Michael Riedlin
Department of Physics
mvriedlin@ucdavis.edu
Michael Van Veen
Department of Computer Science
michael@mvanveen.net

Wealth of Automatons

Economics, the study of the management of goods, has been a topic of interest to man throughout time. In general, the economist is concerned with the most efficient distribution of resources. Adam Smith proposed in his Wealth of Nations, that the free market provides an "invisible hand" that drives a market towards its equilibrium and most efficient state. In this project we sought to test this notion with a multi-agent model, where a number of agents are given some initial conditions and then set loose upon an unsuspecting free market.

Introduction

The housing bubble bursts, creditors tank, and the global economy slumps. If the base assumption of economics is that individual entities universally act in their own self interest, one must wonder how something could go so catastrophically wrong. Our money is not backed by anything real, it's only worth what society thinks it's worth. That makes nonsense at a surface glance.

The above, while motivating interests and explainable, are regrettably outside the scope of this project. Given the scarce resources allocated for the project we sought the behavior of a simple two-good-market. This proved to be more difficult than we originally thought. The two good market looks at a very basic problem. People have guns and butter, but no one is satisfied with the guns and butter that they have. Some people want more of something, some people want less.

Background

The most important idea we need introduce is that of a production possibility frontier. A production possibility frontier is the set of all possible things an entities can produce. The rates in the simulation represent this.

Dynamical System

For our dynamical system, we chose to model a two good market. We set about this by first defining what an agent would look like and how it should behave. An agent knows seven things. It knows how much of each resource it has, how rapidly that resource is replenished or depleted and it knows how much of that resource it needs to survive. In addition to those we found it useful to determine whether or not the agent knew if was dead.

Agent behavior is admittedly very simple. The agents we designed for this project have no conception of "future", they only know how to be satisfied at the present. This lack of foresight also appear in an agent's choice to buy or sell something. Our agents will always sell their surplus over the threshold until it's gone or there is no buyer. Trade rates are decided by the individual agents based on their individual production possibility frontiers discussed above.

We then have a market. What a market does is generate X agents with characteristics taken from a Gaussian. The market handles the trades by comparing the trade values between the agents and allowing the ones that that satisfy each other to trade. The order in which that is handled is mostly random.

Methods

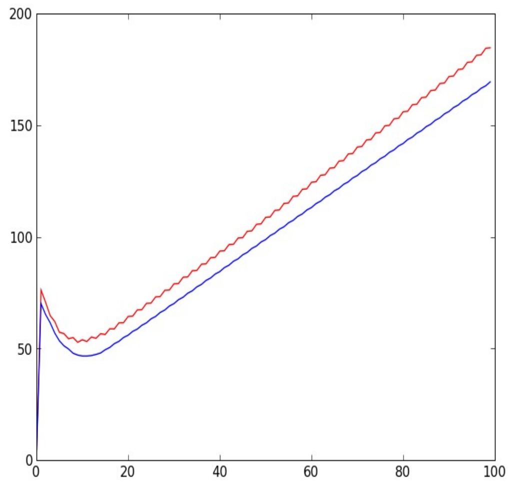
We ran our simulation over and over, looked at the graphs and tried to deduce what types of initial conditions lead to what trends. The thing that's been looked at most extensively is the average value of agents as time goes on. This was originally implemented as a simple sanity check but it was observed to exhibit some interesting behavior and that the average value could be seen as a measure of global wealth for the system.

Results

Initially, our simulation had the agent rates setup so as to perform multiplicative updates. This setup is gross and wrong for real world simulations for a couple of reasons. The first reason is that it means for rates less than zero, the agent values will be negative every other iteration. Another reason is that it's just too fast. For rates greater than one, you quickly reach a point where values jump by orders of magnitude each time step. It is interesting to note however that in this scenario, for rates between zero and one there is what appears to be an exponential decay until some critical point in the rate value where it would switch over to some flavor of exponential growth.

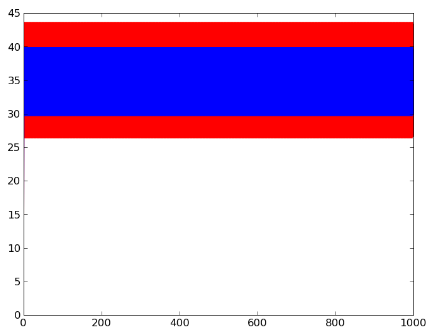
So, the rates are set to be additive instead. This didn't completely work out either but it did make more sense. For nonzero rates we observed exponential trends but they seemed more reasonable. Also, a negative rate meant a decay, a positive rate always meant a growth and zero rate meant nothing changed. There was a recurrent edge case that we could not appropriately address and that was a negative rate creating a negative sloping line for average values.

