier Index" (ROI) from each player's MM%, ASD, and rating.
- Large "Open" tournaments have hundreds of players in a "Swiss System" (not knockout) format.
- The top 10-20 or so games are on auto-recording boards that can broadcast moves.
- Some tournament staffs type up the rest of the games from scoresheets submitted by players.

Third Googly: No Such Thing As Being "In Form"?

- I routinely "screen" 5,000+ games per week in SinglePV mode.
- Not my full model, just a simple "Raw Outlier Index" (ROI) from each player's MM%, ASD, and rating.
- Large "Open" tournaments have hundreds of players in a "Swiss System" (not knockout) format.
- The top 10-20 or so games are on auto-recording boards that can broadcast moves.
- Some tournament staffs type up the rest of the games from scoresheets submitted by players.
- Others do not—those tournaments I mark with `Avail` in filenames.

Third Googly: No Such Thing As Being "In Form"?

- I routinely "screen" 5,000+ games per week in SinglePV mode.
- Not my full model, just a simple "Raw Outlier Index" (ROI) from each player's MM%, ASD, and rating.
- Large "Open" tournaments have hundreds of players in a "Swiss System" (not knockout) format.
- The top 10-20 or so games are on auto-recording boards that can broadcast moves.
- Some tournament staffs type up the rest of the games from scoresheets submitted by players.
- Others do not—those tournaments I mark with `Avail` in filenames.
- After Round 1, the top boards have people who have done well in recent rounds.

Third Googly: No Such Thing As Being "In Form"?

- I routinely "screen" 5,000+ games per week in SinglePV mode.
- Not my full model, just a simple "Raw Outlier Index" (ROI) from each player's MM%, ASD, and rating.
- Large "Open" tournaments have hundreds of players in a "Swiss System" (not knockout) format.
- The top 10-20 or so games are on auto-recording boards that can broadcast moves.
- Some tournament staffs type up the rest of the games from scoresheets submitted by players.
- Others do not—those tournaments I mark with `Avail` in filenames.
- After Round 1, the top boards have people who have done well in recent rounds.
- Hence `Avail` files skew massively toward "in form" players.

Third Googly: No Such Thing As Being "In Form"?

- I routinely "screen" 5,000+ games per week in SinglePV mode.
- Not my full model, just a simple "Raw Outlier Index" (ROI) from each player's MM%, ASD, and rating.
- Large "Open" tournaments have hundreds of players in a "Swiss System" (not knockout) format.
- The top 10-20 or so games are on auto-recording boards that can broadcast moves.
- Some tournament staffs type up the rest of the games from scoresheets submitted by players.
- Others do not—those tournaments I mark with `Avail` in filenames.
- After Round 1, the top boards have people who have done well in recent rounds.
- Hence `Avail` files skew massively toward "in form" players.
- But *no significant difference* in ROI

Third Googly: No Such Thing As Being "In Form"?

- I routinely "screen" 5,000+ games per week in SinglePV mode.
- Not my full model, just a simple "Raw Outlier Index" (ROI) from each player's MM%, ASD, and rating.
- Large "Open" tournaments have hundreds of players in a "Swiss System" (not knockout) format.
- The top 10-20 or so games are on auto-recording boards that can broadcast moves.
- Some tournament staffs type up the rest of the games from scoresheets submitted by players.
- Others do not—those tournaments I mark with `Avail` in filenames.
- After Round 1, the top boards have people who have done well in recent rounds.
- Hence `Avail` files skew massively toward "in form" players.
- But *no significant difference* in ROI (if anything, the opposite).

Third Googly: No Such Thing As Being "In Form"?

- I routinely "screen" 5,000+ games per week in SinglePV mode.
- Not my full model, just a simple "Raw Outlier Index" (ROI) from each player's MM%, ASD, and rating.
- Large "Open" tournaments have hundreds of players in a "Swiss System" (not knockout) format.
- The top 10-20 or so games are on auto-recording boards that can broadcast moves.
- Some tournament staffs type up the rest of the games from scoresheets submitted by players.
- Others do not—those tournaments I mark with `Avail` in filenames.
- After Round 1, the top boards have people who have done well in recent rounds.
- Hence `Avail` files skew massively toward "in form" players.
- But *no significant difference* in ROI (if anything, the opposite).
- No "Hot Hand" in chess?

