On the Viability of Self-reproducing Classical Machines

Virgil Griffith, Doyne Farmer[1], Ole Peters[1]

August 23, 2005

Santa Fe Institute REU Paper

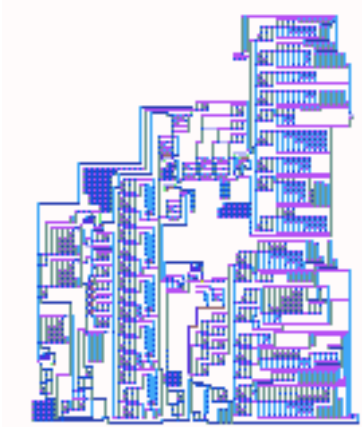
Abstract

In the early 1950's John von Neuman was investigating self-reproduction of analog machines from a "sea of parts." For simplicity, he assumed an idealized error-free environment and eventually developed a self-reproducing Cellular Automaton. John von Neumann's 29 stage 200,000 cell self-reproducing machine (as well as other self-reproducing CA's) was complex and extremely brittle to any noise in the environment. Given its extreme sensitivity to noise, Von Neuman's model is unrealistic for any physical machine, and is also less interesting as it makes evolution impossible.

This leads to the question of what are the necessary conditions to achieve self-reproduction in a stochastic environment in which there are errors in reproduction. We developed a simplified model to study when it is possible for a machine to copy itself with enough fidelity to be viable. One worries that if a machine is too complicated, and the noise is too high, there will be an error catastrophe in which the machine cannot reproduce sufficiently accurate copies to sustain itself. We make an estimate of the critical error threshold under a given set of assumptions about the fitness function characterizing the machines and their environment.

1 Introduction

Early work in machine self-replication began with in the late 1940's with Jon von Neumann at Los Alamos National Laboratory. At the time von Neumann was studying the logical possibility of self-replication. He initially thought about physical machines working from a sea of parts but eventually decided actual machines were too complicated to be tractable. Based on a suggestion from Stan Ulam he made began to study idealized forms of self-replication in an error-free environment. In 1966 with the publication of *The Theory of Self-reproducing Automata* he not only proved that self-replication was possible, but had designed a highly intricate 29-stage ~200,000 cell cellular automaton that was able to generate copies of itself. Subsequent work in cellular automata theory greatly simplified von Neumann's initial design. However, the vast majority of subsequent work also assumed an error-free environment.



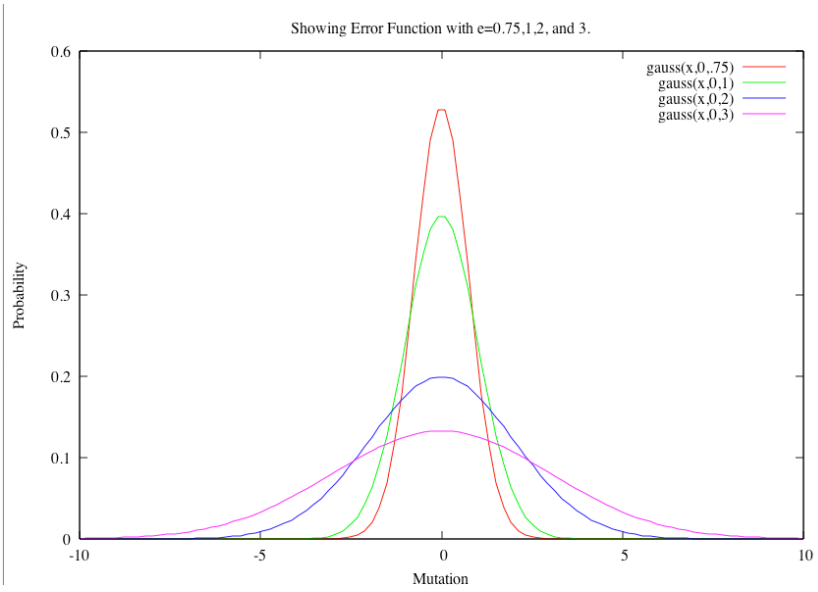
However, actual reproduction takes place in a complex and noisy world with numerous errors and mutations occurring between one generation and the next. In this study we investigate simulations of asexual self-replicating machines of varying amounts of complexity, copying fidelity, and robustness.

