THE IMPORTANCE OF (EXPONENTIALLY MORE) COMPUTING POWER

Neil C. Thompson[1*], **Shuning Ge**[2], **Gabriel F. Manso**[3]

[1]MIT Computer Science and A.I. Lab,
MIT Initiative on the Digital Economy, Cambridge, MA USA
[2]MIT, Cambridge MA, USA
[3]FGA, University of Brasilia, Brasilia, Brazil

[*]To whom correspondence should be addressed; E-mail: neil_t@mit.edu.

ABSTRACT

Denizens of Silicon Valley have called Moore's Law "the most important graph in human history," and economists have found that Moore's Law-powered I.T. revolution has been one of the most important sources of national productivity growth. But data substantiating these claims tend to either be abstracted — for example by examining spending on I.T., rather than I.T. itself — or anecdotal. In this paper, we assemble direct quantitative evidence of the impact that computing power has had on five domains: two computing bellwethers (Chess and Go), and three economically important applications (weather prediction, protein folding, and oil exploration). Computing power explains 49%-94% of the performance improvements in these domains. But whereas economic theory typically assumes a power law relationship between inputs and outputs, we find that an *exponential* increase in computing power is needed to get *linear* improvements in these outcomes. This helps clarify why the exponential growth of computing power from Moore's Law has been so important for progress, and why performance improvements across many domains are becoming economically tenuous as Moore's Law breaks down.

Keywords Computing Power · Moore's Law · Productivity, Computer Chess · Computer Go · Protein Folding · Weather Prediction · Oil Exploration

1 Introduction

The key question at the heart of this paper is how more powerful computers are improving outcomes across society. We analyze this question by examining the growing use of computing power increases across five key application areas and then estimating those production functions. We find that systems designers have paid handsomely to grow computing power at a rate faster than the underlying hardware progress, but that these investments have paid off in the form of more capable, better-performing systems. Collectively, our results highlight the important role that exponential improvement in computing power have had for generating progress across diverse applications.

Production functions, be they macroeconomic models of the economy or microeconomic models of individual agents or firms, attempt to model and parameterize how changes in key inputs produce changes in output. Traditional models, for example [1, 2], focused on two inputs: labor and capital, which are mutually complementary, but independently face decreasing marginal returns. But not all capital is the same, and researchers have argued that I.T. capital should be split from other forms of capital [3, 4], consistent with the popular view of data-driven business, where I.T. capital is more, not less, important to businesses with large amounts of other capital (e.g. computer-driven planning is more important for FedEx because it has so many trucks). Some modern growth models now explicitly recognize this elevated role for I.T. in production functions and its complementarity with other forms of capital [5, 6, 7].

At the macro-level, the elevation of I.T. capital as primary input in production functions has been validated by studies showing that I.T. has contributed more than one-third of all improvement in productivity since 1974 [8], as well as earlier studies showing that IT contributed nearly all the improvements in total factor productivity from in the late 1990s [9, 10]. The contributions of I.T. also extend beyond the economics, to include other measures of well-being [11].

Studies at the micro-level have found similarly strong effects on the importance of I.T. on firm productivity [12, 13, 14, 15, 16]. For example Brynjolfsson and Hitt [3] showed I.T. capital, as distinct from other types of capital, contributed significantly to marginal firm output and Thompson showed that a substantial fraction of growth in firm total factor productivity after 2005 could be explained by better abilities to harness modern computer chips [17]. Most studies, however, take an abstracted view of computing capital, measuring inputs in dollars spent rather than in the productive capacity of what was bought. As Devaraj and Kholi [18] pointed out, this is inferior to measuring I.T. in the form of actual usage. And while this same critique could be made for almost all input factors to production functions, the distinction between spend and productive capacity is more strongly divergent for computing power because of exponential increases in computing power per dollar[1]. This mismatch is evidenced, for example, by the finding that $1 in computer hardware spend is associated with $10 of increased firm value [20]).

In this paper, we also study the micro-foundations of how I.T. capital improves performance, but we do it in the natural units of computer processors: computing power, as measured by the number of operations that the processor can perform. This is very much in the spirit of Thompson, Greenewald, Lee, and Manso [21], which shows that many of the most-important improvements in machine learning are heavily dependent on improvements in computing power. In this paper, we extend this type of analysis to other areas, and show that a much broader conclusion can be reached: progress across many areas of computing is dependent on exponential increases in computing power. To reach this conclusion, we gather detailed records on the performance and usage of computing power across five domains.

Chess and Go are important bellwethers for computing performance because both were traditionally viewed as areas of human expertise. Therefore, progress against human acumen could be used to track the development of programs that could play these games. For example, chess master David Levy said "Until 1977, there seemed to be no point in my playing a formal challenge match against any chess program because none of them were good enough, but when [the program] CHESS 4.5 began doing well... it was time for me to defend the human race against the coming invasion." [22]. Of the two games, Go is much harder. In 1997, astrophysicist Piet Hut, from the Institute for Advanced Study in New Jersey, told the New York Times (in retrospect incorrectly) that "It may be a hundred years before a computer beats humans at Go — maybe even longer." [23]. Because these two games are such bellwethers, they have attracted substantial attention from computer scientists which has led to a broad exploration of computational algorithms, hardware and software approaches for playing. As such, they present a promising way to study the importance of computing power for progress.

We also consider the impact of computing power in three economically-important settings: weather prediction, protein folding, and oil exploration. Weather prediction has far-reaching economic consequences, from agriculture to transportation, from military deployment to disaster preparedness. Oil exploration is a difficult, often dirty process, but one that is crucial to our fossil-fuel based economies. Doing it better can save enormous investment costs and avoid unnecessary environmental damage. Protein folding is an emerging area and is expected to be important for drug-discovery because it allows predictive modelling, rather than expensive clinical testing, to help determine which drugs will be effective.

Across all of these different domains, we find that computing power is an important part of the production function. In weather prediction, increasing computing power by 10× improves 3-day-ahead weather predictions by one-third of a degree. Benefits from such improvement would redound broadly. The literature [24] estimates that weather prediction is worth $31.5B[2] to the U.S. economy per year [25]. In oil exploration, we find substantial differences in the effect that additional computing yields for different companies, perhaps because of differing geologies. But, for example, BP's drilling success rate goes up by 43 percentage points with each 10× increase in the amount of computing used (but where drilling using the same amount of computing gets harder by 14 ppt per year as easier wells become less available). To put that in context, drilling an onshore well is estimated to cost $4.9M - $8.3M depending on the depth [26], so better predictions of which wells will be successful is very valuable. In protein folding, we find that the simulated match to real molecules improves by 6.7 points (in the 0-to-100 GTD-TS score) for each 10x increase in the computing power used. Protein folding is widely expected to dramatically accelerate the pace at which diseases and therapeutics are understood and treated [27] and is estimated to have a market size of $2.6B in 2021 [28].

[1] Official statistics do attempt to address this issue via deflators, for example based on the SPECInt computing benchmark, but these make the strong implicit assumption that computing power has the same value in different domains [19] which we later show is problematic.

[2] 40.8 B in 2021 dollars.

We see similarly large effects from increases in computing power on Chess and Go. In Chess, a 10× increase in computing power correlates with an increase of 242-point Elo – half the point difference between Masters from Grandmasters. In Go, we find a similar result with a 10x increase in computing power leading to a 246-point Elo improvement.

Perhaps as interesting as our overall findings about the importance of computing, is our ability to contrast the contributions that computing power is making to improved performance to those arising from other sources. Using an analysis-of-variance approach, we find that computing power explains 49-94% of the variation in output. Put another way, for most of these applications, increases in computing power are at least as important as all other factors put together.

Our findings are particularly important at a time when computer progress is slowing, and thus the ability to get improvements in performance is getting harder [29, 30]. Perhaps even more worrisome, if sources of computer improvement (such as Moore's Law) are running out, then the cost of improvement will rise proportionally to computing power increases. But, since exponential increases in computing power would then come with exponential increases in cost, such improvements are likely to be economically unappetizing, and thus we would expect the rate of progress in many areas to diminish as increases in computing power slow.

2 Theory

Production functions, of the type pioneered by Romer [31], take the following form:

$$Y = AL^{\alpha}K^{\beta} \tag{1}$$

Where Y represents output, K represents capital, L labor, and A represents a productivity multiplier that is calculated as a residual. $0 < \alpha < 1$ and $0 < \beta < 1$ parameterize the marginal contribution as each input grows. As Syverson [32] has pointed out, this form has the advantage of being a linear approximation to any production function. Two key features of such production functions are that they have decreasing marginal returns in each input, and that they are complementary between inputs. Thus, adding ever more capital becomes less and less efficient, but growing capital and labor proportionally yields mutually reinforcing benefits. As I.T. capital becomes increasingly important and evidence mounts that it is complementary, rather than substitutive, to other forms of capital [3], it makes sense to add it separately as another input factor, to get:

$$Y = AL^{\alpha}K^{\beta}IT^{\gamma} \tag{2}$$

For the cases that we will be observing, the rate of change of these inputs will be dramatically different. In particular $\frac{\partial IT}{\partial t}, \frac{\partial A}{\partial t} \gg \frac{\partial L}{\partial t}, \frac{\partial K}{\partial t}$, that is, the rate of increase of I.T. capital and of productivity are much higher than those for labor and non-I.T. capital. This is because there has been large exponential growth in the provision of computing power [33, 29, 30] and in the improvement of using that computing power more efficiently through algorithms [34, 35, 36] — this latter term being much faster in specific computing applications than it is for the economy overall. At the same time, there has been relatively little change in labor and non-IT capital[3].

These growth rates have two important implications. First, exponential growth in an input can be sufficient to overcome decreasing marginal returns and thus provide a constant (or rising) share of the improvements to output. This can be seen by re-parameterizing IT to show exponential growth over time (i.e. $IT_t = IT_0 \cdot e^{vt}$) as compared to some initial period t=0. Substituting this in yields: $Y_t = A_t L_t^{\alpha} K_t^{\beta} (IT_0 \cdot e^{vt})^{\gamma}$, where e^v is the rate of exponential increase. As this makes clear, if $e^{v \cdot \gamma} > 1$ then the decreasing returns to scale are more than overcome by the pace of computing power growth.

The second implication of faster rates of growth for I.T. capital and productivity is that other rate terms are likely to be negligible, i.e.

$$\frac{\partial \mathbf{Y}}{\partial \mathbf{t}} = \frac{\partial \mathbf{Y}}{\partial \mathbf{A}} \cdot \frac{\partial \mathbf{A}}{\partial \mathbf{t}} + \frac{\partial \mathbf{Y}}{\partial \mathbf{IT}} \cdot \frac{\partial \mathbf{IT}}{\partial \mathbf{t}} + \frac{\partial \mathbf{Y}}{\partial \mathbf{L}} \cdot \frac{\partial \mathbf{L}}{\partial \mathbf{t}} + \frac{\partial \mathbf{Y}}{\partial \mathbf{K}} \cdot \frac{\partial \mathbf{K}}{\partial \mathbf{t}} \approx \frac{\partial \mathbf{Y}}{\partial \mathbf{A}} \cdot \frac{\partial \mathbf{A}}{\partial \mathbf{t}} + \frac{\partial \mathbf{Y}}{\partial \mathbf{IT}} \cdot \frac{\partial \mathbf{IT}}{\partial \mathbf{t}} \tag{3}$$

In the case studies that follow, we estimate how performance (Y) and computing power (IT) have changed over time for these important areas of computing. We find that, as expected, IT has strongly decreasing marginal effects on output,

[3] As an example, the number of employees at the National Oceanographic and Atmospheric Association (NOAA) has decreased from nearly 13,000 in 1997 [37] to 11,000 in 2015 [38]

but that $\frac{\partial \mathbf{IT}}{\partial \mathbf{t}}$ has grown so rapidly that $\frac{\partial \mathbf{Y}}{\partial \mathbf{IT}} \cdot \frac{\partial \mathbf{IT}}{\partial \mathbf{t}}$ accounts for most of $\frac{\partial \mathbf{Y}}{\partial \mathbf{t}}$. That is, growth in computing power explains most of the growth in output. In the analysis that follows, we estimate these parameters explicitly, test the robustness of these effects, and consider what these mean for the economic viability of improving outcomes using I.T..

An alternate way of modeling the contribution of I.T. would be via the ideas production function [1, 39, 40], which models the contribution to knowledge accumulation and productivity improvements, rather than total output. In some ways, this might seem a more natural fit, since we are measuring the skill of systems, which might be imagined as a form of the accumulation of knowledge. However, ideas production functions implicitly need to model not just performance, but performance per unit of input, and that is not how we measure our outcomes. Consequently, we use the traditional output-based form of the prodution function for our modeling.

3 Case Studies

In what follows, we present case studies in five domains about how computing power has been associated with improved performance. In each case, we present the history, summary trends, and correlational evidence. In discussing these, we will use the language of causality, which we will defend in section 4 when we analyze these areas collectively.

3.1 Computer Chess

Chess is one of the most popular board games in the world and has been played since at least the 6th century A.D. [41]. The first discussion of chess, from a computational perspective, came in 1950 when the mathematician Claude Shannon published a paper entitled "Programming a Computer for Playing Chess" [42]. Shortly thereafter, Alan Turing created the first algorithm that a computer could use to play chess. Unfortunately, because of the state of computing at the time, Turing's chess algorithm had to be executed manually. It took 15-30 minutes to calculate each move and the algorithm only considered the consequences of its actions two moves in advance. It could not play a full game, much less beat a professional chess player. According to Kasparov and Friedel, Turing's algorithm would take less than five milliseconds to calculate each move in a modern computer in 2017 [43].