Third Googly: No Such Thing As Being "In Form"?

- I routinely "screen" 5,000+ games per week in SinglePV mode.
- Not my full model, just a simple "Raw Outlier Index" (ROI) from each player's MM%, ASD, and rating.
- Large "Open" tournaments have hundreds of players in a "Swiss System" (not knockout) format.
- The top 10-20 or so games are on auto-recording boards that can broadcast moves.
- Some tournament staffs type up the rest of the games from scoresheets submitted by players.
- Others do not—those tournaments I mark with `Avail` in filenames.
- After Round 1, the top boards have people who have done well in recent rounds.
- Hence `Avail` files skew massively toward "in form" players.
- But *no significant difference* in ROI (if anything, the opposite).
- No "Hot Hand" in chess? Or maybe nerves offset form?...

Fourth Googly: A "Firewall at Zero

Surely Y = the frequency of large errors ("blunders") ought to be continuous as a function of X = the value of the position.

Fourth Googly: A “Firewall at Zero

Surely Y = the frequency of large errors (“blunders”) ought to be continuous as a function of X = the value of the position. But:

Fourth Googly: A "Firewall at Zero

Surely Y = the frequency of large errors ("blunders") ought to be continuous as a function of X = the value of the position. But:

Elo 2600–2850	Komodo 9.3				Stockfish 7 (modified)			
Value range	#pos	d10	d15	d20	#pos	d10	d15	d20
-0.30 to -0.21	4,710	9	13	18	4,193	13	10	14
-0.20 to -0.11	5,048	11	10	13	5,177	6	9	11
-0.20 to -0.01	4,677	11	13	16	5,552	8	9	16
0.00 exactly	9,168	24	25	28	9,643	43	40	38
+0.01 to +0.10	4,283	6	1	2	5,705	8	3	2
+0.11 to +0.20	5,198	7	5	3	5,495	10	5	3
+0.21 to +0.30	5,200	7	2	1	4,506	3	4	2

Reason evidently that 0.00 is a big *basin of attraction* in complex positions that may force one side to give perpetual check or force repetitions to avoid losing.

Fourth Googly: A “Firewall at Zero

Surely Y = the frequency of large errors (“blunders”) ought to be continuous as a function of X = the value of the position. But:

Elo 2600–2850	Komodo 9.3				Stockfish 7 (modified)			
Value range	#pos	d10	d15	d20	#pos	d10	d15	d20
-0.30 to -0.21	4,710	9	13	18	4,193	13	10	14
-0.20 to -0.11	5,048	11	10	13	5,177	6	9	11
-0.20 to -0.01	4,677	11	13	16	5,552	8	9	16
0.00 exactly	9,168	24	25	28	9,643	43	40	38
+0.01 to +0.10	4,283	6	1	2	5,705	8	3	2
+0.11 to +0.20	5,198	7	5	3	5,495	10	5	3
+0.21 to +0.30	5,200	7	2	1	4,506	3	4	2

Reason evidently that 0.00 is a big *basin of attraction* in complex positions that may force one side to give perpetual check or force repetitions to avoid losing. Safety net provided $v_1 > 0$ but absent when $v_1 < 0$.

Fourth Googly: A "Firewall at Zero

Surely Y = the frequency of large errors ("blunders") ought to be continuous as a function of X = the value of the position. But:

Elo 2600–2850	Komodo 9.3				Stockfish 7 (modified)			
Value range	#pos	d10	d15	d20	#pos	d10	d15	d20
-0.30 to -0.21	4,710	9	13	18	4,193	13	10	14
-0.20 to -0.11	5,048	11	10	13	5,177	6	9	11
-0.20 to -0.01	4,677	11	13	16	5,552	8	9	16
0.00 exactly	9,168	24	25	28	9,643	43	40	38
+0.01 to +0.10	4,283	6	1	2	5,705	8	3	2
+0.11 to +0.20	5,198	7	5	3	5,495	10	5	3
+0.21 to +0.30	5,200	7	2	1	4,506	3	4	2

Reason evidently that 0.00 is a big *basin of attraction* in complex positions that may force one side to give perpetual check or force repetitions to avoid losing. Safety net provided $v_1 > 0$ but absent when $v_1 < 0$. Failure to charge adequately for large "notional errors."