[1] Description of the model reproduced from a working draft by Farmer and Peters.

2 The Model

We consider a simple representation of a classical machine as a list of real numbers. This list could include the dimension, location, and material composition of all the parts, as well as any procedures or routines that are necessary for the machine to run itself. The machine can thus be represented by a vector x, whose dimension D is the length of the list, and can be identified as the complexity of the machine. Here we mean complexity simply in the sense of complication – a more complex machine is one with more essential parts. When a machine reproduces the machine makes a copy of itself, which contains some errors. In each generation the ensemble of N_t machines x_t gives rise to a new ensemble of N_t+1 machines x_{t+1}. The number of copies that machine x_i makes is dependent on its fitness $F(x_{it})$. For simplicity we will assume non-overlapping generations.

In passing from one generation to the next we assume there are copying errors, i.e. $x_{i,t+1} = x_{it} + M_{it}$, where M_{it} is a D dimensional vector of independent and identically distributed random numbers. For simplicity we assume that the components of M_{it} are random numbers drawn from a normal distribution with mean zero and standard deviation ε.



The fitness $F(x_{it})$ is a real valued positive function with D arguments with a range from zero to one. To determine the number of children produced by each machine, we multiply the fitness by the maximum number of children η. This result is simply the mean number of off-spring of machine x_{it} in generation t+1. As each machine is copied we have to deal with the problem that the number of children typically has a noninteger value (e.g. 1.1). Obviously, a machine can only have an integer number of offspring. For this, we just assume that the number of offspring is either the integer just greater or the integer just less than the η * fitness, and that the probability of the two outcomes is determined by the difference. I.e., if the fitness is F, let f' be the largest integer less than F, and let the probability of f' offspring be $F - f'$ and the probability of $f' + 1$ offspring be $f' + 1 - F$.

There are many plausible fitness functions mapping x to a fitness between zero and one. However, for our simulations we assumed a gaussian fitness function with an argmax at a given value β and standard deviation f (f is not related to the number of offspring f'). For convenience we arbitrarily set $\beta = 0$. We can think of the value of f as determining the robustness of a machine's parts. The larger the value of f, the less a machine is punished for having values far away from β. We initially begin with a population of N_0 machines all clones of each other with perfect parts β. However, regardless of the initial values of x at N_0 the machines will eventually converge upon a stable distribution that is a function of ε, f, and η. For given values of ε, f, and η we can by simulation study the behavior of N_t and the evolution of $\Pi_t(x)$, the distribution of x at timestep t.

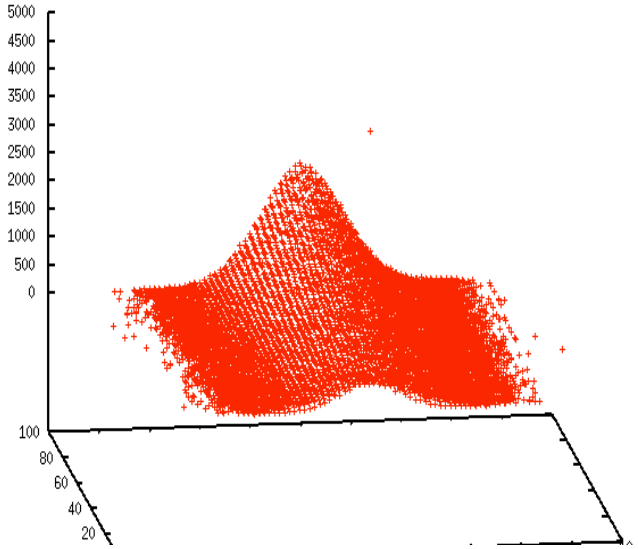
Note that we are assuming no interaction between the machines, e.g. there is no competition for resources. As a result of no finite limit, we expect the population will either explode or go to zero. If it explodes, a machine with the given parameters of ε, f, and η is viable. In the real world competition dynamics would eventually limit the size of the population, but this population would still be non-zero. If $N_\infty = 0$, then a machine given the given parameters is not viable.