Roughly speaking, all computer chess programs do two types of operations: (1) look ahead to potential future board states and (2) evaluate how likely any board position is to produce a win. For example, a program might look ahead 3 moves (called "plies") and for each potential result it will evaluate whether it is in a good or bad position by counting when opponent's pieces can be constrained or taken. In computer science, the resulting analysis is thought of as a tree, where nodes are the board position and edges are the possible moves for each player, as shown in Figure 1 where in d) the computer (white player) checkmates in 3 plies.

Each node has an associated probability of winning. In general, these guide the system towards good moves, but can be misleading if the computer can't look far enough into the future (and see a potential problem spot) or if it incorrectly evaluates how valuable different board configurations are. Additional computing ameliorates these deficiencies by looking further ahead in the tree or doing better evaluation of a board configuration. Looking farther ahead, however, requires exponentially more computing power since the number of possible moves grows exponentially as you project forward. For example, if each player had 10 potential moves each ply, then there are 10,000 potential evaluations after four plies (10^4), whereas looking six plies into the future would take 1,000,000 evaluations (10^6). Smarter evaluation algorithms help this by decreasing the number of moves considered at each point, but at the cost of doing more computation to evaluate each board position. Most programs find a balance between spending time exploring farther ahead and evaluating positions more carefully - although there are exceptions that heavily favor one approach or another[4].

By 1957, Alex Bernstein was able to develop a chess program running on an IBM 704 mainframe that was capable of playing a full game [44]. Ten years later, the MacHack IV program was the first to play in a tournament. It ended up with a score of one draw and four losses against amateur J.Conroy [45]. 30 years later, in 1996, after enormous work by the computer chess community Deep Blue, an IBM RS/6000 SP supercomputer capable of calculating 200 million chess positions per second[5], beat the world chess champion, Garry Kasparov.

Algorithm and hardware improvements have continued to progress since Deep Blue. One recent study showed that a modern computer chess algorithm, running on a single 1994-era 486-DX4 100 MHz machine, would have achieved the Kasparov-levels of performance — several years ahead of when Deep Blue achieved this with massively more computing power [47]. On the other hand, old chess programs could also achieve much higher performance running on modern hardware [48]. As of July 2021, the best chess program was Stockfish 13. With an Elo score of 3547, Stockfish

[4] One such example is AlphaZero, a 2017 program by DeepMind that focuses more on evaluating positions than depth of search.

[5] By way of comparison, NASA's Pathfinder used the same IBM RS/6000 technology for its onboard flight to Mars [46]

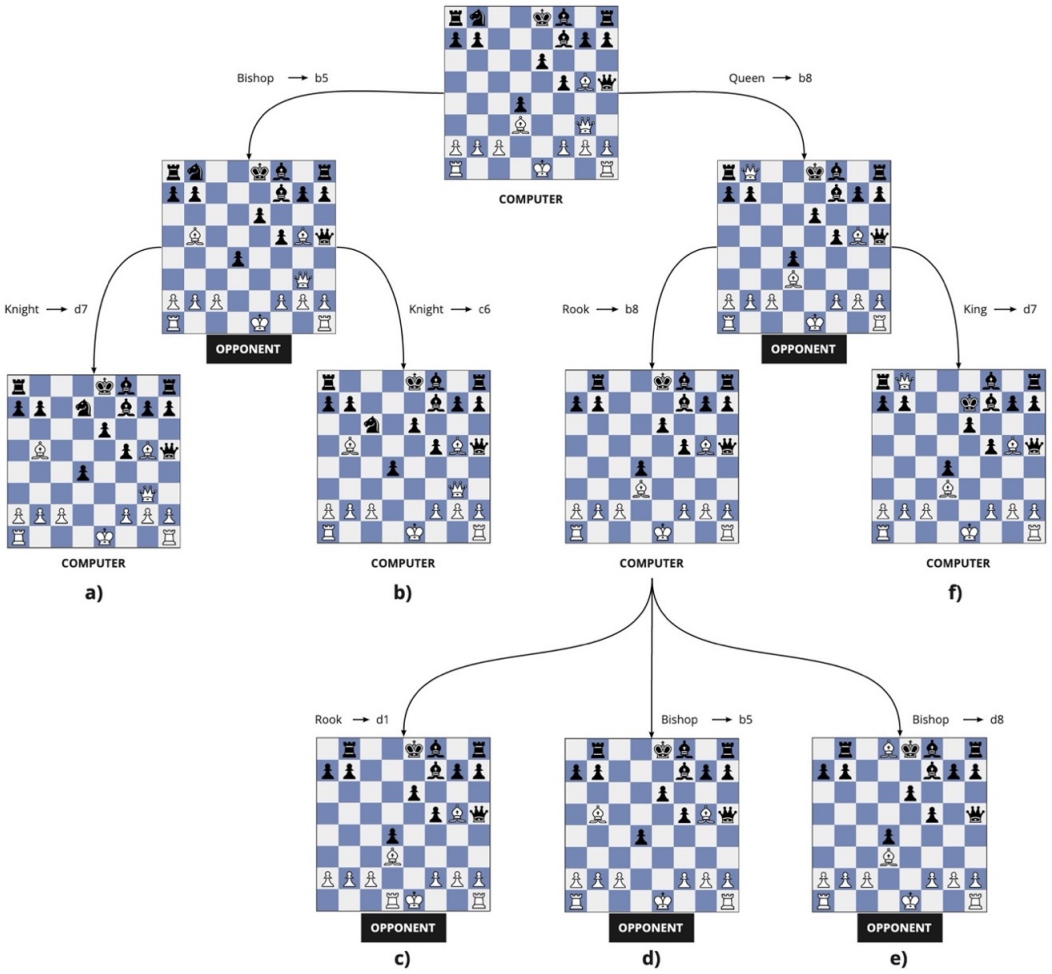

Figure 1: Small subset of a decision tree for chess.

14 is 665 points better than the best rating ever achieved by a chess human player, Magnus Carlsen at 2882 in May 2014 [49]. This means that the best players in the world are as likely to win a game against Stockfish 13 as a top amateur player would be to win a game against a grandmaster.

3.1.1 Computing Power in Computer Chess

To assess the progress of chess computers over time, we developed an extensive history of computer chess programs from 1957 (Bernstein's program) to 2019 (Komodo 13.1 at the World Computer Chess Championship). For data from 1957 to 2006 we use matches between computers and humans, as gathered from records from international chess associations, research papers, books, databases provided by the community, and others (see Appendix 7). From 2006 on we use data from the World Computer Chess Championship, where computers face each other, since at that point computer performance is super-human. Figure 2a shows how the performance of computer chess has evolved.

Since Bernstein's 1957 model, computer chess performance went from a novice Elo score to a super-human Elo score of 3547 and increased their computing power usage by a factor of 10^8. On average, chess programs improved by 37.6 Elo points per year and increased this computing power used by 38% per year (=$10^{0.14}$).

Analyzing the performance and the number of chess positions shows that an increase of 10× in computing power is associated with an increase in Elo rating of 242 points (statistically significant at the 0.01 level as shown in Table 1), as shown in Figure 2c. This is similar to the estimate made by the Deep Blue team when planning the hardware needed to beat Kasparov "there is a 200-point ELO rating improvement for each order of magnitude improvement in computing speed of the chess machine platform" [50]. The variation in computing power explains 88% of the variation

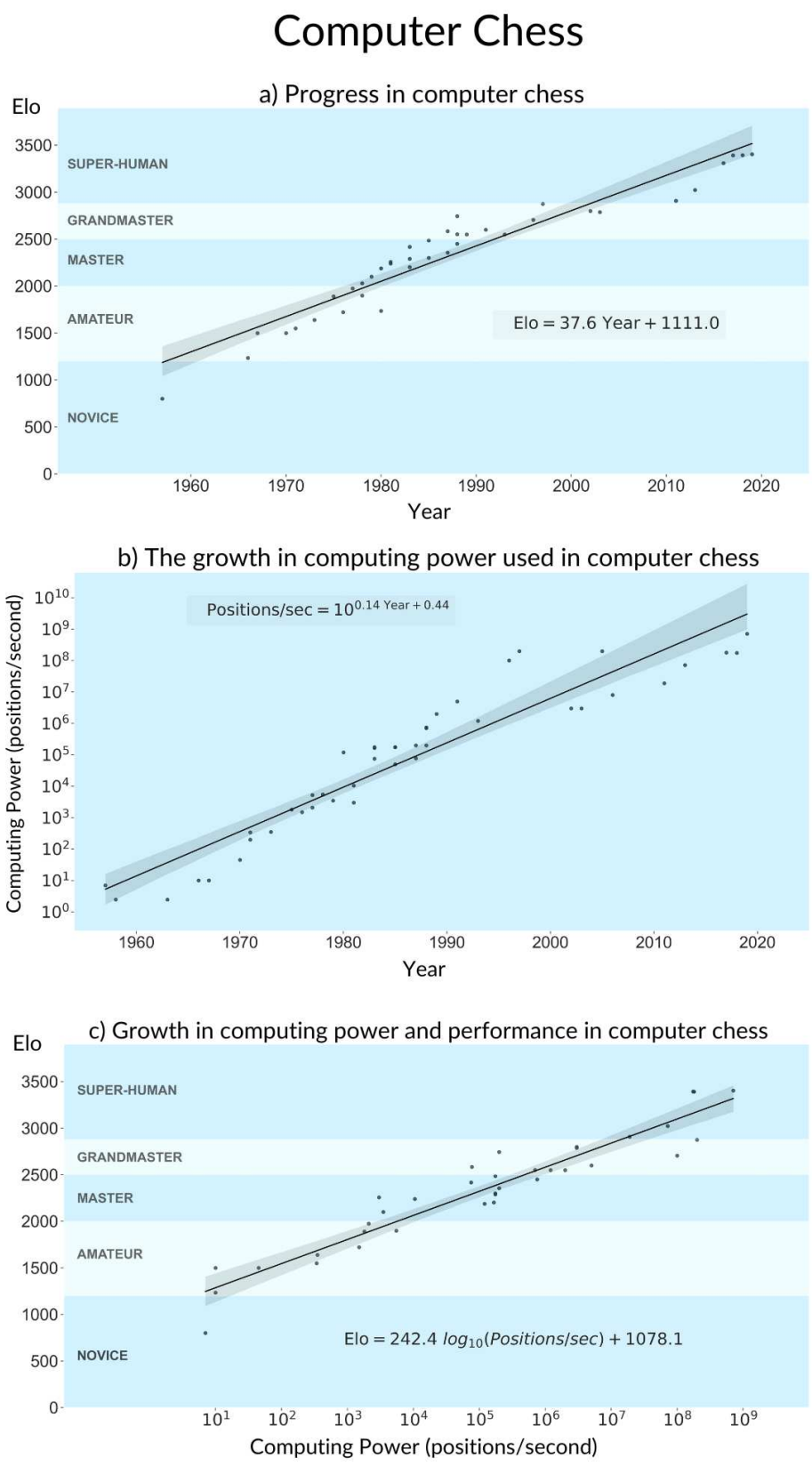

Figure 2: Computer Chess: (a) Elo scores over time, (b) computing power used over time, and (c) Elo scores as computing power increases. In each subfigure, the dots are individual programs, the lines are linear regressions, and the shaded region is the 95% confidence interval for the regression.

in performance, while the residual variation (e.g. due to algorithmic improvement independent of computing power) only explains 12%.

3.2 Computer Go

Go is the oldest board game played in the world, having been invented in China approximately four thousand years ago [51]. The basic rules of Go are simple: players take turns placing their stones on a 19×19 board and get points by completely surrounding the other player's stones. This simplicity, however, masks an enormous amount of computational complexity due to the large number of possible moves that players can make each turn. To put this into perspective, after each player makes a single move in chess, there can be 400 possible board configurations, in Go, there can be more than 130,000. Another way of understanding this difference is by comparing the hardware that was used when programs beat the best humans. When Deep Blue beat Garry Kasparov at chess, it used 30 chips containing 480 specialized processors. Twenty years later, when processor chips where 100× better [52], AlphaGo beat Lee Sedol using, roughly, $75\times$ as many processors[6]. That is, using $\sim 7500\times$+ more computing power.

As with chess, the computational intensity of Go comes from the challenges of looking farther ahead in the game and from evaluating board position. Because of the much larger number of possible moves, the exponential explosion of looking ahead in the game happens much more quickly.

Computer Go began with Albert Zobrist's 1970 dissertation. Programs – such as Interim.2, The Many Faces of Go and Go++ – emerged shortly thereafter. In 2006, an algorithmic improvement[7] significantly improved performance [55, 45], allowing a Go computer to achieve a rank of an advanced amateur (1 dan) on a smaller (9×9) board [56]. In 2015, AlphaGo defeated the European Go champion, Fan Hui, in a five-round game by 5-0 [57]. This was the first time that an AI system had beaten a human professional player without a handicap. One year later AlphaGo sealed a 4-1 victory over Lee Sedol [58], considered by many to be the best Go player of all time [59].

3.2.1 Computing Power in Computer Go

To assess how Go programs use computing power and how it is impacting their performance we gathered data on computer versus human games. We sourced this data from the British Go Association, the European Go Federation, research papers, tournament and personal reports, books, databases provided by the community, and others (see Appendix 7). A key source in this work is the database assemble by Nick Wedd [60]. To fill in missing data, we directly contacted Go developers, teams, professional players and other members of the Go community.

As Figure 3a shows, there has been enormous improvement in computer Go from the 1970s until today, with an average of 84 Elo points being added to performance per year [8]. The best performing systems are now much better than humans, with AlphaGo Zero achieving an Elo of 5135, compared to the human champion Shin Jinseo, who is a 3800 Elo player.

Unfortunately, very little data is available on the computing power used by the early Go systems, so we focus our analysis on the period since 1990, when better data is available. Since that time, the amount of computing power (as measured by floating point operations per second) used by Go programs has increased approximately one hundred billion-fold, a doubling every year (=$10^{0.3}$), as shown in Figure 3b.