Fifth Googly—Clearing Hash Does Matter

- Retaining hash apparently also retards "later-blooming" moves.

Fifth Googly—Clearing Hash Does Matter

- Retaining hash apparently also retards "later-blooming" moves.
- Effect only 0.25–0.35%, not 2–3%, but significant now.

Fifth Googly—Clearing Hash Does Matter

- Retaining hash apparently also retards "later-blooming" moves.
- Effect only 0.25–0.35%, not 2–3%, but significant now.
- Clearing is better for scientific reproducibility but further from actual playing conditions.

Fifth Googly—Clearing Hash Does Matter

- Retaining hash apparently also retards "later-blooming" moves.
- Effect only 0.25–0.35%, not 2–3%, but significant now.
- Clearing is better for scientific reproducibility but further from actual playing conditions.

Thus my original "simple and self-evident" model needs substantial adjustment for all of these factors—to say nothing of factors like the scaling which I caught at the beginning. . .

Fifth Googly—Clearing Hash Does Matter

- Retaining hash apparently also retards "later-blooming" moves.
- Effect only 0.25–0.35%, not 2–3%, but significant now.
- Clearing is better for scientific reproducibility but further from actual playing conditions.

> Thus my original "simple and self-evident" model needs substantial adjustment for all of these factors—to say nothing of factors like the scaling which I caught at the beginning...

To conclude on a philosophic note:

Fifth Googly—Clearing Hash Does Matter

- Retaining hash apparently also retards “later-blooming” moves.
- Effect only 0.25–0.35%, not 2–3%, but significant now.
- Clearing is better for scientific reproducibility but further from actual playing conditions.

Thus my original “simple and self-evident” model needs substantial adjustment for all of these factors—to say nothing of factors like the scaling which I caught at the beginning. . .

To conclude on a philosophic note: “Big Data” is critiqued for abandoning *theory*. Need not be so—my chess model is theory-driven and “severely underfitted.”

Fifth Googly—Clearing Hash Does Matter

- Retaining hash apparently also retards "later-blooming" moves.
- Effect only 0.25–0.35%, not 2–3%, but significant now.
- Clearing is better for scientific reproducibility but further from actual playing conditions.

> Thus my original "simple and self-evident" model needs substantial adjustment for all of these factors—to say nothing of factors like the scaling which I caught at the beginning. . .

To conclude on a philosophic note: "Big Data" is critiqued for abandoning *theory*. Need not be so—my chess model is theory-driven and "severely underfitted." *But theory cannot abandon data*

Fifth Googly—Clearing Hash Does Matter

- Retaining hash apparently also retards "later-blooming" moves.
- Effect only 0.25–0.35%, not 2–3%, but significant now.
- Clearing is better for scientific reproducibility but further from actual playing conditions.

> Thus my original "simple and self-evident" model needs substantial adjustment for all of these factors—to say nothing of factors like the scaling which I caught at the beginning...

To conclude on a philosophic note: "Big Data" is critiqued for abandoning *theory*. Need not be so—my chess model is theory-driven and "severely underfitted." *But theory cannot abandon data*—nor a full understanding of the *history* and *hidden biases* it may embody.

A Sixth Lesson: Weighting and Bootstrap

- This does not involve my model, only chess program evaluation functions $v = v(p)$ of positions p.

A Sixth Lesson: Weighting and Bootstrap

- This does not involve my model, only chess program evaluation functions $v = v(p)$ of positions p.
- Graph v versus scoring frequency $e(v)$ from positions of value v.