If machines have only a single vital part, it is easy to know the extreme cases. If $f >> \varepsilon$ and $\eta >> 1$, then the machine will be viable. If $\varepsilon >> f$, errors will overpower any ability of the fitness function to shape the distribution causing the machines to quickly die out. If $\eta < 1$, then for any value of x (even $x=\beta$), the machines will not be able to repopulate sufficiently to sustain their population and will eventually shrink to zero.

For more than one vital part, the appropriate fitness function to map x to f' is less obvious. In our simulations we explored the least generous case of taking the minimum fitness across all dimensions of x. However, there exist many other more generous fitness functions for the multi-gene case that have yet to be explored.

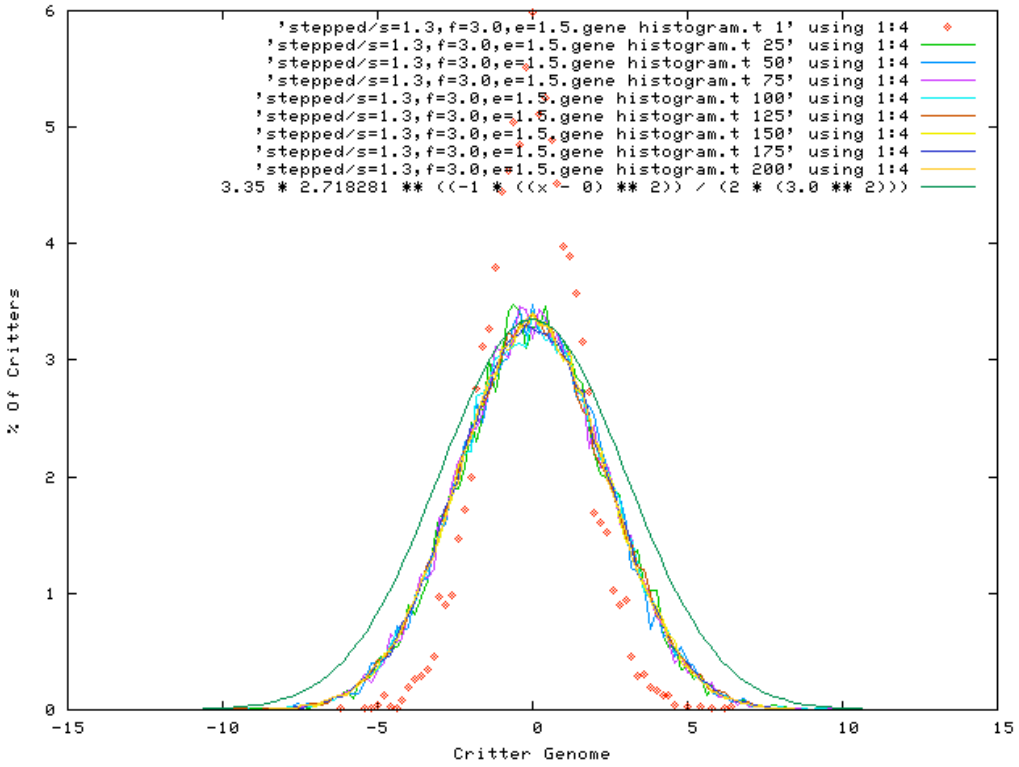
3 Results

We begin with 5,000 machines at timestep 0 with a single part x initially set to β (the highest fitness), for convenience β is zero (the single point at above the curve is timestep 0). On the figure we see x along the x-axis, time along the y-axis, and the number of machines with that value of $\Pi(x)$ along the z-axis. The distribution of machines quickly fans out due to the mutation rate. However, the fitness function gradually pulls the distribution



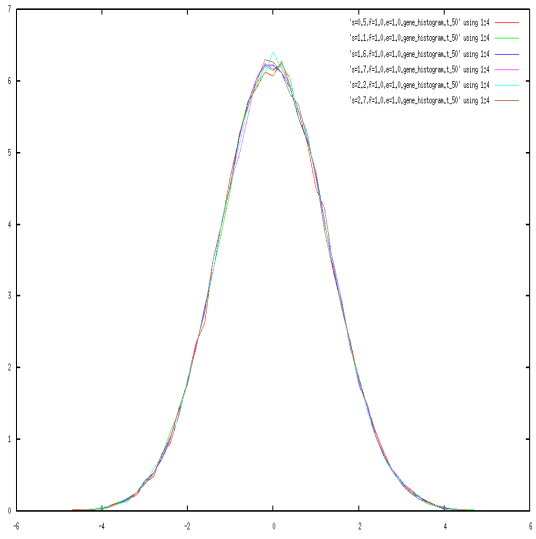
inward eventually resulting in a stable rising population by timestep 100. To assist us in understanding the dynamics of $\Pi_t(x)$, we can think of the values of ε and f opposing one another. ε tries to push the distribution outward while f continually refocuses the distribution towards β.