Graph (c) compares the growth of computing power in Go with the performance of those programs, revealing a highly significant correlation between them (statistically significant at the p=0.01 level as shown in Table 1). In Go, a 10× increase in computing power increases Elo by 246 points on average. In Go, the variation in computing power explains 49% of the variation in system performance.

3.3 Weather Forecasting

It is estimated that 96% of the U.S. population relies on the weather forecasts provided by the National Oceanic and Atmospheric Association (NOAA), either directly (e.g. through their website) or indirectly via third-parties that process the data that they provide (e.g. the Weather Channel on TV or weather apps like AccuWeather on mobile devices)[25, 61, 62]. The economic importance of forecasts is also substantial. Better weather forecasts can guide

[6]The exact number of processors used by AlphaGo is unknown. Here we make the simplifying assumption that each modern chip has a similar number of processors as the IBM machines (16 processors/chip) and thus AlphaGo's 1920 CPUs, 280 GPUs, and 48 TPUs [53, 54] represent at least $\sim$ 36,000 "processors-equivalents". In actual fact, the number may be much larger since, for example, a single TPU has $\sim$ 65,000 multiply units, which would make our number an underestimate.

[7]The introduction of Monte-Carlo tree search (MCTS).

[8]To measure the Elo ratings of Go programs, we use the ranking of the opponent and the number of handicap stones given to infer the rank of the players in kyu/dan, and then use the European Go Federation's conversion from ranks into Elo.

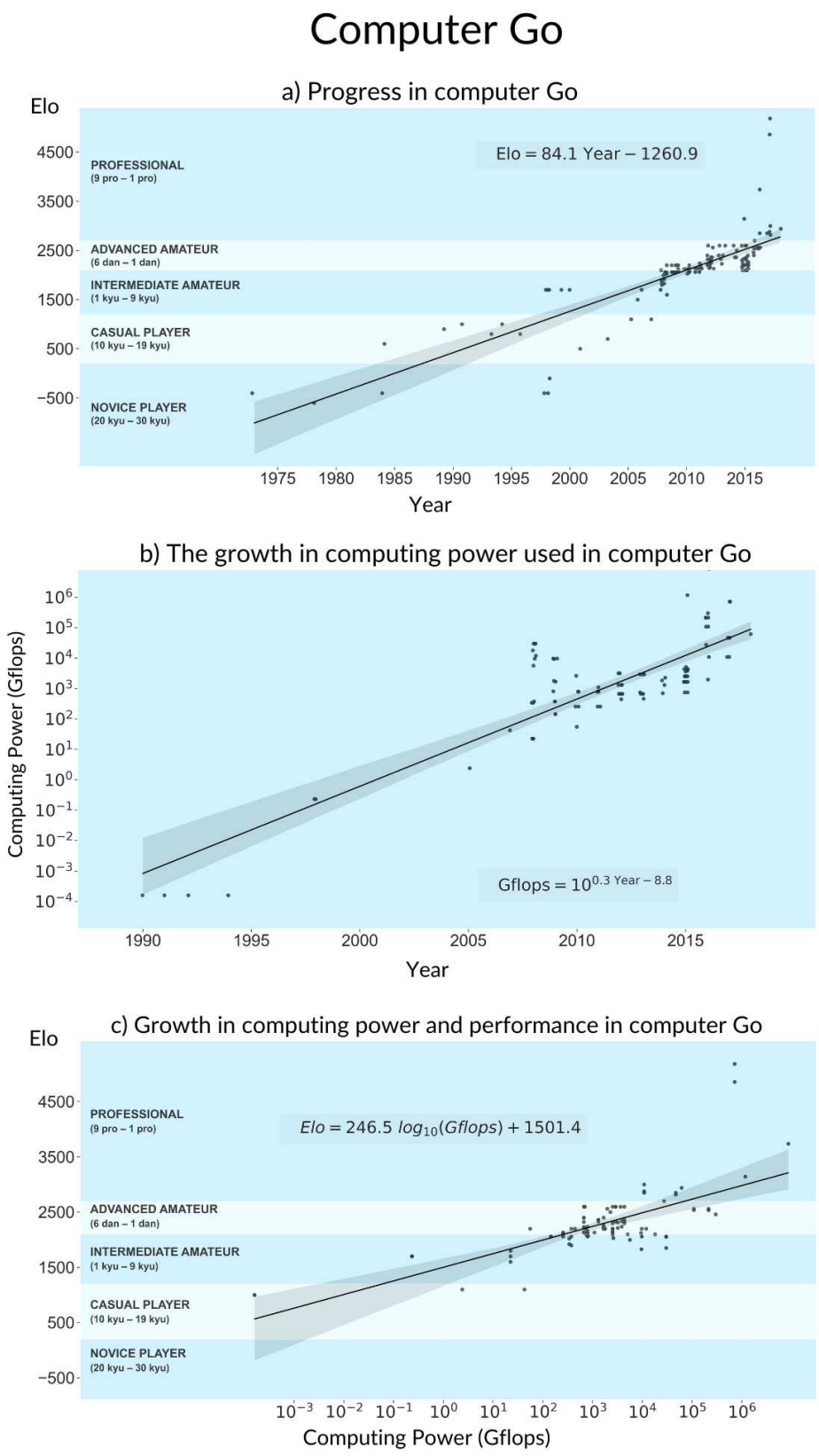

Figure 3: Computer Go: (a) Elo scores over time, (b) computing power used over time, and (c) Elo scores as computing power increases. In each subfigure, the dots are individual programs, the lines are linear regressions and the shaded region represents the 95% confidence interval for the regression.

farmers in when to gather their crops before the first frost, help energy utilities know which generators to have at the ready to heat/cool peoples' homes, and can advise people to take shelter if a hurricane is headed their way. These, and the many other ways that forecasts guide people in the economy, are estimated to be worth approximately $31.5 billion per year [25].

Before the mid-nineteenth century, weather forecasting was done based on weather lore, personal observations and simple measurements (e.g. of humidity) [63]. Once the telegraph was invented, observers in different locations communicated to produce the earliest weather maps. Today, there is a vast web of sensors that perform these tasks, including over 10,000 manned and automatic surface weather stations, 1,000 upper-air stations (e.g. weather balloons), 7,000 ships, 100 moored and 100 drifting buoys, hundreds of weather radars and 3,000 specially equipped commercial aircraft, together with 30 meteorological and 200 research satellites [64]. These observations are sent to supercomputers that estimate enormous systems of equations to produce predictions. Flynn [65] estimates that even the earliest numerical weather forecast models would require 204,800 people to solve manually.

The computational burden of numerical weather prediction can largely be explained by three factors: resolution, dynamics and variations. The resolution of the model refers to the geographic space modeled as a single unit. For example, the current U.S. model divides the atmosphere into chunks that are 13km long $\times$ 13km wide [66]. The resolution of models has changed dramatically over time. For example, the global leader in weather forecasting models is the European Centre for Medium-Range Weather Forecasts (ECMWF) [67], which has increased in resolution from 210 kilometers in 1979 down to a 9 kilometers in 2016 [68, 69]. Improvements in these three dimensions of resolution are computationally expensive, with each doubling requiring an (2^3) 8-fold rise in the amount of computation [70]. A common goal in the weather prediction community is to get models to 1 km resolution with 200 vertical levels and 100 variables for the dynamics, which would help accurately model mountain ranges and other important features but would require 2,000 times as much computing power [71].

The dynamics of weather prediction refer to how different units of space (of whatever resolution) interact with each other and how this is modeled. Many different techniques are used for this, for example using partial differential equations of physical interactions to model variables like temperature, wind, pressure, and other hydrodynamics and thermodynamics variables [72]. The models take the initial state of the model and predict what will happen in 24, 48, 72, or more hours. Because small variations in the initial conditions can produce large differences in outcomes, weather prediction often uses ensemble methods, running the model many times with slightly perturbed initial conditions in order to explore the variation in outcomes that this produces. This can easily increase the amount of computing needed for a calculation by a factor of 50 to 100 [71].

3.3.1 Computing Power in Weather Forecasting

To assess progress in temperature prediction, we analyze data provided by NOAA on the performance of their high-performance computing systems since the 1950s. We measure performance with the mean absolute error (in Fahrenheit degrees) between the temperature prediction and the actual temperature [9]. As shown in Figure 4(a), the errors made by NOAA in predicting temperatures have fallen dramatically since the 1970s. For example, predictions for weather 3 days in the future have dropped from an error of 5.8 degrees Fahrenheit in 1972 to 3.0 degrees in 2017 a drop of 47%.

The computing power used by NOAA for weather prediction has also escalated dramatically, growing nearly a trillion-fold from 1956 until 2017, as shown in Figure 4(b), representing an increase of 48.2% per year.[10] The large jumps in this figure represent new supercomputers coming online, whereas the smaller jumps come from expansions to existing systems.

Combining the data from Figure 4(a) and Figure 4(b) allows us to examine the progress in prediction as computing power has varied. Figure 4(c) shows the resultant tight correlation. In weather prediction, increasing computing power by 10× yields a decrease in prediction error of 1/3 of a degree Fahrenheit (statistically significant at a p-value < 0.01 (see Table 1). Perhaps even more notable, variation in the amount of computing power used explains 73-94% of the variation in performance, suggesting that improvements in computing power (and in the algorithms needed to harness it) are responsible for the vast majority improvements in weather prediction performance.

[9] Consistent with the norms in this field, only the error in the prediction of maximum and minimum temperature is shown, but this result holds when we use other temperature indicators such as average temperature. Trends in hurricaine prediction, another area that NOAA predicts, show similar progress

[10] Note: unlike for Chess and Go, this measurement is about the functionality of the whole system, not the computation needed for a particular task. That said, discussions with practitioners indicate that the time window used for calculations have been quite stable due to reporting requirements (e.g. updating forecasts hourly) and thus the two should be highly correlated.

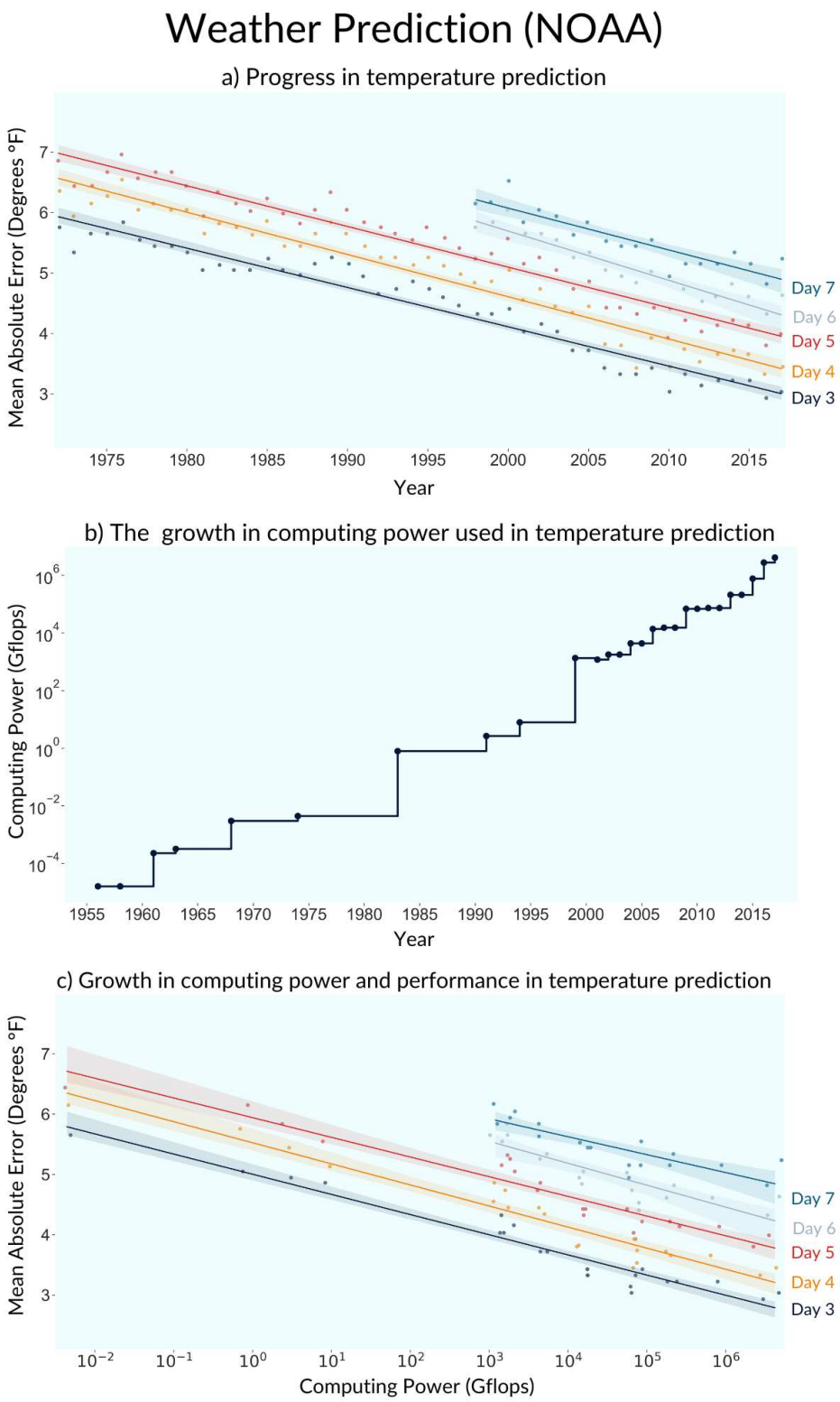

Figure 4: Temperature Prediction: (a) Forecast accuracy over time, (b) Computing power over time, as measured by system capacity, and (c) forecast accuracy as computing power increases. Lines with shaded regions are linear regressions with 95% confidence intervals [73]

.