A Sixth Lesson: Weighting and Bootstrap

- This does not involve my model, only chess program evaluation functions $v = v(p)$ of positions p.
- Graph v versus scoring frequency $e(v)$ from positions of value v.
- Fantastic logistic fit $e(v) = A + \frac{1-2A}{1+\exp(-Bv)}$, B depends on rating.

A Sixth Lesson: Weighting and Bootstrap

- This does not involve my model, only chess program evaluation functions $v = v(p)$ of positions p.
- Graph v versus scoring frequency $e(v)$ from positions of value v.
- Fantastic logistic fit $e(v) = A + \frac{1-2A}{1+\exp(-Bv)}$, B depends on rating.
- Has $R^2 > 0.9999999$ but what are the error bars on B?

A Sixth Lesson: Weighting and Bootstrap

- This does not involve my model, only chess program evaluation functions $v = v(p)$ of positions p.
- Graph v versus scoring frequency $e(v)$ from positions of value v.
- Fantastic logistic fit $e(v) = A + \frac{1-2A}{1+\exp(-Bv)}$, B depends on rating.
- Has $R^2 > 0.9999999$ but what are the error bars on B?
- Can weight regression by number N_v of positions of value v. Concentrated near $v = 0$.

A Sixth Lesson: Weighting and Bootstrap

- This does not involve my model, only chess program evaluation functions $v = v(p)$ of positions p.
- Graph v versus scoring frequency $e(v)$ from positions of value v.
- Fantastic logistic fit $e(v) = A + \frac{1-2A}{1+\exp(-Bv)}$, B depends on rating.
- Has $R^2 > 0.9999999$ but what are the error bars on B?
- Can weight regression by number N_v of positions of value v. Concentrated near $v = 0$.
- But cross-check by **Bootstrap** of B is off by factor of 2.

A Sixth Lesson: Weighting and Bootstrap

- This does not involve my model, only chess program evaluation functions $v = v(p)$ of positions p.
- Graph v versus scoring frequency $e(v)$ from positions of value v.
- Fantastic logistic fit $e(v) = A + \frac{1-2A}{1+\exp(-Bv)}$, B depends on rating.
- Has $R^2 > 0.9999999$ but what are the error bars on B?
- Can weight regression by number N_v of positions of value v. Concentrated near $v = 0$.
- But cross-check by **Bootstrap** of B is off by factor of 2.
- Instead of "X-side" weighting, can use $1/\sigma$ of "Y-side" instead.

A Sixth Lesson: Weighting and Bootstrap

- This does not involve my model, only chess program evaluation functions $v = v(p)$ of positions p.
- Graph v versus scoring frequency $e(v)$ from positions of value v.
- Fantastic logistic fit $e(v) = A + \frac{1-2A}{1+\exp(-Bv)}$, B depends on rating.
- Has $R^2 > 0.9999999$ but what are the error bars on B?
- Can weight regression by number N_v of positions of value v. Concentrated near $v = 0$.
- But cross-check by **Bootstrap** of B is off by factor of 2.
- Instead of "X-side" weighting, can use $1/\sigma$ of "Y-side" instead.
- Not $\sim \sqrt{N_v}/2$ but rather $\sim \sqrt{e(v)(1 - e(v))N_v}$. Different in tails.

A Sixth Lesson: Weighting and Bootstrap

- This does not involve my model, only chess program evaluation functions $v = v(p)$ of positions p.
- Graph v versus scoring frequency $e(v)$ from positions of value v.
- Fantastic logistic fit $e(v) = A + \frac{1-2A}{1+\exp(-Bv)}$, B depends on rating.
- Has $R^2 > 0.9999999$ but what are the error bars on B?
- Can weight regression by number N_v of positions of value v. Concentrated near $v = 0$.
- But cross-check by **Bootstrap** of B is off by factor of 2.
- Instead of "X-side" weighting, can use $1/\sigma$ of "Y-side" instead.
- Not $\sim \sqrt{N_v}/2$ but rather $\sim \sqrt{e(v)(1-e(v))N_v}$. Different in tails.
- Eliminates the discrepancy from bootstrap results.

