Has the Advent of Chess Engines Increased the Speed with which Top Players Become More Accurate?

Since Garry Kasparov’s loss to IBM’s Deep Blue in 1997, it has been widely accepted that chess computers (engines) are stronger than even the top human players. While some at the time bemoaned the fact that chess had been ruined due to our newfound inferiority, the past 25 years have shown that players have embraced engines as an invaluable analysis tool and generator of chess ideas and motifs.

Traditional engines work in a similar fashion to our own brains, but at a much larger scale: they calculate lines up to 20 moves deep and assess a position as better for white or black, as measured in centipawns. [[1]](#endnote-1) This allows us to see continuations that we might have otherwise not been able to exhaustively evaluate and provides us new ways of thinking about attacking and defending in chess.

It would be incorrect to say that before the widespread adoption of engines chess players were not improving over time. *Opening theory*, the compendium of sets of moves at the beginning of a game which are deemed optimal, has been constantly developing since the 18th century. As a result, players have become increasingly accurate over the early stages of a game, since they can commit to memory the best ways to play.

The question I would like to explore is the following: have chess engines significantly accelerated the rate with which humans play optimal moves? The way I planned to examine this question is by taking all the games of the top five players of several generations,[[2]](#endnote-2) and look at their accuracy over time. By starting with players who were born well before the advent of engines, we can see how overall accuracy was trending over time before this new technology. We can then compare these players to younger talent and see whether their accuracy trends vary between different parts of their careers.

I will collect the game data from Chessbase, a paid database subscription service (to which I already have access) of almost 9 million games played in official tournament settings. This will ensure that our data is good and will prevent me from having to scrounge the various resources to find all the games I would like to analyze. Given the parameters of the players and games I would like to look at I laid out above, I approximate that I will end up with ~60,000 games. If we assume an average game length of 40 moves (or 80 total moves; likely an underestimate), I will have to analyze over 2.4 million total moves. Even if we do not start engine analysis at the beginning of the game (for reasons explained below), we can assume that my program will have to run through almost 2 million data points. I would like to an engine at a depth of 30 plies (i.e. the engine will analyze branches of up to 15 moves per player), as this is generally considered sufficient depth for the engine to find the best line, as well as provide an accurate evaluation of the position. It is for these reasons, the size of the data set as well as the computational complexity of the project, that I would like to use the Dove Cluster to execute my code. As an example, I ran a sample game (I believe it was 38 moves long), through engine analysis on my personal computer which has 32GB of RAM, 8 cores running at 4.2GHz, and that took me almost 5 minutes to run a complete analysis of the game. If we extrapolate that speed out to all 60,000 games, we can deduce that it will take an inordinate amount of time to finish running.

The way I plan on measuring accuracy in a game is by analyzing the mean difference between the optimal move as suggested by the engine and the move made by the player. I also plan on incorporating a “positional complexity” variable into my analysis, which will help control for the difficulties in finding the best move in a particular position. In simpler positions, it is typically the case that there are several (sometimes upwards of 5) moves that will produce roughly similar objective evaluations, whereas in complicated attacking/defending positions, one wrong move can spell disaster. Therefore, it is important that we account for this nuance, as we do not want to disproportionately affect our accuracy calculation to favor those who play simpler chess, as there are inherently fewer chances for mistakes in dryer games. This complexity variable will be calculated every move, and it will be defined as the difference between the first move recommended by the engine minus the second, plus the square root of the difference between the second move recommended by the engine minus the third. We put less emphasis on the second difference, as sharp positions typically have one move which is highly superior to the rest (e.g. the first, second, and third best moves might be +6.5, +2, and +0.6, respectively).

I plan on using Python to gather my engine evaluation data, as the “chess” package has everything that I will need. Specifically, the two subpackages, “chess.pgn” and “chess.engine” allow for the interpretation of chess games which are fed in the form of PGNs (Portable Game Notation), as well as analysis of this information through a chess engine of the users choice. I plan on using Stockfish version 15, as it is somewhat of the industry standard, and it is open source and easy to work with. It also has an estimated Elo strength of ~3700, which makes it significantly stronger than any human player ever by over 800 points, so its positional assessments should be very accurate.

I intend on starting the analysis not on the first move of a game, but rather somewhere after the tenth move. This would help eliminate bias towards modern players, as their knowledge of opening theory is larger than their older counterparts. Each game will receive a score based on this analysis, and we can compare scores over time, both within a generation and between several. To attempt to demonstrate a change in the rate of improvement due to computers, we can construct a model for those players whose games were uninfluenced by computers (e.g. games of players before 2000). To do this, we can construct an “aggregate accuracy frontier” (i.e. the average accuracy of each player in their games for a given age) for each generation of players, and analyze any changes between those curves. Ideally, this will give us a baseline for the average improvement of players over time without the assistance of engines. We can then conduct the same analysis for generations of players who have access to computers during their lifetimes and compare the rate of change of their improvement to their predecessors. (There are several players who started their careers out without computers and have since integrated them into their routines, and so looking at their trajectories will also be interesting. Viswanathan Anand, former World Champion, and current top 10 player who was born in 1969, will be very fascinating to look at, since a significant portion of his career came before Kasparov’s loss to Deep Blue.)

While the question outlined in this proposal does not directly pertain to the Fed’s mission, I believe it can contribute to SRA’s broader goals. If we view chess as a system in which players are continuously trying to optimize their decisions, and we think of engines as a helpful technology (not too big of a stretch, I think), then we can reframe this question as one about the impacts of new technologies on decision-making capabilities. This generalized inquiry is almost certainly applicable to topics in which SRA has more direct interest, such as consumer education on questions about credit.

1. In chess, pieces are generally accepted as having certain intrinsic value given their ability to move. Pawns have a value of 1 (or 100 centipawns), knights and bishops 3, rooks 5, queens 9, and the king is invaluable. [↑](#endnote-ref-1)
2. A generation in the scope of chess is generally regarded as 6 years. The way I have selected these players is by picking the five players with the highest peak rating within +/- 2 years of a particular age (e.g. for the 24 year old category, the five peak players between the ages of 22 and 26). [↑](#endnote-ref-2)