3.4 Protein Folding

Proteins are the molecular machines of the body, performing virtually all the important biochemical processes [74]. When first produced in the body, proteins consist of a linear chain of units[11] which then fold into a three-dimensional structure that interacts with the world. Biologists care a great deal about protein folding because the folding pattern determines a protein's structure, what it binds to, its stability, and other properties that are important for health and medicine [75]. An early discussion of predicting protein folding highlighted the incredible difficulty of these calculations: a moderate-sized protein might have 10^{300} possible configurations [76].

Assessments of protein folding skill happen at events such as the Critical Assessment of Protein Structure Prediction Experiments (CASP), where hundreds of protein sequences are presented to the computational modelling community who attempt to predict their folded structure. Success is measured with a GDT_TS (Global Distance Test_Total Score) which calculates the similarity between the results of protein structure prediction and the experimentally determined structure spotted by NMR or X-RAY crystallography. The score ranges from 0 when the two structures are completely different to 100 when the predicted structure is exactly the actual structure. In the analysis that follows we focus on GDT_TS scores for free modelings problems, where the full folding problem must be solved (there are also simpler contests where similar sequences can be used as templates, vastly simplifying the difficulty of the problem).

3.4.1 Computing Power in Protein Folding

Tracking the effects of computing power is significantly harder in protein folding than other domains. Because, while it is easy to source information from CASP on how good structural predictions are [77], virtually none of the related papers on these models report their computing power usage[12]. After carefully reviewing all the publications from CASP 7 to CASP 13, around 200 papers in total, we are only able to extract computation data for 5, including AlphaFold1[13] and AlphaFold2. Fortunately, these models span a 100,000 fold difference in computing power so we are still able to see significant variation, as shown in Figure 5.

Overall, we observe rapid progress. In the six years from 2015 to 2021 performance has risen from 65 to 69, or 92 . This has been accompanied by a huge increase in computing power being used, from 421M Gflops to 12.2 trillion Gflops, a compound rate of increase of $4\times$ per year. We also find that computing power changes explain 87% of the variation, with each 10x increase in computing power being associated with an 6.7 increase GDT_TS performance.

Since this analysis is based on only a small number of data points, we also conduct an alternative empirical analysis in Appendix 8. That also finds strong support for role of computing power in performance increases.

3.5 Oil Exploration

Drilling for oil is expensive, costing $3.3 trillion worldwide yearly [78] and representing 8.5% of the capital expenditures for a major oil company such as Chevron (Chevron, 2019). Drilling an on-shore well is estimated to cost between $4.9 M to $8.3 M on average [26], and off-shore wells are estimated to cost $650 M on average [79]. With such high costs, there is great value in assuring that drilling produces active wells and not 'dry holes'. Computational methods can be of great value in guiding drilling. The probability of drilling a successful well is 70% with 3D seismic methods, while 30%-35% without 3D [80].

Seismic modelling emerged in the 1950s. It models the physical structure of the earth by understanding how seismic waves (e.g. concussions done by the mappers themselves) travel and reflect off of different materials. Computationally, this involves solving a series of wave equations that contain important physical parameters of the geologic materials. Unfortunately, even in the late 1970s, computing power was insufficient for solving the comprehensive set of wave equations in more complex seismic models [81]. Instead, data was gathered in two dimensions, with repeated sampling needed to form a set of slices spanning a volume of the subsurface.

Understanding improved when modeling was done in 3D, yielding better resolution and structural information [82]. However, adding dimensions also means adding parameters, which is computationally expensive [83]. As the earliest literature that described this method said, "because of the large number of unknowns…the calculation…by finite-differences using small increments of each of the variables…would lead to many hundreds of evaluations…hence to many hundreds of solutions of the wave equation…This would be computationally prohibitive." [81], Fortunately, as

[11] Amino acid residues.

[12] Either directly or via a measure of computing hardware burden [21], calculated as # processors × Computation Rate × time.

[13] The computing power for Alphafold1 is estimated as 4 times the computation used in ProSPr (Billings, 2019). This estimation might miss the massive training size and potential ensembles embedded in the original work of Alphafold1.

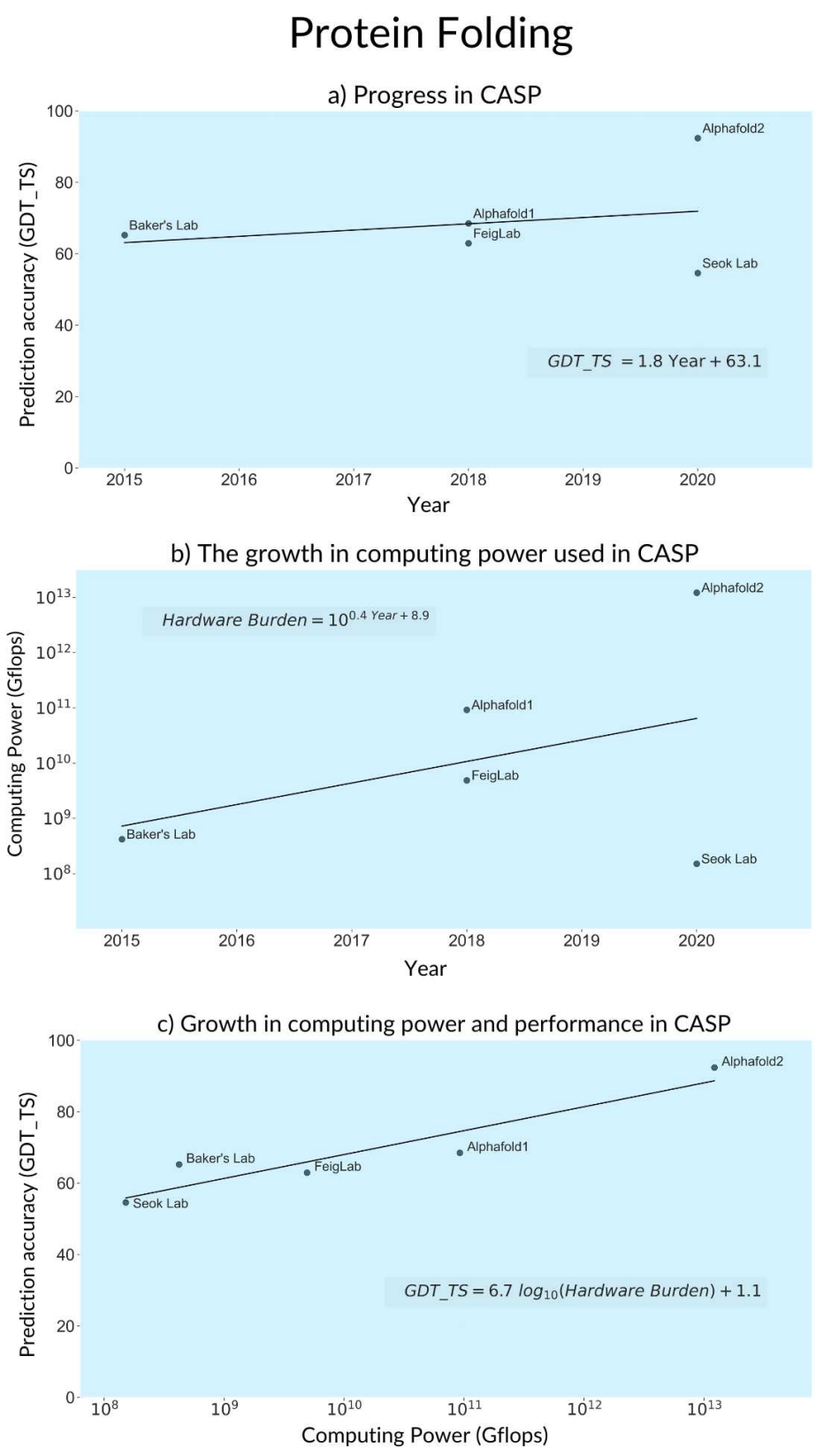

Figure 5: Protein Folding: (a) Prediction accuracy (as measured by GDT_TS scores) over time, (b) computing power used over time, and (c) Prediction accuracy (GDT_TS scores) as computing power increases. All data from CASP competitions. Lines are linear regressions, shaded areas are their 95% confidence intervals.

the cost of computing power fell and available computing power grew, 3D seismic modeling became possible by 1982. By 1991, this was further extended to include time-lapse (so-called "4D modeling").

3.5.1 Computing Power in Oil Exploration

According to the Energy Information Administration, the rate of drilling success (i.e. not getting dry wells) in the United States improved from 10% in the 1940s to 70% in the 2010s [84]. To understand how drilling performance has improved, we considered the computing power used by the largest oil companies around the world: Total, ExxonMobil, Chevron, and BP. Reflecting the high value of seismic modeling, each of these companies has a supercomputer more powerful than the supercomputer that NOAA uses for weather prediction for the whole country. And, in fact, another petroleum company, Eni, has the world's most powerful supercomputer in commercial use - it has 3,200 Nvidia Tesla GPUs and 18.6 petaflops operational capacity [85]. Companies also greatly increased the power of their computers over time: Exxon Mobil from $\sim$ 79 Gflops in 1998 to 26,000,000 Gflops in 2019 (330K-fold increase), Chevron from 75 Gflops in 1991 to 2,000,000 Gflops in 2012 (a 27K-fold increase), and BP from $\sim$ 100 Gflops in 1999 to 20,000,000 Gflops in 2020 (a 200K-fold increase).

Using data from iHS Markit energy portal U.S. Data Online, a comprehensive well archive with reports on over 4 million wells and 2.7 million producing entities, we analyzed the success that BP had in exploratory drilling for offshore wells [86]. We focus on off-shore wells because these are the ones where seismic is the main investigation method, and of those we focus on wildcat wells which are exploratory (rather than drilling into a known area). Figure 6 presents the drilling success rates for these wells over time, after removing the time trend because oil drilling gets progressively more difficult as easier areas are exhausted.

These data show that a 10× increase in computing power is been associated with over 43 percentage point improvement in the drilling success rate for BP (significant at a p-value of 1%). Importantly, this improvement happens against a backdrop of drilling getting harder as easier locations have already been drilled — that is, over time the drilling success rate with a given amount of computing power is falling. After de-trending these rates, computing power explains 85% of the variation in the drilling success rate, affirming the importance of computing power in this area.

There are important caveats to this finding. First, our intention had been to analyze four companies in our analysis: BP, Chevron, ExxonMobil and Total. We excluded ExxonMobil and Total for having too few data points, and Chevron because we did not trust the data quality after observing a giant, unexplained drop in its drilling success in 1992, as well as frequent year-to-year jumps between a 0 and 100% success rate which seemed implausible. Second, because we are looking at the de-trended data, we are only explaining the residual variance after that detrending, not all the variance.

3.6 Summary of dependence on computing power

Looking across each of these five domains, we find (Table 1) the following relationships between computing power and additional performance:

4 Analysis and Discussion

Having presented our case studies graphically, we now explicitly estimate the four key parameters from our theory model, in particular: $\psi = \frac{\partial IT}{\partial t}$ the rate of increase of computing power, γ the exponent for I.T., $\frac{\partial Y}{\partial IT} \cdot \frac{\partial IT}{\partial t}$ the proportion of improvement in performance explainable from improvement in I.T., and $\rho \equiv \frac{\partial Y}{\partial t} - \frac{\partial Y}{\partial IT} \cdot \frac{\partial IT}{\partial t}$ the residual portion of growth not explained by increases in computing power.

Throughout this section, we shall use the language of causality in interpreting our results. Normally with time series data, this would be problematic without a natural experiment, instrumental variable or other means of providing statistical identification. But here, we rely not on statistical analysis of our data to get causality, but on the extensive scientific experimentation done by those in these fields. That is, in all five of these areas, engineering and scientific understanding has been built using experiments that prove that computing power causally improves performance (e.g. [87, 70]). For example, NOAA does tests showing that weather forecasts can be improved by running a for a longer time than it would otherwise for a calculation. For example, to test whether a $2\times$ increase in computing would give a better result, they could run a calculation that needs to be done in 1 hour for 2 hours. The results from this testing are then part of the funding approval process for getting sufficient computing power to run the new method in the required amount of time. Thus, it is the experimental testing in these fields, not after-the-fact statistical identification, that we use to get causality[14].

[14] One potential caveat to this is chess in the period since 1997. Since that time, the cost of computing being used has fallen, suggesting that there may have been less vigilance on the budget and thus on proving additional performance.