Seventh Seal: Cross-Validation and Fitting Horror

- The fitting of s, c, h can be done in many other ways. . . .

Seventh Seal: Cross-Validation and Fitting Horror

- The fitting of s, c, h can be done in many other ways. . . .
- The model is "severely underfitted"—theory-heavy.

Seventh Seal: Cross-Validation and Fitting Horror

- The fitting of s, c, h can be done in many other ways. . . .
- The model is "severely underfitted"—theory-heavy.
- How well does your favorite fitting method work?

Seventh Seal: Cross-Validation and Fitting Horror

- The fitting of s, c, h can be done in many other ways....
- The model is "severely underfitted"—theory-heavy.
- How well does your favorite fitting method work?
- Maximum Likelihood Estimation: minimize $\sum_t \log(1/p_{t,i_t})$ where i_t is the index of the played move at each game turn t.

Seventh Seal: Cross-Validation and Fitting Horror

- The fitting of s, c, h can be done in many other ways. . . .
- The model is "severely underfitted"—theory-heavy.
- How well does your favorite fitting method work?
- Maximum Likelihood Estimation: minimize $\sum_t \log(1/p_{t,i_t})$ where i_t is the index of the played move at each game turn t.
- Performs relatively poorly—a known phenomenon with underfitting.

Seventh Seal: Cross-Validation and Fitting Horror

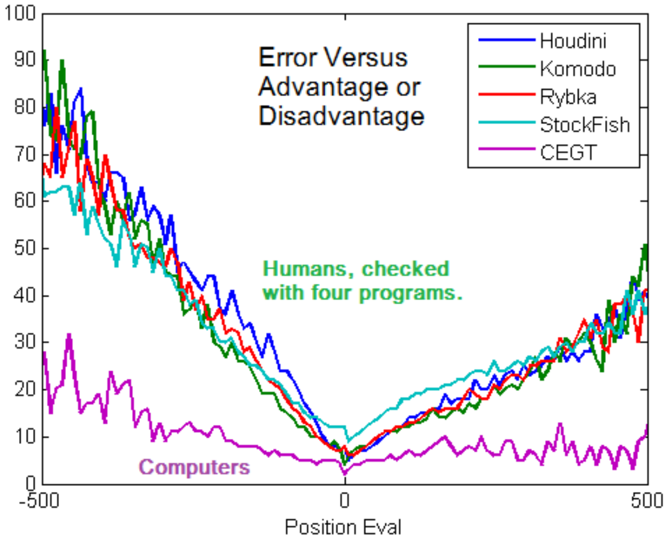
- The fitting of s, c, h can be done in many other ways. . . .
- The model is "severely underfitted"—theory-heavy.
- How well does your favorite fitting method work?
- Maximum Likelihood Estimation: minimize $\sum_t \log(1/p_{t,i_t})$ where i_t is the index of the played move at each game turn t.
- Performs relatively poorly—a known phenomenon with underfitting.
- In the 3- and 4-parameter models, *chaos breaks loose.* **Literally.**

Seventh Seal: Cross-Validation and Fitting Horror

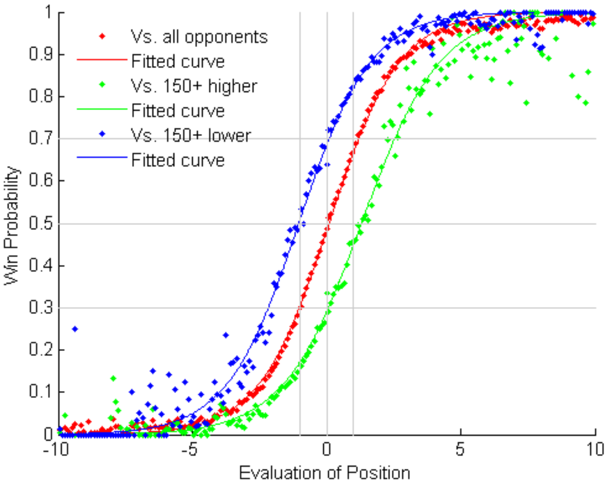
- The fitting of s, c, h can be done in many other ways. . . .
- The model is "severely underfitted"—theory-heavy.
- How well does your favorite fitting method work?
- Maximum Likelihood Estimation: minimize $\sum_t \log(1/p_{t,i_t})$ where i_t is the index of the played move at each game turn t.
- Performs relatively poorly—a known phenomenon with underfitting.
- In the 3- and 4-parameter models, *chaos breaks loose.* **Literally.**
- **Segue to posts on the *Gödel's Lost Letter* blog:**