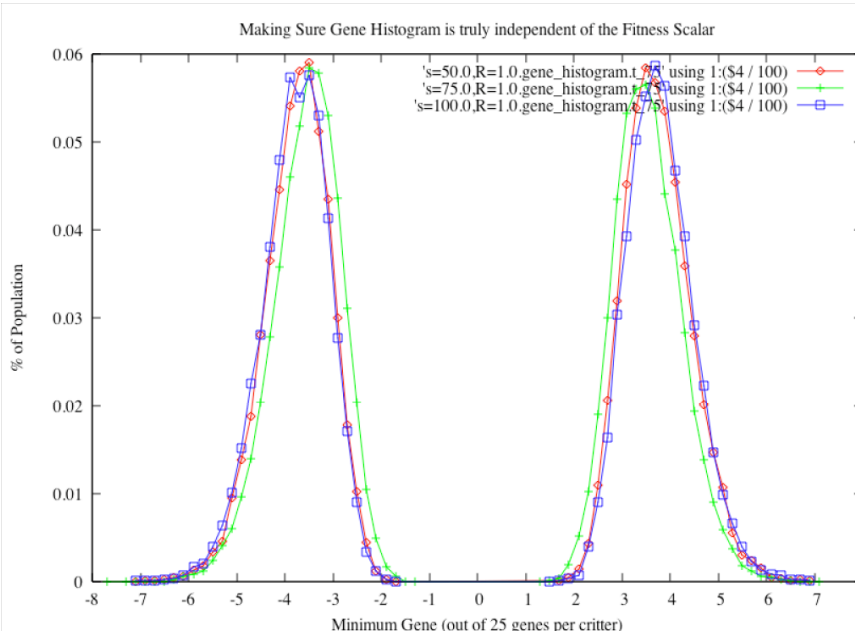
To better assess the dynamics in the distribution of genes $\Pi_t(x)$, over time it is helpful to convert the above 3-dimensional representation to multiple staged 2-dimensional representation. The figure below shows $\Pi_t(x)$ at t=1, 25, 50, 75, 100, 125, 150, 175, and 200. As before the entire population starts out with a single part with $x = \beta$. The red dots represent $\Pi_1(x)$. We can see that already by timestep 1 $\Pi(x)$ has already fanned out considerably. By timestep 25 we can see that $\Pi(x)$ has reached a stable state. This stable state continues indefinitely with the only fluctuations due to a finite population size.



3.1 Some Surprises

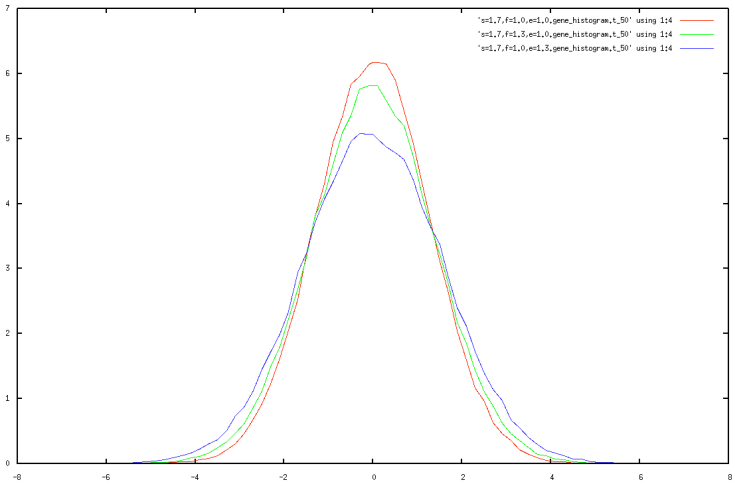
We had initially stated our input parameters as ε,f, and η. However, we discovered that the behavior of the distribution of x was invariant to both η and the precise values of ε and f. The attractor (stable) state was solely determined by the ratio (ℜ) of f / ε. This was true of both the single gene and multi-gene cases. The first graph shows when re-normalized the histogram $\Pi(x)$ is independent of the precise values of f and ε. The curves below left show the asymptotic distribution for varying parameters f and ε but holding the ratio (ℜ), constant. We can see the distribution is the same. On the right we see the same kind of plot except for 25 vital parts instead of one. We see the same trend