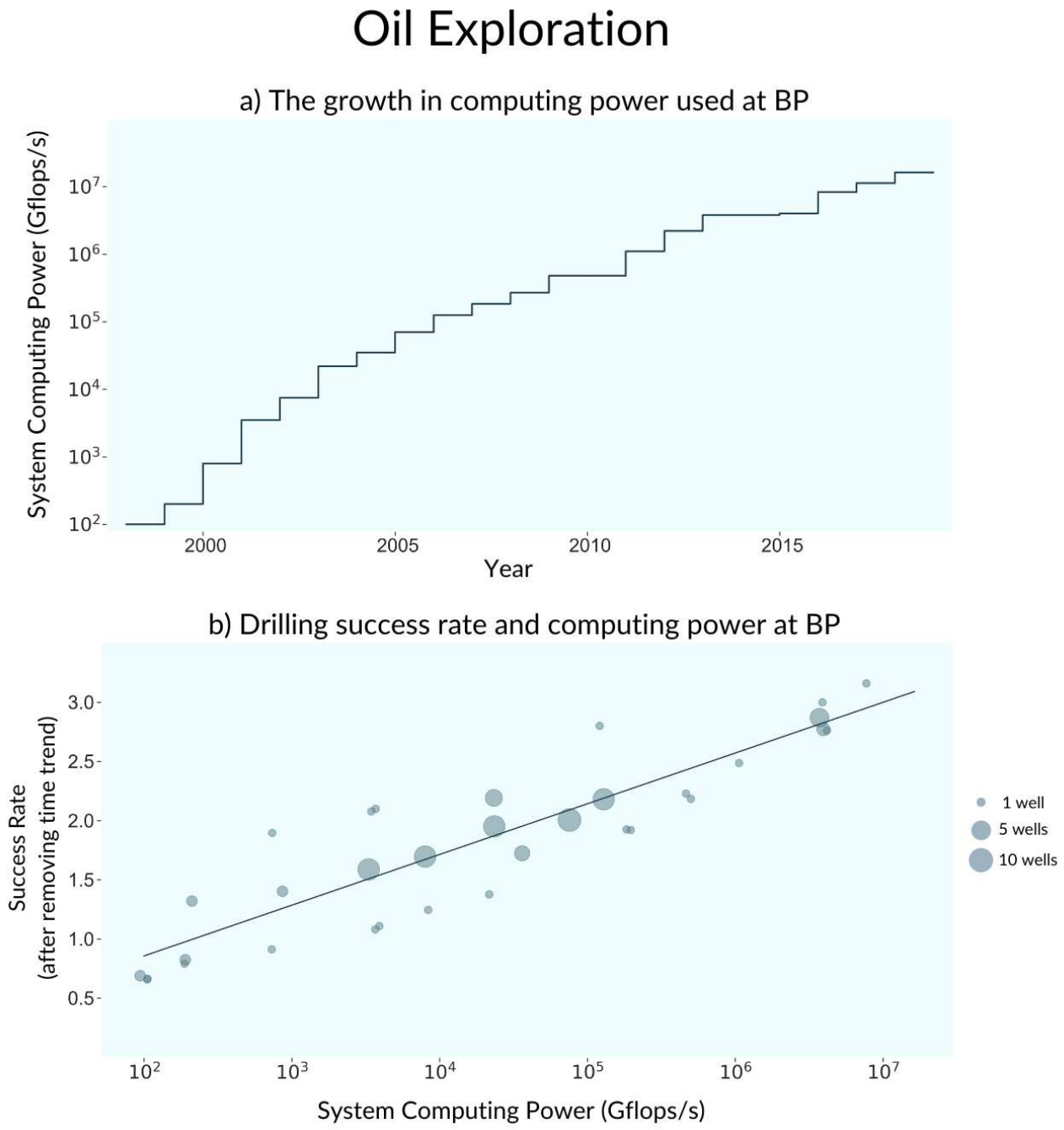

Figure 6: Oil Exploration: a) The growth in computing power used for oil exploration at BP. b) Increase in the drilling success rate at BP as computing power used increases. The success rate shown is after excluding the overall time trend (which is why it can exceed 1). Each circle represents the number of wells in a specific year at a specific depth interval.

Table 1: Performance over computation in logarithm scale.

	Computer Chess	Computer Go	Oil Exploration (BP)	Protein Folding (CASP)
	ELO	ELO	Drilling Success %	GDT_TS
	(1)	(2)	(3)	(4)
$Constant$	1,078***	1,501***	0.00002	1.11
	(82)	(87)	(0.15)	(15.05)
$log_{10}(Computing\ Power)$	242***	246***	0.43***	6.69**
	(16)	(24)	(0.03)	(1.47)
Observations	31	104	33	5
R^2	0.88	0.49	0.85	0.87
Adjusted R^2	0.88	0.49	0.85	0.83
Residual Std. Error	180.33 df=29	395.54 df=102	0.37 df=31	5.81 df=3
F Statistic	218*** df=1; 29	218*** df=1; 102	182*** df=1; 31	20** df=1; 3

	Weather Prediction				
	Day 3	Day 4	Day 5	Day 6	Day 7
	(4)	(5)	(6)	(7)	(8)
$Constant$	5.00***	5.53***	5.94***	6.64***	6.81***
	(0.08)	(0.11)	(0.09)	(0.24)	(0.21)
$log_{10}(Computing\ Power)$	-0.33***	-0.35***	-0.33***	-0.36***	-0.30***
	(0.02)	(0.02)	(0.02)	(0.05)	(0.04)
Observations	22	22	22	18	18
R^2	0.94	0.90	0.92	0.76	0.73
Adjusted R^2	0.94	0.90	0.91	0.74	0.71
Residual Std. Error	0.19 df=20	0.26 df=20	0.22 df=20	0.23 df=16	0.21 df=16
F Statistic	309*** df=1; 20	188*** df=1; 20	220*** df=1; 20	50*** df=1; 16	43*** df=1; 16

Note: *p<0.1; **p<0.05; ***p<0.01

The economic cost of the computing systems used provides a second argument for why the causality runs from computing power to performance: revealed preference. Even after accounting for rapid hardware improvement rates, all of these areas have shown enormous increases in the cost of the computing power being used. Literally, hundreds of millions of dollars have been spent on these systems because their owners were convinced that causality runs in this direction. Had this not been true, we would have expected the owners of these systems to update their computers to get the benefit of newer hardware while maintaining or lowering costs.

While we find the experiments conducted by scientists to be particular compelling evidence of causality (much more so that typical statistical measures), we also recognize that this way of establishing it is unorthodox in this literature. As such, in Appendix 6, we also present more-traditional statistical tests for establishing causality.

4.1 The Growth in Computing Power

Table 2 shows our estimates for ψ the rate of increase of computing power across our domains:

Table 2: Growth in computing power per year.

	Computer Chess	Computer Go	Weather Prediction	Oil Exploration (BP)	Protein Folding (CASP)
	Positions/sec (log_{10}) (1)	Gigaflops (log_{10}) (2)	Gigaflops (log_{10}) (3)	Gigaflops (log_{10}) (4)	Hardware Burden (log_{10}) (5)
$Constant$	0.44***	-8.81***	-5.22***	2.78***	8.86
	(0.27)	(0.69)	(0.17)	(0.15)	(1.88)
$Year$	0.14***	0.29***	0.18***	0.23***	0.39
	(0.01)	(0.02)	(0.003)	(0.01)	(0.51)
Observations	43	109	27	22	5
R^2	0.88	0.74	0.98	0.94	0.16
Adjusted R^2	0.88	0.74	0.99	0.95	−0.12
Residual Std. Error	0.80 df=41	0.93 df=107	0.39 df=25	0.37 df=20	2.09 df=3
F Statistic	324.97*** df=1; 41	303.32*** df=1; 107	2,205.71*** df=1; 25	366.61*** df=1; 20	0.58 df=1; 3
ψ	1.3803	1.9498	1.5135	1.6982	2.4547
Yearly Computing Power Increase	38.0%	94.9%	51.3%	69.8%	145.4%

Note: *p<0.1; **p<0.05; ***p<0.01

As this shows, for example for Chess, $\frac{\partial log_{10}(ComputingPower)}{\partial Year} = 0.14$ (statistically significant at p-value $< 1\%$). Exponentiating shows that computer power usage in chess ($\psi = \frac{\partial ComputingPower}{\partial Year}$) is 1.38. That is, the amount of computing used for chess has grown by 38% per year, on average. In other areas, this effect is even stronger, with Go increasing at 95%, weather prediction by 51%, oil exploration by 70% for BP, and protein folding by 145%.

4.2 Contributions of I.T. to Performance Improvement

Table 3 shows our estimates for γ, the coefficient that describes how increased computing power changes performance across these domains. Here we explicitly model the production function $Y = AL^{\alpha}K^{\beta}IT^{\gamma}$ by taking logs to get our

estimating equation: $log(Y) = log(A) + \alpha\, log(L) + \beta\, log(K) + \gamma\, log(IT)$ where we collectively estimate the non-IT independent variables as part of our residual.

Table 3: Performance improvement as computation grows.

	Computer Chess	Computer Go	Oil Exploration (BP)	Protein Folding (CASP)
	ELO (log_{10}) (1)	ELO (log_{10}) (2)	Drilling Success (log_{10}) (3)	GDT_ TS (log_{10}) (5)
$Constant$	3.07***	3.2***	-0.24***	1.43*
	(0.02)	(0.01)	(0.05)	(0.09)
$log_{10}(Computing\ Power)$	0.05***	0.05***	0.11***	0.04
	(0.005)	(0.01)	(0.011)	(0.01)
Observations	31	104	33	5
R^2	0.80	0.63	0.78	0.88
Adjusted R^2	0.80	0.63	0.77	0.84
Residual Std. Error	0.06 df=29	0.06 df=102	0.13 df=31	0.03 df=3
F Statistic	119.15*** df=1; 29	174.85*** df=1; 102	107.80*** df=1; 31	21.45 df=1; 3
γ	0.05	0.05	0.11	0.04

	Weather Prediction				
	Day 3 (log_{10}) (3)	Day 4 (log_{10}) (4)	Day 5 (log_{10}) (5)	Day 6 (log_{10}) (6)	Day 7 (log_{10}) (7)
$Constant$	0.7000***	0.7430***	0.7752***	0.8386***	0.8433***
	(0.0099)	(0.0119)	(0.0092)	(0.0203)	(0.0167)
$log_{10}(Computing\ Power)$	-0.0355***	-0.0333***	-0.0283***	-0.0313***	-0.0235***
	(0.0023)	(0.0028)	(0.0022)	(0.0044)	(0.0036)
Observations	22	22	22	18	18
R^2	0.92	0.88	0.89	0.76	0.73
Adjusted R^2	0.92	0.87	0.89	0.75	0.71
Residual Std. Error	0.02 df=20;	0.03 df=20;	0.02 df=20;	0.02 df=16;	0.02 df=16;
F Statistic	231.67*** df=1; 20	141.07*** df=1; 20	169.87*** df=1; 20	51.54*** df=1; 16	43.17*** df=1; 16
γ	0.04	0.03	0.03	0.03	0.02

Note: *p<0.1; **p<0.05; ***p<0.01

These show that computing power has, in general, very low exponent γ, ranging from 0.02 for weather prediction to 0.11 for oil drilling for BP (recall: because weather prediction is measured by error, those values are -γ). This is what we would expect. Adding a unit of computation to today's powerful machines has much less impact on outcomes that did ones when computers were brand new.

By contrast with these estimates for the returns to additional computing power, recent estimates for the impact of physical capital on national GDP growth are 0.25-0.4 [88]. That is, the benefit of an additional unit of traditional capital far exceed those for ICT capital. However, so many more units of ICT capital are provided that it again becomes important. For example, according to the Conference Board, although the share of Non-ICT capital in GDP is much higher than ICT capital (by 27 percentage points), the growth of capital services provided by ICT assets is much higher than physical capital (7.5 times faster growth), such that the contribution of capital services provided by ICT versus non-ICT on GDP growth is almost the same [89].

These results underscore the importance of computing to the economy, but also the need for exponential increases in computing for these contributions to be meaningful.

4.3 Analysis of variance

Figure 7 summarizes the R^2 results from these regressions, showing the fraction of the variation explained by computing power (dark) and that from all other factors (light).

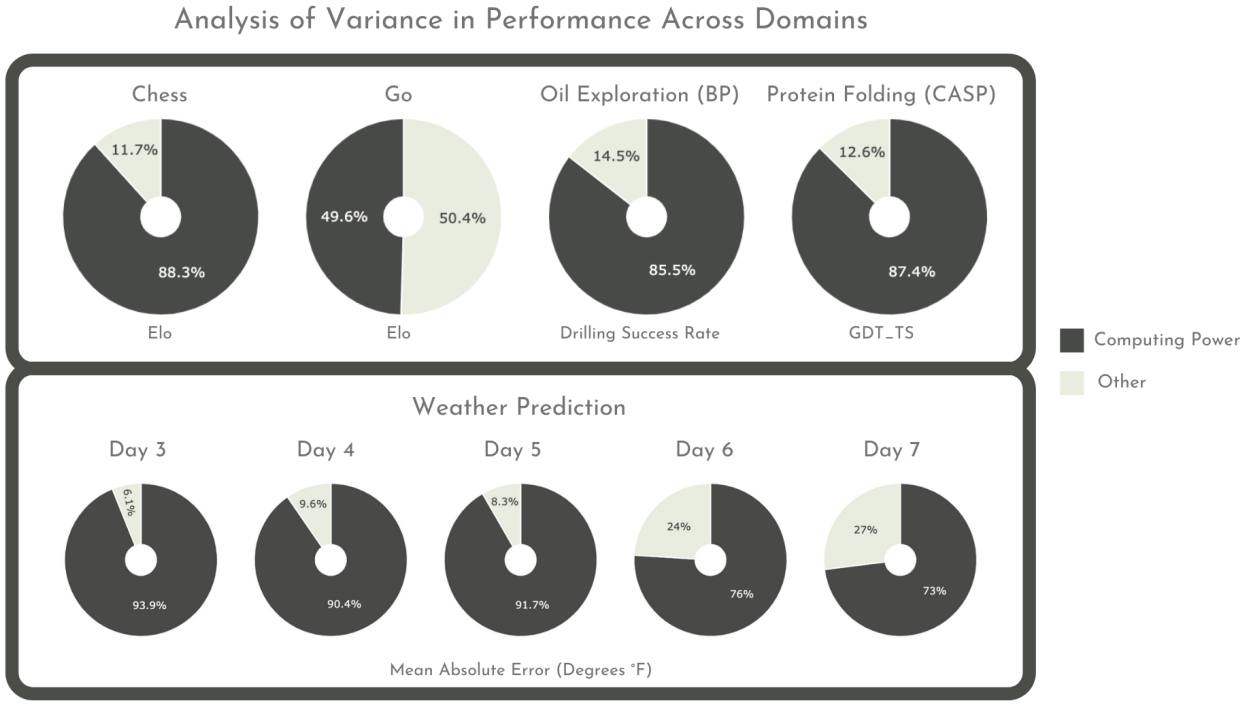

Figure 7: Analysis of Variance in Performance Across Domains.

As noted earlier, these results suggest that, in many areas, computing power has been overwhelmingly important as a source of gains in performance.