"Unskewing the Election"
"Stopped Watches and Data Analytics"

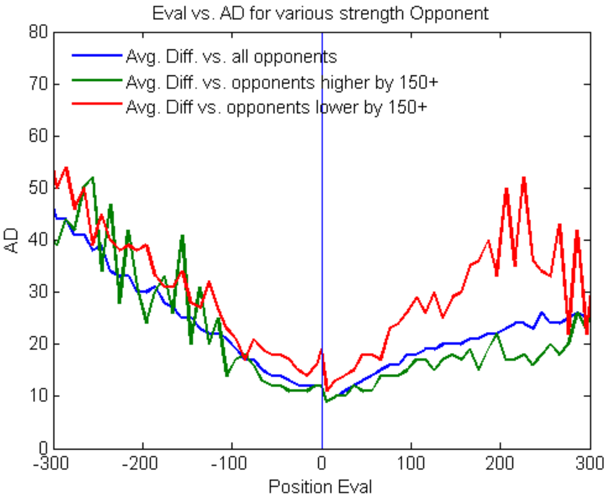
Extras: Human Versus Computer Phenomena



Human Versus Computer Phenomena



Eval-Error Curve With Unequal Players



Computer and Freestyle IPRs

Analyzed Ratings of Computer Engine Grand Tournament (on commodity PCs) and PAL/CSS Freestyle in 2007–08, plus the Thoresen Chess Engines Competition (16-core) Nov–Dec. 2013.

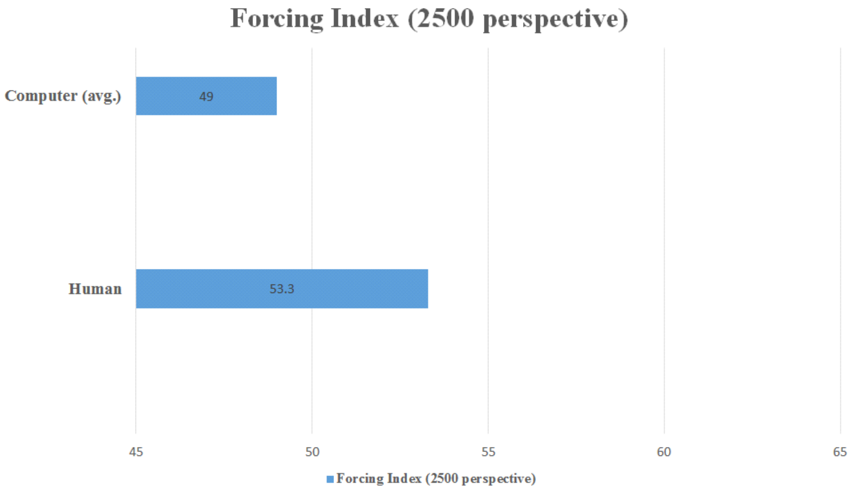
Event	Rating	2σ range	#gm	#moves
CEGT g1,50	3009	2962–3056	42	4,212
CEGT g25,26	2963	2921–3006	42	5,277
PAL/CSS 5ch	3102	3051–3153	45	3,352
PAL/CSS 6ch	3086	3038–3134	45	3,065
PAL/CSS 8ch	3128	3083–3174	39	3,057
TCEC 2013	3083	3062–3105	90	11,024