holds. The rightward multi-gene graph is especially interesting. Not only does it follow the same patterns as the single gene, but it shows specialization. Here there are two distinct populations of machines which have roughly the same fitness, but the constuition of their least fit gene is markedly different. Although this phenomena was not explicitly predicted from the outset it makes sense considering fitness function can only "see" the fitness of the genes, not the genes themselves.

Further evidence that it is indeed the ratio ($\Re$) of f / ε that matters and not any other relationship between these two variables (such as f * ε) can seen in the histogram below.



What we see here is the gene histogram Π(x) at the attractor state for slightly different values of f and ε. The red curve is for both f,ε = 1. The green curve is for ε=1 and f=1.3, and the blue curve ε=1.3 and f=1. If Π(x) were determined by f * ε or any other commutative operator both the blue and green curve would lie on top of one another. This fact coupled with intuitive understanding of ε pushing the graph outward and f pulling the graph inward gives us great confidence that is ℜ, and only ℜ that matters.

3.2 The Boundary of Survival

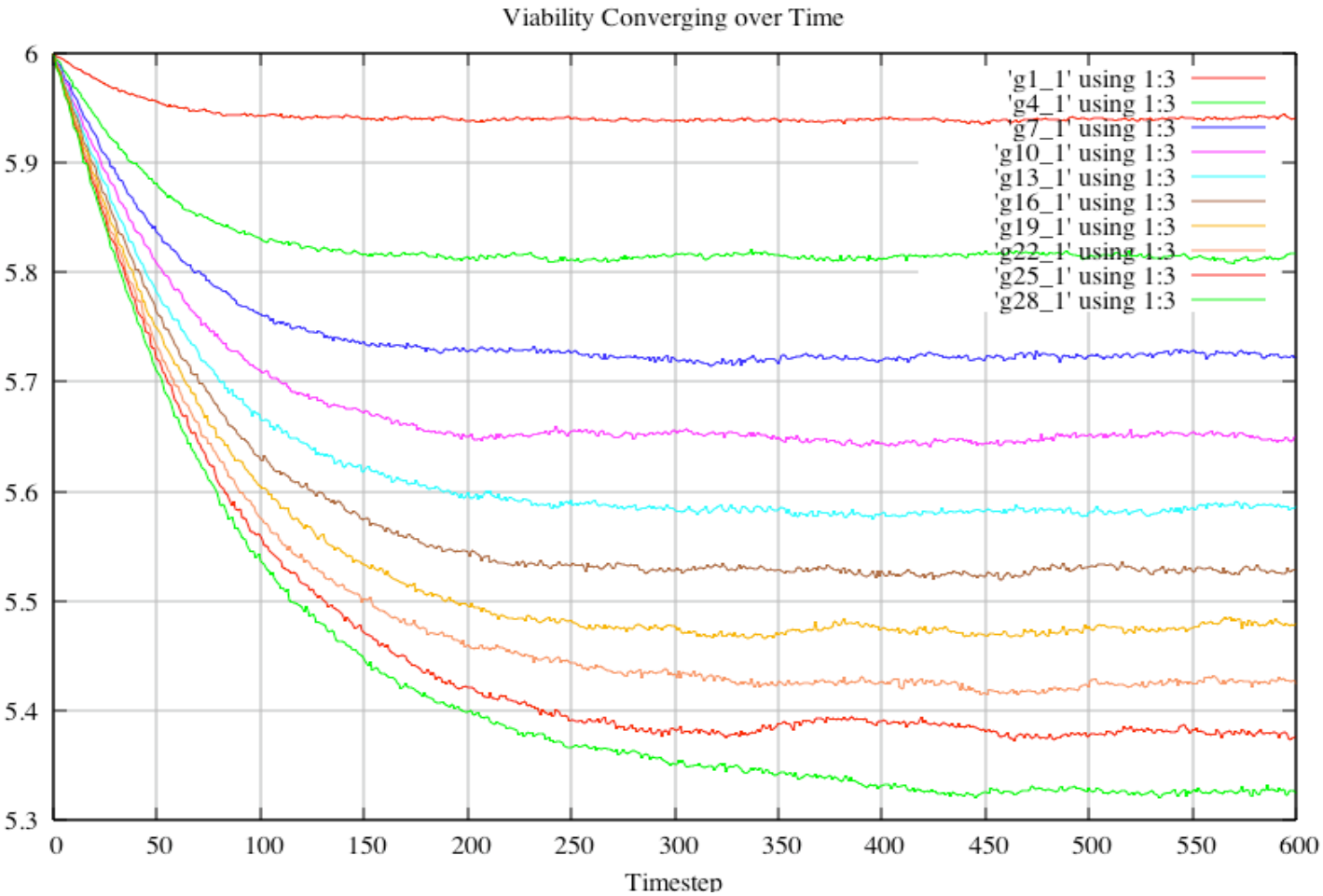
We had initially set out to discover the parameters of ε,f, and η that would result in viable populations. Now that we know that all-important properties of the histogram are a function of ℜ our task is much simpler. Here we attempted to sketch the "boundary of survival" across many values of ℜ and η. This was done by waiting until the histogram had reached its steady attractor state and then dividing the number of children by the number of parents. This illustrates the "branching ratio" or viability for a machine with specified parameters ℜ and η. Values ≥ 1 constitute viable populations, values < 1 constitute populations that will slowly shrink to nothing.