4.4 Implications for future performance improvement

This paper shows that, in the areas of Computer Chess, Computer Go, Weather Prediction, Protein Folding and Oil Exploration, progress depends on getting exponential increases in the amount of computing power. We propose that this is likely a broader phenomenon, because the computational techniques used for these areas are also used in many others. In particular, we suggest that progress in science and engineering often requires finer approximations, more complicated evaluation, repeated analysis under different starting conditions, greater search depth, and more degrees

of freedom. Support for this view is provided by Hyperion, a computing research firm, that gathered 690 examples of high-performance computing. They found that investment in High Performance Computing significantly improve economic success and increase scientific innovation across many areas [90].

Across our case study areas, Computer Chess, Computer Go, Weather Forecasting, Oil Exploration (BP), and Protein Folding (CASPs), we find that computing power has increased $140\times$, $390{,}000{,}000\times$, $1{,}600{,}000\times$, $160{,}000\times$, $58{,}000\times$ respectively, representing yearly compound growth rates of $14 - 90\%$.

Fortunately for the procurers of these systems, costs have not risen proportionally to these increases, principally because Moore's Law provided ever-cheaper computing power [29]. The exact pace of this countervailing improvement in costs from Moore's Law is, however, ambiguous. For example, indexes of this pace are created by macroeconomic agencies, such as the federal reserve banks, and by computer science enthusiasts, for example on Wikipedia [91, 92]. We find that none of these provide even an approximate fit for the systems where we are able to calculate cost and computing power. For this reason, we estimate the cost index for Computer Chess and Computer Go using actual system costs of programs over time[15]. We found that, in Computer Chess, computing power cost has decreased in a rate of 51% over time. This number is very consistent with what we see in Moore's law ($\sim$48% per year). However, in Computer Go, costs per gigaflop dropped much more rapidly, droping 117% per year, approaching the estimates coined by Huang's law [93] perhaps because of the usage of GPUs in modern systems.

Figure 8 show the cost of using ever more computing in these domains, after considering the falling cost of computing power (measured by reported actual system costs across years).

As Figure 8a shows, the cost of the system used for chess peaked at the time when Deep Blue was playing Kasparov (1996 and 1997) and has since then fallen. This is a good example of a phenomenon identified by Edelman "Unit costs of hardware continue to decline. It is only because usage seems to rise even faster that our total hardware costs continue to rise. It is to be hoped that sooner or later the product of the two - number of operations and cost per operation - will actually decline." [94]. In Chess, we see that indeed total costs did rise in the lead-up to playing Kasparov, but that post-Kasparov the desire for lower-cost systems won out.

At the other extreme, the cost of computation in Go (Figure 8b) had two remarkable moments. The first is mainly due to the emergence of the MCTS algorithm [95]. In a moment of breakthrough in the face of the hope of beating a professional Go player on a 19x19 board, in 2008, the creators of MoGo, in addition to adapting their program with this new algorithm, chose to use computational resources provided by supercomputers [16]. Eight years later, the cost of computing hit a new peak when AlphaGo defeated Lee Sedol.

By combining our estimates for the cost of computation and the effect of additional computation on performance, we are also able to show, in Figure c), how expensive it is to increase performance by 100 Elo points over time.

5 Conclusion

This article provides detailed quantitative case studies of five computing domains: the computing bellwethers of Chess and Go, and the economically important areas of weather prediction, protein folding, and oil exploration. In all cases, we examine the contribution of more computing power to better outcomes, but unlike most previous analyses that analyze based on the spending on IT, we analyze it based on the natural units of computing power. We find that computing power (and implicitly the algorithm changes needed to harness it) account for half or more of all improvement. The size of this contribution is remarkable since a change in computing power has only a tiny effect per unit. But since 1990, the compound growth rates of computing power have ranged from 38% to 145% in these areas. With such rapid exponential increases, computing power can nevertheless be the dominant source of improvements in these areas. Fortunately, such high growth has not come at the full price, since Moore's Law driven cost decreases made each unit of computing power more affordable. Nevertheless, all of these areas have experienced long stretches of time where costs rose by orders of magnitude.

Overall, this paper paints a coherent picture of computing power improvements as a central driver of progress across many areas over decades, quantifying long-held views about the centrality of I.T. in general, and of Moore's Law in particular as a driver of long-term performance improvement across society. It shows the importance of exponentially more computing power, and raises the specter of what we would lose absent that growth.

[15] Insufficient data was available for the actual costs of weather forecasting systems, protein folding and oil exploration, so we do not include them in this analysis.

[16] According to Couetoux, MCTS is an algorithm that generally scales well and it is expected that parallelizing it on a supercomputer would make it perform slightly better [96].

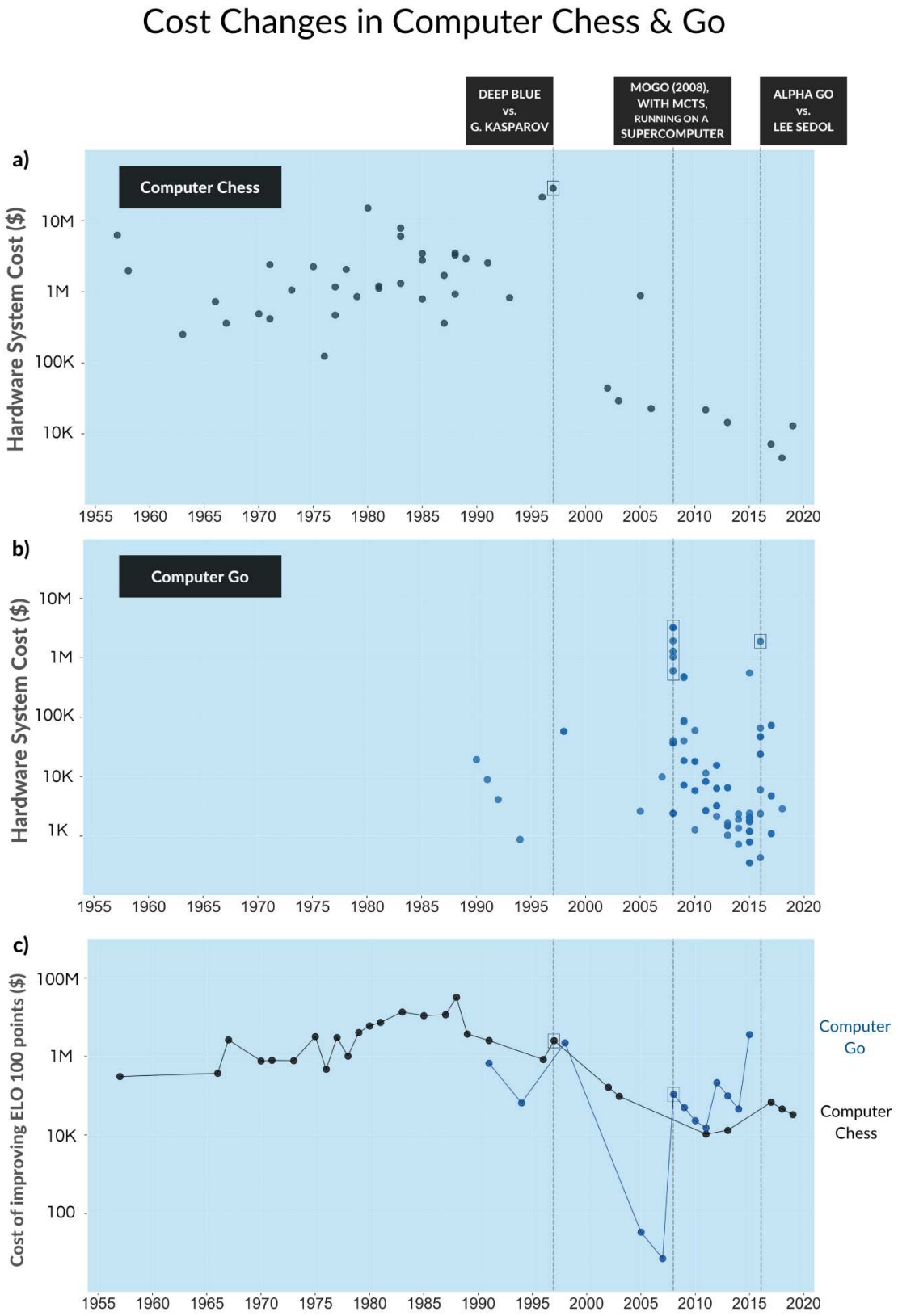

Figure 8: Inferred Cost Changes in Computer Chess & Go: a) Hardware system cost in Computer Chess. b) Hardware system cost in Computer Go. c) Cost of hardware needed to improve ELO by 100 points in Computer Go and Computer Chess.

References

[1] Paul Romer. Endogenous technological change. Working Paper 3210, National Bureau of Economic Research, December 1989.

[2] Paul Romer. Are nonconvexities important for understanding growth? Working Paper 3271, National Bureau of Economic Research, February 1990.

[3] Lorin M. Hitt and Erik Brynjolfsson. Productivity, business profitability, and consumer surplus: Three different measures of information technology value. *MIS Quarterly*, 20(2):121, June 1996.

[4] Erik Brynjolfsson. Paradox lost? firm-level evidence on the returns to information system spending. *Management Science*, 42:541–558, 02 1997.

[5] Daron Acemoglu. Artificial intelligence, automation and work. Working Paper 24196, National Bureau of Economic Research, January 2018.

[6] Daron Acemoglu. The race between man and machine: Implications of technology for growth, factor shares, and employment. *American Economic Review*, 108(6):1488–1542, June 2018.

[7] Zhuo (June) Cheng and Barrie R. Nault. Industry Level Supplier-Driven IT Spillovers. *Management Science*, 53(8):1199–1216, August 2007.

[8] David M. Byrne, Stephen D. Oliner, and Daniel E. Sichel. Is the Information Technology Revolution Over? *International Productivity Monitor*, 25:20–36, Spring 2013.

[9] Dale Jorgenson, Mun Ho, and Kevin Stiroh. Projecting productivity growth: Lessons from the u.s. growth resurgence. *Economic Review*, pages 1–13, 02 2002.

[10] Stephen Oliner and Daniel Sichel. Information technology and productivity: Where are we now and where are we going? *Economic Review*, 25:15–44, 02 2002.

[11] Kartik Ganju, Paul Pavlou, and Rajiv Banker. Does information and communication technology lead to the well-being of nations? a country-level empirical investigation. *MIS Quarterly*, 40:417–430, 06 2016.

[12] Rajiv D Banker and Robert J Kauffman. Reuse and productivity in integrated computer-aided software engineering: An empirical study. *MIS quarterly*, pages 375–401, 1991.

[13] Jennifer E Gerow, Varun Grover, Jason Thatcher, and Philip L Roth. Looking toward the future of it–business strategic alignment through the past. *MIS quarterly*, 38(4):1159–1186, 2014.

[14] Rajiv Sabherwal and Anand Jeyaraj. Information technology impacts on firm performance. *MIS quarterly*, 39(4):809–836, 2015.

[15] Lawrence W Foster and David M Flynn. Management information technology: Its effects on organizational form and function. *MIS quarterly*, pages 229–236, 1984.

[16] Alain Pinsonneault and Suzanne Rivard. Information technology and the nature of managerial work: From the productivity paradox to the icarus paradox? *MIS quarterly*, pages 287–311, 1998.

[17] Neil Thompson. The economic impact of moore's law: Evidence from when it faltered. *SSRN Electronic Journal*, 2017.

[18] Sarv Devaraj and Rajiv Kohli. Performance impacts of information technology: Is actual usage the missing link? *Management science*, 49(3):273–289, 2003.

[19] David M Byrne, Stephen D Oliner, and Daniel E Sichel. How fast are semiconductor prices falling? *Review of Income and Wealth*, 64(3):679–702, 2018.

[20] Adam Saunders and Erik Brynjolfsson. Valuing it-related intangible assets. *MIS Quarterly, Forthcoming*, 2015.

[21] Neil C. Thompson, Kristjan Greenewald, Keeheon Lee, and Gabriel F. Manso. The computational limits of deep learning, 2020.

[22] David Levy and Monroe Newborn. *All About Chess and Computers*. Springer Berlin Heidelberg, 1982.

[23] Danielle Muoio. Ai experts thought a computer couldn't beat a human at go until the year 2100, 2016. Accessed: 2021-11-09.

[24] Thomas J Teisberg, Rodney F Weiher, and Alireza Khotanzad. The economic value of temperature forecasts in electricity generation. *Bulletin of the American Meteorological Society*, 86(12):1765–1772, 2005.

[25] Jeffrey K Lazo, Rebecca E Morss, and Julie L Demuth. 300 billion served: Sources, perceptions, uses, and values of weather forecasts. *Bulletin of the American Meteorological Society*, 90(6):785–798, 2009.

[26] US EIA. Trends in us oil and natural gas upstream costs. *US Energy Information Administration*, 2016.

[27] Juexin Wang, Joseph Luttrell, Ning Zhang, Saad Khan, NianQing Shi, Michael X Wang, Jing-Qiong Kang, Zheng Wang, and Dong Xu. Exploring human diseases and biological mechanisms by protein structure prediction and modeling. In *Translational Biomedical Informatics*, pages 39–61. Springer, 2016.

[28] IMARC. Protein engineering market: Global industry trends, share, size, growth, opportunity and forecast 2022-2027, 2021. [Online; accessed 20-May-2022].

[29] Charles E Leiserson, Neil C Thompson, Joel S Emer, Bradley C Kuszmaul, Butler W Lampson, Daniel Sanchez, and Tao B Schardl. There's plenty of room at the top: What will drive computer performance after moore's law? *Science*, 368(6495), 2020.

[30] Neil C Thompson and Svenja Spanuth. The decline of computers as a general purpose technology. *Communications of the ACM*, 64(3):64–72, 2021.

[31] Paul M. Romer. Increasing returns and long-run growth. *Journal of Political Economy*, 94(5):1002–1037, 1986.