Computer and Freestyle IPRs—To Move 60

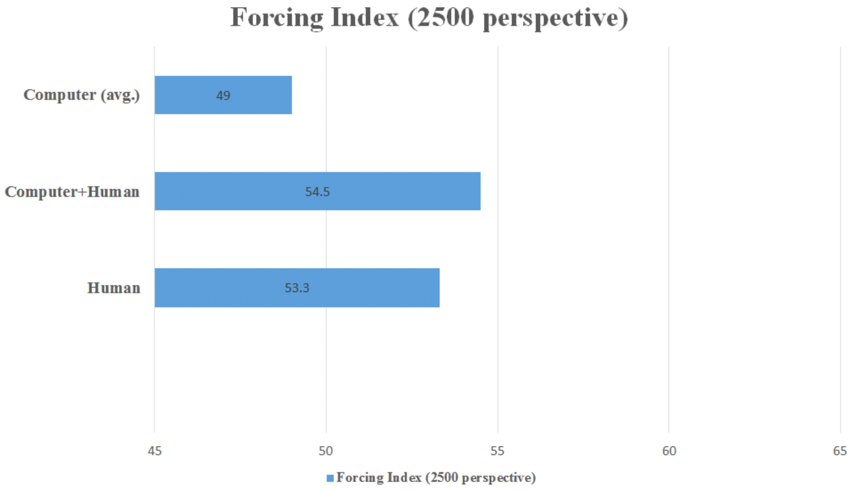
Computer games can go very long in dead drawn positions. TCEC uses a cutoff but CEGT did not. Human-led games tend to climax (well) before Move 60. This comparison halves the difference to CEGT, otherwise similar:

Sample set	Rating	2σ range	#gm	#moves
CEGT all	2985	2954–3016	84	9,489
PAL/CSS all	3106	3078–3133	129	9,474
TCEC 2013	3083	3062–3105	90	11,024
CEGT to60	3056	3023–3088	84	7,010
PAL/CSS to60	3112	3084–3141	129	8,744
TCEC to60	3096	3072–3120	90	8,184