R=0.5	R=0.6	R=0.7	R=0.8	R=0.9	R=1.0	R=1.1	R=1.2	R=1.3	R=1.4	R=1.5	R=1.6	R=1.7	R=1.8	R=1.9	Childre
0.412	0.473	0.522	0.562	0.590	0.620	0.644	0.669	0.688	0.707	0.718	0.739	0.750	0.761	0.772	n=1.0
0.453	0.517	0.570	0.616	0.649	0.681	0.711	0.732	0.757	0.775	0.792	0.808	0.827	0.838	0.852	n=1.1
0.499	0.568	0.629	0.676	0.705	0.743	0.778	0.799	0.825	0.848	0.866	0.883	0.900	0.915	0.926	n=1.2
0.541	0.609	0.675	0.733	0.770	0.806	0.841	0.867	0.895	0.917	0.938	0.956	0.971	0.989	1.003	n=1.3
0.580	0.661	0.727	0.788	0.828	0.869	0.907	0.934	0.963	0.989	1.011	1.031	1.048	1.065	1.080	n=1.4
0.622	0.711	0.778	0.842	0.883	0.931	0.969	1.001	1.035	1.062	1.080	1.104	1.126	1.140	1.154	n=1.5
0.667	0.754	0.828	0.897	0.945	0.988	1.036	1.069	1.102	1.132	1.154	1.180	1.198	1.217	1.235	n=1.6
0.706	0.805	0.877	0.954	1.004	1.054	1.098	1.136	1.171	1.199	1.224	1.251	1.277	1.294	1.312	n=1.7
0.747	0.847	0.935	1.011	1.062	1.112	1.165	1.202	1.240	1.273	1.297	1.324	1.350	1.374	1.391	n=1.8