[32] Chad Syverson. What determines productivity? *Journal of Economic literature*, 49(2):326–65, 2011.

[33] Andrew Danowitz, Kyle Kelley, James Mao, John P Stevenson, and Mark Horowitz. Cpu db: recording microprocessor history. *Communications of the ACM*, 55(4):55–63, 2012.

[34] John P Holdren, Eric Lander, and H Varmus. Report to the president and congress: Designing a digital future: federally funded research and development in networking and information technology. *Executive Office of the President and President's Council of Advisors on Science and Technology*, page 148, 2010.

[35] Yash Sherry and Neil Thompson. How fast do algorithms improve. Technical report, Mimeo, 2020.

[36] Johannes K Fichte, Markus Hecher, and Stefan Szeider. A time leap challenge for sat-solving. In *International Conference on Principles and Practice of Constraint Programming*, pages 267–285. Springer, 2020.

[37] United States. Congress. House. Committee on Science. Subcommittee on Energy and Environment. *Budget Hearing on FY 1997 Request for DOE, NOAA, and EPA's Office of Research and Development (ORD); and Safe Drinking Water Act R&D Reauthorization: Hearing Before the Subcommittee on Energy and Environment of the Committee on Science, U.S. House of Representatives, One Hundred Fourth Congress, Second Session, March 21, 1996*. Number v. 2, no. 1 in Budget Hearing on FY 1997 Request for DOE, NOAA, and EPA's Office of Research and Development (ORD); and Safe Drinking Water Act R&D Reauthorization: Hearing Before the Subcommittee on Energy and Environment of the Committee on Science, U.S. House of Representatives, One Hundred Fourth Congress, Second Session, March 21, 1996. U.S. Government Printing Office, 1997.

[38] Wikipedia . National oceanic and atmospheric administration — Wikipedia, the free encyclopedia, 2021. [Online; accessed 9-November-2021].

[39] Paul M. Romer. *Advanced macroeconomics*. McGraw-Hill advanced series in economics. McGraw-Hill, 1996.

[40] Charles I. Jones. Paul romer: Ideas, nonrivalry, and endogenous growth. *The Scandinavian Journal of Economics*, 121(3):859–883, June 2019.

[41] H.J.R. Murray. *A History of Chess: The Original 1913 Edition*. Skyhorse, 2015.

[42] Claude E Shannon. Xxii. programming a computer for playing chess. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 41(314):256–275, 1950.

[43] Garry Kasparov and Frederic Friedel. Reconstructing turing's "paper machine". *EasyChair Preprint*, (3), 2017.

[44] Alex Bernstein and Michael de V. Roberts. Computer v. chess-player. *Scientific American*, 198(6):96–107, 1958.

[45] Bill Bill Wall. The machack attack by bill wall, 2008. [Online; accessed 9-November-2021].

[46] Computerworld. Computerworld, 1997.

[47] Hippke. Measuring hardware overhang, 2020. [Online; accessed 9-November-2021].

[48] Hippke. A closer look at chess scalings (into the past), 2021. [Online; accessed 9-November-2021].

[49] CCRL. Computer chess rating lists, 2021. [Online; accessed 9-November-2021].

[50] CJ Tan. Deep blue: computer chess and massively parallel systems. In *Proceedings of the 9th international conference on Supercomputing*, pages 237–239, 1995.

[51] British Go Association. A brief history of go, 2020. [Online; accessed 9-November-2021].

[52] John L Hennessy and David A Patterson. *Computer architecture: a quantitative approach*. Elsevier, 2011.

[53] THEECONOMIST. Showdown, 2016. [Online; accessed 9-November-2021].

[54] David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. Mastering the game of go without human knowledge. *nature*, 550(7676):354–359, 2017.

[55] Sylvain Gelly and David Silver. Monte-carlo tree search and rapid action value estimation in computer go. *Artificial Intelligence*, 175(11):1856–1875, 2011.

[56] Sylvain Gelly, Levente Kocsis, Marc Schoenauer, Michele Sebag, David Silver, Csaba Szepesvári, and Olivier Teytaud. The grand challenge of computer go: Monte carlo tree search and extensions. *Communications of the ACM*, 55(3):106–113, 2012.

[57] BBCNEWS. Google achieves ai 'breakthrough' by beating go champion, 2016. [Online; accessed 9-November-2021].

[58] Steven Borowiec. Alphago seals 4-1 victory over go grandmaster lee sedol, 2016. [Online; accessed 9-November-2021].

[59] Elizabeth Gibney. Go players react to computer defeat. *Nature News*, 2016.

[60] Nick Wedd. A list of computer go tournaments, 2018. [Online; accessed 9-November-2021].

[61] Jim Foerster. What's the difference between private weather companies and the national weather service?, 2020. [Online; accessed 9-November-2021].

[62] Marshall Shepherd. When it comes to u.s. weather forecasting: Private, public or both?, 2016. [Online; accessed 9-November-2021].

[63] Steve Graham. Weather forecasting through the ages: Feature articles. 2002.

[64] WMO. Observations, 2020.

[65] Claire Flynn. Forecasts in retrospect: A history of numerical weather prediction, 2018. [Online; accessed 9-November-2021].

[66] National Weather Service. Gfs (the global forecast system) documentations, 2020. [Online; accessed 9-November-2021].

[67] USA. Restoring u.s. leadership in weather forecasting: Hearing before the subcommittee on environment, committee on science, space, and technology, house of representatives, one hundred thirteenth congress, first session., 2013.

[68] ECMWF. Forty years of acvancing global numerical weather prediction, 2015. [Online; accessed 9-November-2021].

[69] Thomas C Schulthess, Peter Bauer, Nils Wedi, Oliver Fuhrer, Torsten Hoefler, and Christoph Schär. Reflecting on the goal and baseline for exascale computing: a roadmap based on weather and climate simulations. *Computing in Science & Engineering*, 21(1):30–41, 2018.

[70] Philipp Neumann, Peter Düben, Panagiotis Adamidis, Peter Bauer, Matthias Brück, Luis Kornblueh, Daniel Klocke, Bjorn Stevens, Nils Wedi, and Joachim Biercamp. Assessing the scales in numerical weather and climate predictions: will exascale be the rescue? *Philosophical Transactions of the Royal Society A*, 377(2142):20180148, 2019.

[71] Peter Bauer. Today's weather forecast: Good with a strong chance of improvement, 2016.

[72] Peter Lynch. *The emergence of numerical weather prediction: Richardson's dream*. Cambridge University Press, 2006.

[73] WPC. Wpc verification statistics. `https://www.wpc.ncep.noaa.gov/html/hpcverif.shtml`, 2020. (Accessed on 05/16/2022).

[74] Yang Zhang. Progress and challenges in protein structure prediction. *Current opinion in structural biology*, 18(3):342–348, 2008.

[75] Peter E Wright and H Jane Dyson. Intrinsically unstructured proteins: re-assessing the protein structure-function paradigm. *Journal of molecular biology*, 293(2):321–331, 1999.

[76] Cyrus Levinthal. Mossbauer spectroscopy in biological systems. In *Proceedings of a meeting held at Allerton House. P. Debrunner, JCM Tsibris, and E. Munck, editors. University of Illinois Press, Urbana, IL*, 1969.

[77] CASP. Protein structure prediction center, 2020.

[78] Investopedia. What percentage of the global economy is the oil and gas drilling sector?, 2020.

[79] Amy Harvey. How do average costs compare among various oil drilling rigs?, 2020.

[80] Kirk Rundle, Susan Nissen, Richard Lockhart, and Ernest Morrison. 3-d seismic applications by independent operators in kansas. *Young*, 1:35, 2003.

[81] A Bamberger, G Chavent, Ch Hemon, and P Lailly. Inversion of normal incidence seismograms. *Geophysics*, 47(5):757–770, 1982.

[82] Richard J Davies, Simon A Stewart, Joseph A Cartwright, Mark Lappin, Rodney Johnston, Scot I Fraser, and Alistair R Brown. 3d seismic technology: are we realising its full potential? *Geological Society, London, Memoirs*, 29(1):1–10, 2004.

[83] DA Karavaev, BM Glinsky, and VV Kovalevsky. A technology of 3d elastic wave propagation simulation using hybrid supercomputers. pages 26–33, 2015.

[84] EIA. Annual energy review. annual review, Office of Energy Statistics, September 2012.

[85] Doug Black. Eni's gpu-based hpc4 speeds oil reservoir simulations, 2018.

[86] iHSMarkit. Us data online, 2020.

[87] NWS Director. Increasing noaa's computational capacity to improve global forecast modeling. 2010.

[88] Nicholas Crafts and Pieter Woltjer. Growth accounting in economic history: findings, lessons and new directions. *Journal of Economic Surveys*, 2020.

[89] The-Conference-Board. Total economy database. database, The-Conference-Board-Inc, 2021.

[90] Earl C Joseph, Steve Conway, and Chirag Dekate. Creating economic models showing the relationship between investments in hpc and the resulting financial roi and innovation? and how it can impact a nation's competitiveness and innovation. *International Data Corporation (IDC), Technical report*, 2013.

[91] Federal-Reserve-Bank of St-Louis. Economic data: Producer price index by industry: Semiconductor and other electronic component manufacturing. Technical report, Federal-Reserve-Bank-of-St-Louis, 2021.

[92] Wikipedia . Flops — Wikipedia, the free encyclopedia, 2021. [Online; accessed 9-November-2021].

[93] Mims, Christopher. Huang's law is the new moore's law, and explains why nvidia wants arm, 2020.

[94] Franz Edelman. Managers, computer systems, and productivity. *Mis Quarterly*, 1981.

[95] Gabriel F Manso and Neil C Thompson. What the game of go can tell us about algorithm progress and the end of moore's law. MIMO, unpublished, 2022.

[96] Adrien Couëtoux, Martin Müller, and Olivier Teytaud. Monte carlo tree search in go. 2013.

[97] David Levy. Computer chess compendium. 2013.

[98] David NL Levy. *Computer Games I*. Springer Science & Business Media, 2012.

[99] M.R.B. Clarke. *Advances in Computer Chess 3*. Advances in Computer Chess. Pergamon Press, 1982.

[100] Monroe M. Newborn. Computer chess: Ten years of significant progress. *Adv. Comput.*, 29:197–250, 1989.

[101] Peter W Frey. *Chess skill in man and machine*. Springer Science & Business Media, 2012.

[102] Edwin D Reilly. *Concise encyclopedia of computer science*. John Wiley & Sons, 2004.

[103] Feng-hsiung Hsu. Computer chess, then and now: The deep blue saga. In *International Symposium on VLSI Technology, Systems, and Applications*, pages 153–154. IEEE Computer Society, 1997.

[104] Hans Berliner. Computer chess (monroe newborn). *SIAM Review*, 18(3):514, 1976.

[105] David Levy and Monroe Newborn. *All About Chess and Computers: Chess and Computers and More Chess and Computers*. Springer Science & Business Media, 2012.

[106] Monroe Newborn and Monty Newborn. *Deep Blue: an artificial intelligence milestone*. Springer Science & Business Media, 2003.

[107] Jamie Parker Pearson. *Digital at work: Snapshots from the first thirty-five years*. digital press Burlington, MA, 1992.

[108] Nicolae Sfetcu. *Gaming Guide-Gambling in Europe*. Nicolae Sfetcu, 2016.

[109] Monty Newborn. *Kasparov versus Deep Blue: Computer chess comes of age*. Springer Science & Business Media, 2012.

[110] Ernst A Heinz. *Scalable search in computer chess: Algorithmic enhancements and experiments at high search depths*. Springer Science & Business Media, 2013.

[111] Monty Newborn. *Beyond Deep Blue*. Springer London, 2011.

[112] Nils J Nilsson. *The quest for artificial intelligence*. Cambridge University Press, 2009.

[113] Hans P Moravec. *Robot: Mere machine to transcendent mind*. Oxford University Press on Demand, 2000.

[114] Kenneth W. Regan. *Rating Computer Science via Chess*, pages 200–216. Springer International Publishing, Cham, 2019.

[115] Jonathan Schaeffer and T Anthony Marsland. *Computers, Chess and Cognition*. Springer, 1990.

[116] Feng-hsiung Hsu. Ibm's deep blue chess grandmaster chips. *IEEE micro*, 19(2):70–81, 1999.

[117] Berthold K. P. Horn. The image dissector "eyes". 1969.

[118] Murray Campbell, A.Joseph Hoane, and Feng hsiung Hsu. Deep blue. *Artificial Intelligence*, 134(1):57–83, 2002.

[119] Danny Kopec, Monty Newborn, and Mike Valvo. The 22d annual acm international computer chess championship. *Commun. ACM*, 35(11):100–110, nov 1992.

[120] D. Kopec. The 23rd acm international computer-chess championship report on the tournament. *J. Int. Comput. Games Assoc.*, 16:38–40, 1993.

[121] Manuel Tapp. Reseña histórica del ajedrez por computadora (v), 1988.

[122] M. Newborn and D. Kopec. Results of acm's eighteenth computer chess championship. *Commun. ACM*, 31(8):992–995, aug 1988.

[123] COMPUTERWORD. Computerworld. https://books.google.gp/books?id=ArfLuk6qMAMC&printsec=frontcover#v=onepage&q&f=false, 1976. (Accessed on 12/12/2021).

[124] COMPUTERWORD. Computerworld. https://books.google.co.ke/books?id=mskkmVkpIUcC&printsec=frontcover#v=onepage&q&f=false, 1987. (Accessed on 12/12/2021).

[125] Will Knight. Kasparov flummoxes chess computer. https://www.newscientist.com/article/dn3312-kasparov-flummoxes-chess-computer/, 2003. (Accessed on 12/12/2021).

[126] ChessBase. Chess news. https://en.chessbase.com/, 2021. (Accessed on 12/12/2021).