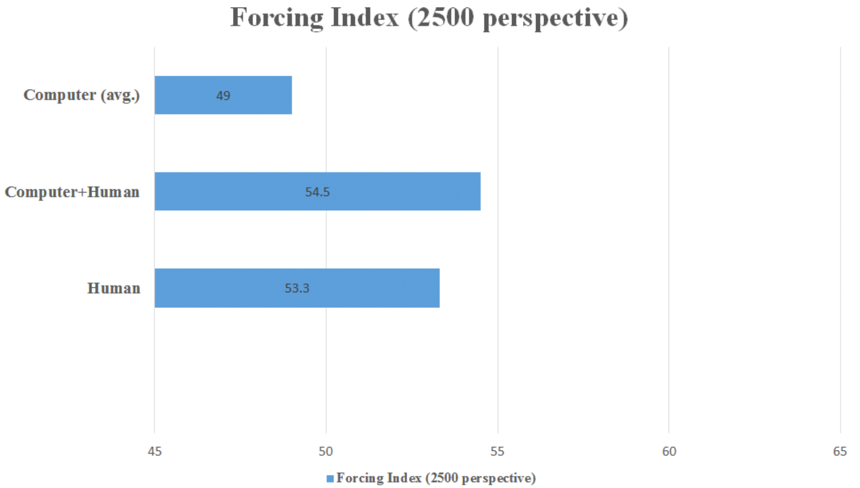
Degrees of Forcing Play



Add Human-Computer Tandems

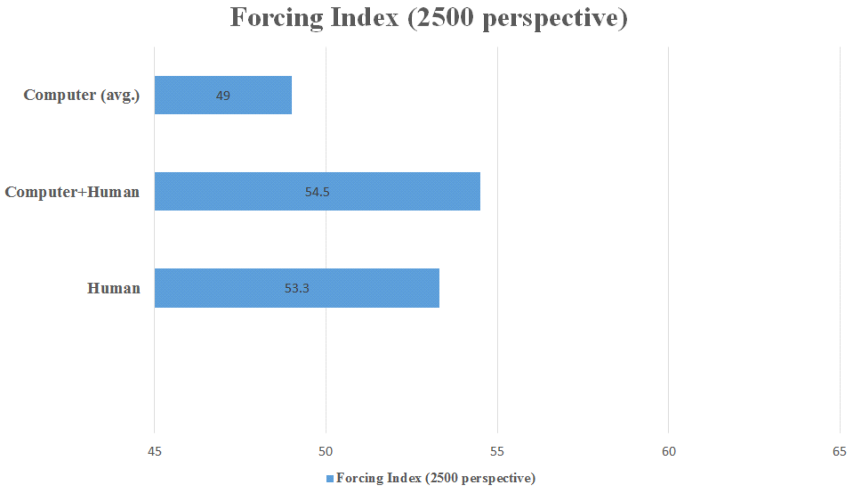


Add Human-Computer Tandems



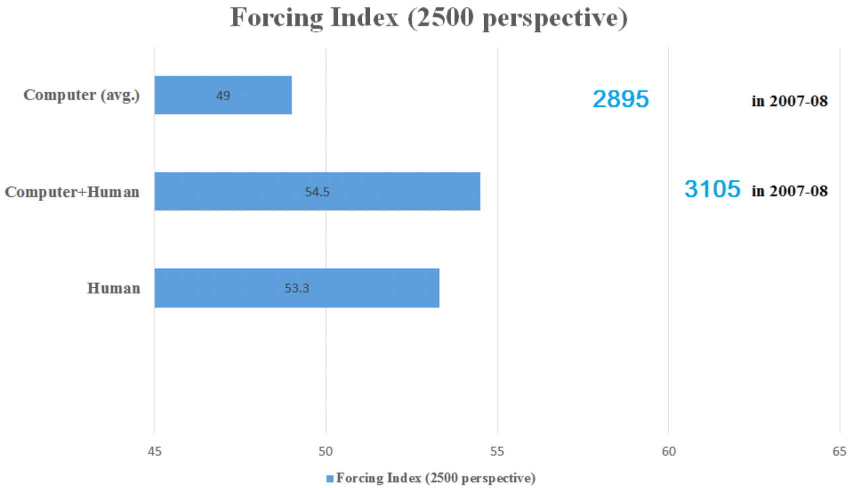
Evidently the humans called the shots.

Add Human-Computer Tandems



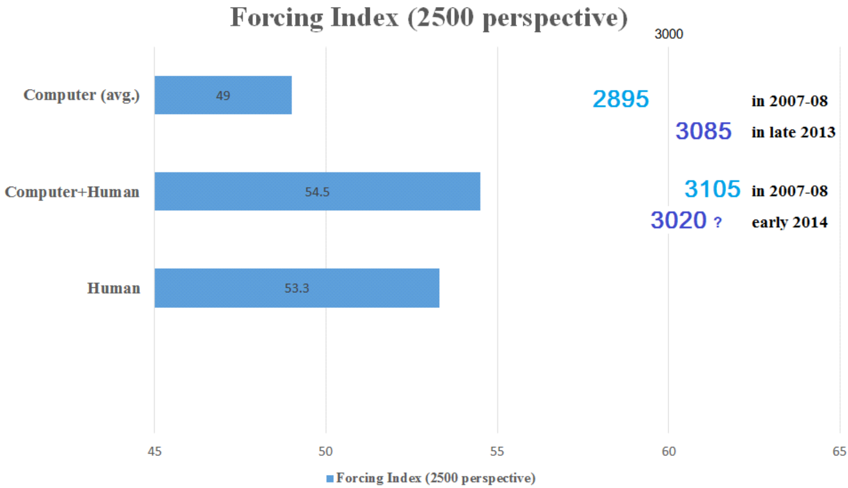
Evidently the humans called the shots. But how did they play?

2007–08 Freestyle Performance



Adding 210 Elo was significant. Forcing but good teamwork.

2014 Freestyle Tournament Performance



Tandems had marginally better W-L, but quality not clear...