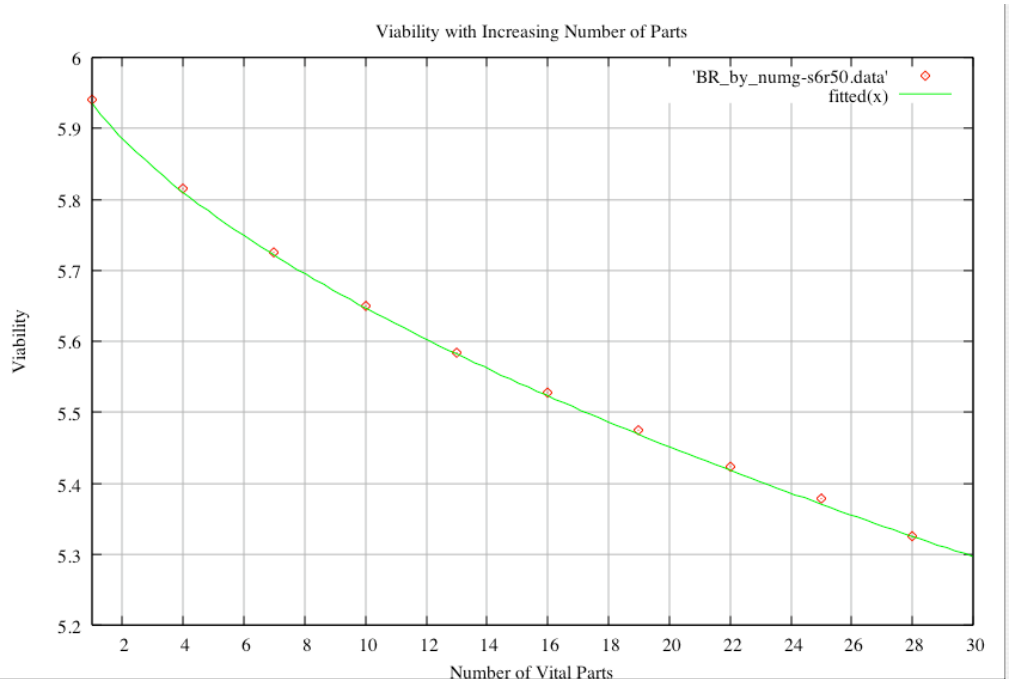
As we can see, the viability of a population follows regular monotonic patterns for both ℜ and η. To predict where any given machine will lie on the side of the boundry, we simply take the slope of η / ℜ. From this we determine that for a machine to be viable the value of η / ℜ for that machine must be ≥ 1.88.

3.3 Putting an Upper-bound on Complexity

We have briefly seen that by taking the minimum fitness across a number of parts the viability of a machine is decreased. Given our previously sculpted boundary of survival, we now ask ourselves that, given a value of ℜ and η, how many vital parts (how much complexity) can a machine have before it becomes unviable?



Here we have a graph showing viability converging over time with $\Re$=50, η=6 and number of vital parts (the dimensionality of x): 1, 4, 7, 10, 13, 16, 19, 22, 25, and 28. We see not only does the viability converge over time (reaffirming the findings from the converging gene histogram), but that with increasing number of vital parts, as expected, the viability decreases. To know the maximum number of vital parts such a machine could tolerate before it became unviable we must plot the converged viability as a function of the number of vital parts.



We do this across many plots like the one above and we find the viability (green line) consistently follows the form of:

$$\text{Viability} = Ae^{(Bx^C)}$$

$$x = \text{number of vital parts}$$

The functional form $Ae^{(Bx^C)}$ is constant across all values of $\Re$ and η, however the values of A,B,C vary with changing $\Re$ or η. With this functional form in hand, to find the x_{crit}, or the maximum number of parts the machine could handle before it is unviable we simply set the equation to 1 and solve for X. Then gives us:

$$x_{crit} = \left(\log_e(A)/B \right)^{1/C}$$

We can now solve for x_{crit} for any value of A, B, and C. All that remains to assess the maximum complexity of a viable machine is know A, B, and C as a function of $\Re$ or η. Plotting A, B, and C as a function of $\Re$ or η and then fitting a generalized function to the data we find:

$$A = \eta + 0.95e^{-0.15\,\Re}$$
$$B = -0.48e^{-0.17\,\Re}$$
$$C \approx 0.63$$

For example, a machine with $\Re = 100$ and $\eta = 6$ can support up to 5,443 vital parts (but no more), and remain viable.

3 Future Work

Although much has been done to assess the theoretical viability of self-reproducing machines, there is always more that could done to make our models more realistic. In particular, we assume that every part in the machine has the same "robustness", or f. However, in our everday experience with machines we know that some vital parts are less robust than others. How does the overall viability of the machine change when f is not constant but a distribution? Secondly, environments change. It would make sense to see the dynamics of both $\Pi(x)$ and viability with putting β on a random walk. Lastly, in the final application of our model to the real world we should consult the mechanical engineering literature to establish sensible realistic values for $\Re$ and η.