[127] Bill Wall. Computers and chess - a history, 2017.

[128] Bill Wall. Kaissa, 2019.

[129] Bill Wall. Machack attack, 2008.

[130] Chess Maniac. Kaissa chess program, 2014.

[131] Reddit. How many positions per second, approximately, was leela chess zero calculating against tang in the rapid games?, 2017.

[132] Pete. Alphazero crushes stockfish in new 1,000-game match, 2019.

[133] Bill Wall. Chess engines and chess programs, 2021.

[134] ChessGenius. Official website of chess genius, 2021.

[135] IBM. A brief history of risc, the ibm rs/6000 and the ibm eserver pseries, 2021.

[136] Gian-Carlo Pascutto. Sjeng : a chess-and-variants playing program, 2021.

[137] Tapani Raiko. Computer go challenge at alternative party 2009. http://users.ics.aalto.fi/praiko/altparty2009/, 2009. (Accessed on 12/12/2021).

[138] AltParty. Alternative party 2009 "man meets machine". http://www.altparty.org/2009/competition-rules.html#csc-comp, 2009. (Accessed on 12/12/2021).

[139] Peter Drake. 2008 us go congress – computer go. http://computer-go.info/events/na/2008/index.html, 2008. (Accessed on 12/12/2021).

[140] Pavol Lisy. Cho chikun 9p defeats ai deepzen by 2-1. https://www.eurogofed.org/index.html?id=89, 2016. (Accessed on 12/12/2021).

[141] Raymond G. Snatzke. Interviews with franz-josef dickhut and rémi coulom. https://blog.codecentric.de/en/2014/10/codecentric-go-challenge-2014-interviews-franz-josef-dickhut-remi-coulom/, 2014. (Accessed on 12/12/2021).

[142] TheGuardian. Go, going, gone? https://www.theguardian.com/technology/2009/apr/30/games-software-mogo, 2009. (Accessed on 12/12/2021).

[143] Chris Garlock. Computer beats pro at u.s. go congress. https://www.usgo.org/sites/default/files/ejournal_archive/20080807/20080807.htm, 2008. (Accessed on 12/12/2021).

[144] WIRED. In two moves, alphago and lee sedol redefined the future. https://www.wired.com/2016/03/two-moves-alphago-lee-sedol-redefined-future/, 2016. (Accessed on 12/12/2021).

[145] BritishGoAssociation. Microgo1 vs. 3kyu. https://www.britgo.org/bgj/06316.html, 1984. (Accessed on 12/12/2021).

[146] Eliezer Yudkowsky. Alphago zero and the foom debate. https://www.lesswrong.com/posts/shnSyzv4Jq3bhMNw5/alphago-zero-and-the-foom-debate, 2017. (Accessed on 12/12/2021).

[147] BritishGoAssociation. History of go-playing programs. https://www.britgo.org/computergo/history, 2018. (Accessed on 12/12/2021).

[148] Peter Shotwell. A time line of supercomputer go: Temporal difference learning to monte carlo programing. https://www.britgo.org/computergo/history, 2011. (Accessed on 12/12/2021).

[149] Jay Burmeister and Janet Wiles. Computer go. https://staff.itee.uq.edu.au/janetw/Computer%20Go/CS-TR-339.html, 1995. (Accessed on 12/12/2021).

Supplemental Materials

6 First Differences

One might worry that correlating two-time trends, both of which are rising over time, might create spurious correlations. We address this by repeating our analysis of how computing power affects performance using a first-differences approach.[17]

In each area, we see statistically significant relationships for how computing power affects performance, although for weather we only see if for the Day 3 predictions.

Table 4: First difference analysis

	First Difference			
	Chess	Go	Protein Folding (CASP)	Oil Exploration (BP)
	Elo	Elo	GDT_TS	Success Rate
$Constant$	24.52	1.45	-1.84	0.17
	(195.14)	(34.79)	(3.56)	(0.10)
$log_{10}(Computing\ Power)$	93.05*	208.53***	7.46**	0.22**
	(55.26)	(37.09)	(1.22)	(0.10)
Observations	82	103	4	27
R^2	0.03	0.24	0.95	0.18
Adjusted R^2	0.02	0.24	0.92	0.15
Residual Std. Error	1766.55(df = 80)	351.72(df = 101)	7.12(df = 2)	0.42(df = 25)
F Statistic	2.83* (df = 1; 80.0)	31.61*** (df = 1; 101)	37.52** (df = 1; 2)	5.46** (df = 1; 25)

	Weather Prediction				
	Day 3 (3)	Day 4 (4)	Day 5 (5)	Day 6 (6)	Day 7 (7)
$Constant$	-0.0337	-0.0901	-0.0759	-0.0586	-0.0192
	(0.0489)	(0.0745)	(0.0572)	(0.0819)	(0.0855)
$log_{10}(Computing\ Power)$	-0.2127***	-0.0899	-0.0954	-0.0598	-0.1744
	(0.0639)	(0.0973)	(0.0747)	(0.2564)	(0.2676)
Observations	21	21	21	17	17
R^2	0.3683	0.0430	0.0790	0.0036	0.0275
Adjusted R^2	0.3350	-0.0074	0.0306	-0.0628	-0.0373
Residual Std. Error	$0.1861\ (df = 19)$	$0.2834\ (df = 19)$	$0.2174\ (df = 19)$	$0.2588\ (df = 15)$	$0.2702\ (df = 15)$
F Statistic	$11.0769^{***}\ (df = 1; 19)$	$0.8536\ (df = 1; 19)$	$1.6306\ (df = 1; 19)$	$0.0544\ (df = 1; 15)$	$0.4247\ (df = 1; 15)$

Note: *p<0.1; **p<0.05; ***p<0.01

7 Data Sources

To assess the progress of Computer Chess programs since 1957 (Bernstein's program), we gathered data from a the resources shown in Table 7.

[17] This is also a more strenuous test because it requires immediate causality, whereas we might expect it to take time for software designers and modelers to fully take advantage of new hardware.

Table 5: Computer Chess Data Sources

Source	Reference
Books & Research Papers	Computer Chess Compendium [97]
	Computer Games I [98]
	Advances in Computer Chess 3 [99]
	Computer Chess: Ten Years of Significant Progress [100]
	Chess Skill in Man and Machine [101]
	Concise Encyclopedia of Computer Science [102]
	Computer Chess, Then And Now: The Deep Blue Saga [103]
	A.C.M monograph series [104]
	All About Chess and Computers [105]
	Deep Blue: An Artificial Intelligence Milestone [106]
	Digital at Work - Snapshots from the first thirty-five years [107]
	The Game of Chess [108]
	Kasparov versus Deep Blue [109]
	Scalable Search in Computer Chess [110]
	Beyond Deep Blue Chess in the Stratosphere-Springer-Verlag London [111]
	The Quest for Artificial Intelligence [112]
	ROBOT, Moravec [113]
	Rating Computer Science Via Chess [114]
	Computers, Chess, and Cognition [115]
	IBM's Deep Blue Chess Grandmaster Chips [116]
	The Image Dissector "Eyes" [117]
	Deep Blue [118]
Conference and Tournament Reports	The 22d Annual ACM International Computer Chess Championship [119]
	The 23rd ACM International Computer-Chess Championship Report on the Tournament [120]
	The 22d Annual ACM International Computer Chess Championship [119]
	Reseña histórica del ajedrez por computadora (V) [121]
	Results of ACM's eighteenth computer chess championship [122]
Newspaper & Magazine Articles	Computerworld Vol.X [123]
	Computerworld, Vol.XXI [124]
	Kasparov flummoxes chess computer [125]
Other	Computer Chess Rating Lists [49]
	ChessNews [126]
	Computers and Chess - A History [127]
	Kaissa [128]
	MacHack Attack [129]
	KAISSA chess program [130]
	How many positions per second, approximately, was Leela Chess Zero calculating against Tang in the rapid games? [131]
	AlphaZero Crushes Stockfish In New 1,000-Game Match [132]
	Chess Engines and Chess Programs [133]
	Official Website of Chess Genius [134]
	A Brief History of RISC, the IBM RS/6000 and the IBM eServer pSeries [135]
	Sjeng : a chess-and-variants playing program [136]

To assess the progress of Computer Go Programs, we gathered data from the resources shown in Table 6.

Table 6: Computer Go Data Sources

Source	Reference
Conference & Tournament Reports	Computer Go Challenge at Alternative Party 2009 [137]
	Alternative Party 2009 "Man Meets Machine" [138]
	2008 US Go Congress – Computer Go [139]
	Cho Chikun 9p defeats AI DeepZen by 2-1 [140]
	Interviews with Franz-Josef Dickhut and Rémi Coulom [141]
Newspaper & Magazine Articles	Go, going, gone? [142]
	Computer Beats Pro at U.S. Go Congress [143]
	In Two Moves, AlphaGo and Lee Sedol Redefined the Future [144]
	Microgo1 vs. 3kyu [145]
Forum Articles & Community Discussions	AlphaGo Zero and the Foom Debate [146]
	History of Go-playing Programs [147]
	A Time Line of Supercomputer Go: Temporal Difference Learning to Monte Carlo Programing [148]
Other	Computer Go [149]
	A database of go-playing programs [60]

8 Protein Folding

Since our main analysis has a limited sample size, we also conduct a second analysis of protein folding's dependence on computing power increases by analyzing the scaling performance of large distributed folding projects. For instance, the Pande Lab at Stanford University uses 100,000 CPUs distributed throughout the world through the Folding@Home project. Similarly, Rosetta@home, developed in David Baker's Lab at University of Washington, uses computing from tens of thousands of distributed computers. To avail ourselves of all the CASP data, we do this analysis at 3 different levels of protein folding complexity (the levels reflect whether there is a known protein of similar structure that can be used as a guide to decrease the analysis that algorithms must do). These levels range from template-based modeling (TBM), which is easier because of the availability of models, to free modeling (FM) which is harder because it lacks such models. There is also an intermediate level (TBM-FM). Using data provided by the Rosetta@home group which performed experiments applying more computing power (by calculating more models for each candidate protein), we observe that for each 10× increase in computation, TBM performance increases by 6%, TBM/FM performance increases by 8%, and FM performance improves by 3% (all statistically significant at p-value < 0.01, see Figure 8 for details). Their tests show that variance in computing power explain 77-97% of variance, although this dominance isn't surprising since other variables aren't changing simultaneously.

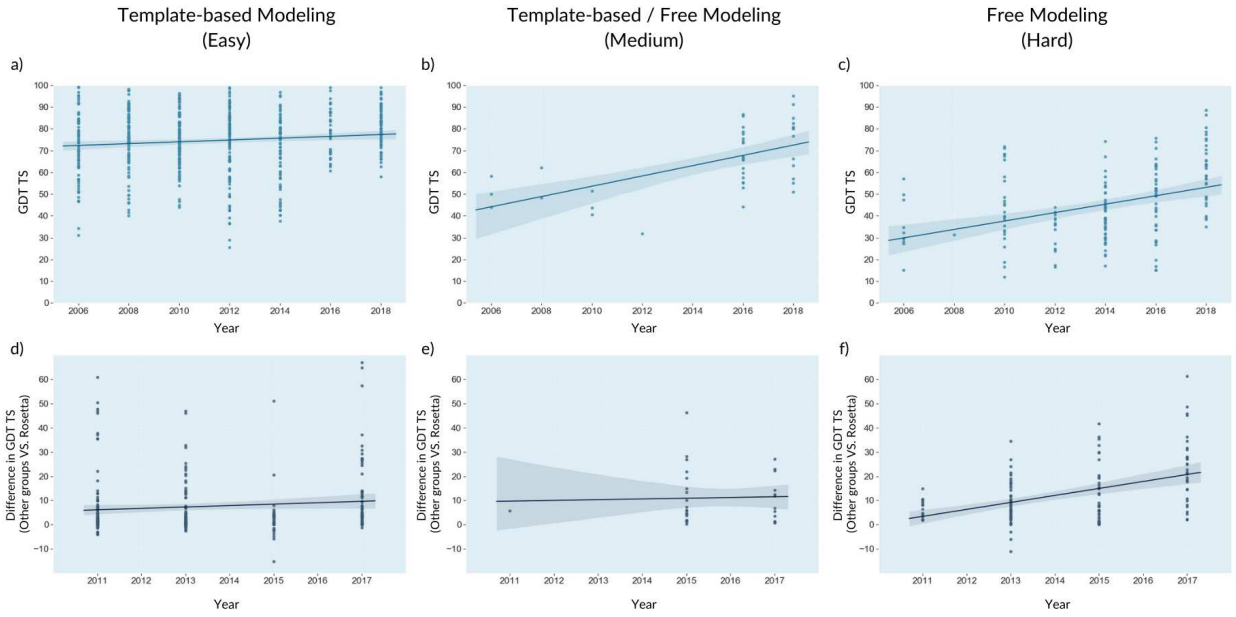

Figure 9: CASP performance in 3 types of protein folding tasks, as measured by protein structure similarity score (GDT-TS) (a-c) Performance (1 dot = 1 model); (d-f) Performance of other server groups compared to to the Rosetta server group that used constant computation.

Table 7: Rosetta@Home: Effect of additional computation on protein folding performance by the difficulty of the folding task.

	Hard	Medium	Easy
	(1)	(2)	(3)
$Constant$	0.321^{***}	0.656^{***}	0.770^{***}
	(0.002)	(0.003)	(0.001)
$log_{10}(Computing\ Power)$	0.003^{***}	0.008^{***}	0.006^{***}
	(0.001)	(0.001)	(0.000)
Observations	10	10	10
R^2	0.771	0.844	0.969
Adjusted R^2	0.742	0.824	0.965
Residual Std. Error (df = 8)	0.002	0.003	0.001
F Statistic (df = 1; 8)	26.892^{***}	43.110^{***}	247.901^{***}
Note:			*p<0.1; **p<0.05; ***p<0.01