Now for something more interesting. First, we implemented a workaround to the ZeroDivisionError that had been barring high iterates by saying that agents would always at minimum have one resource of each type. Then we thought about threshold. As it was implemented, it didn't really mean anything. It was this arbitrary static amount that agents would surpass and not care about. Also, we changed from trading based on quantity, to trading based on production. What that means is we started zeroing the quantities at each iteration. Each agent now has its production rate minus its threshold to trade and meet its quota (still needs to meet the threshold to survive).

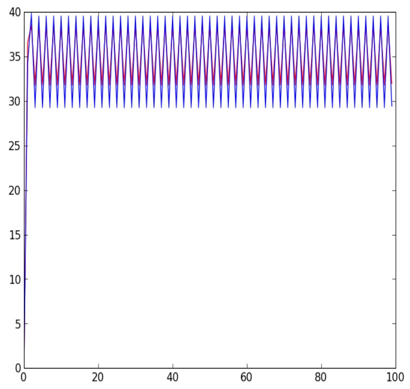


This is the revision 20 agent on with rates drawn from around 80 and thresholds drawn from around 10

To the left what you see is the latest version of the agents. Note the small oscillations in values that occur. Ultimately, rates turn out to be the dominant behavior in many types of settings. Below, I've placed two more graphs off the final version of agent.py. The 1000 iterate graph was initial a test of a workaround, but it produced this misleading graph. A zoom showed it to be just rapidly oscillating. The zoomed



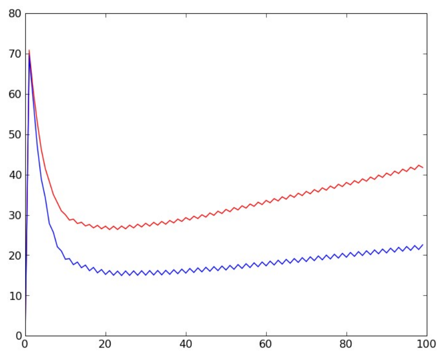
Test run on comparable rates and thresholds



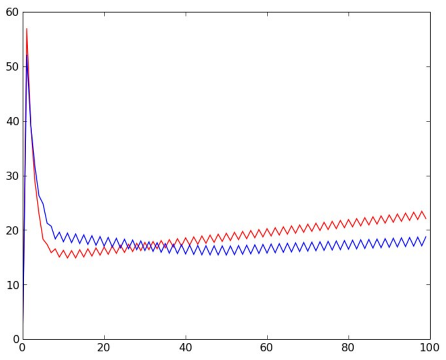
The same driver as the one at the left on a lower iteration run.

picture is another run on the same settings (50 agents, thresholds of 70, rates of 80, standard deviations 10). We ask that the reader remember that all agents are randomly generated, so there is always some variance in results, even with the same initial settings.

Lastly, I'd like to show two graphs that came off the drivers called coupdegrace.py and coupdegrace2.py. It turns out, that the graph of heavily oscillating behavior and the graph with the rise are extreme cases and that a range exists (thresholds ~ 15-20 tested)wherein you see a mix of both behaviors.



graph from coupdegrace.py



Graph off coupdegrace2.py

The coupdegrace2 graph is the most interesting behavior our model has yielded thus far.

Conclusion

Our feelings based on these results and discussion of our simulator, is that this is a good first stab at providing a model. Playing around with agent.py and varying the driver parameters led us to some interesting results and perhaps there are trends we haven't discovered yet. However, agent.py comes off too simple and in some cases wrong. For one, a real agent would seek to maximize its value or profits where as ours will only subsist and throw away it's excess. Also, it is unclear to us that the agents are dying properly. There is very convincing evidence that they are, but not all metrics are consistent.

A few avenues of exploration are immediately obvious to us. An expanded analysis tool and more thorough exploration of initial conditions would likely provide more information for the system. Introducing buyer preference into market.py has also been considered. As it is now, agents trade essentially at random but it might be more accurate to reality to say the top 20% go first. Monetizing the system would allow us to make more abstract decisions about trading and would allow us to remove the two-good restriction. We could also potentially modify the agents to have varied behaviors and a sense of history or success, like a genetic algorithm.