Money Talks...But All Mine Ever Says is Good-Bye. Using Monte Carlo Methods to Model Financial Transactions.

Victor Ramirez, Nat Hawkins

April 20, 2017

Abstract

Econophysics is a branch of Physics that has been around since the mid 1990's. It focuses on the use of concepts and practices from multiple areas of study, specifically Statistical Mechanics, and how they can be carried over to observe modern day economical phenomenon. Originally, Physics-based approaches were meant to be used as a secondary approach to provide a secondary explanation for economic occurrences, but soon after extended to observing and predicting new situations. The term was coined due to the large number of papers published by physicists written on topics such as stock market behaviors, wealth distribution modeling, and financial data analysis [1]. In this project, we created a basic model for financial transactions between two individuals which was ultimately dictated by random number generation in Python. This was then extended to include additional considerations such as savings, interactions with individuals who are close in wealth, and transactions that include some preference for previous interactions between agents. The latter processes are dictated by random number generation and the householder algorithm. We will display results from our analysis as well as draw comparisons to published literature and discuss avenues for future work in Econophysics based on our modeling in this project.

1 Introduction

1.1 Motivation

The allocation of wealth and how income is distributed in society are often used to debate modern economic issues. Most of the wealth tends to be concentrated, as Bernie Sanders would say whilst wagging his index finger, millionaires and billionaires while the rest of the population are left with little. Politicians and economists have been attempting to find ways to reduce the impoverished population and close the income disparity gap. Understanding how this phenomena occurs is an important step in possibly solving this issue.

This phenomena was notably studied by Vilfredo Pareto, an Italian economist and engineer who studied how a system of bodies reached equilibrium in the 1870's. Pareto would expand this study into economics and develop insights on how income distribution equilibrates. The result would become known as the Pareto Distribution, a power law probability distribution that has applications in many fields including physics. Namely, the Pareto Distribution can also be used to describe the sizes of meteors and the Bose-Einstein Codensate. The Pareto Distribution is also an integral building block to the growing field of Econophysics. [2]

1.2 Mathematics

One of the reasons we chose to pursue this avenue of study in this project is the computational simplicity of this type of modeling. The equations that dictate the behavior of the agents during their various transactions are relatively simple and can be programmed into Python in just a few lines of code. Despite the brevity of the code, the insights we developed were profound enough to dedicate an entire section to it.

The base for our modeling exercise is the modeling of Pareto's work [2]. From Paretos work (V. Pareto, 1897) it is known from empirical studies that the higher end of the distribution of money follows a distribution of wealth, w_m given as

$$w_m \propto m^{-1-\alpha}$$

where me denotes the amount of money in question and $\alpha \in [1, 2]$. This equations states that the distribution of wealth within a "society" should follow an exponential relationship.

In all of these financial transactions, two agents selected by index, i and j, have some amount of money prior to the transaction given as m_i and m_j respectively. We make the assumption that no agent can go in to debt ($m > 0$ for all transactions). This removes the possibility of negative values of wealth being introduced into society. While this is unrealistic, we can relate this to statistical mechanics concepts such as ideal gas behavior.

The transaction at a base level then involves the generation of a random number ϵ from 0 to 1, which is generated using the random.uniform library in Python. The amount of money that each agent has after the transaction is given as

$$\begin{aligned} m_i &\to m_i' \\ m_j &\to m_j' \\ m_i' &= \epsilon(m_i + m_j) \\ m_j' &= (1-\epsilon)(m_i + m_j) \end{aligned}$$

The random number generation predicts the percentage of the total money between the two agents each person ends up with after the transaction has occurred. In a typical sense, we would expect that one person is the "seller" and gets some amount of money and the other is the "buyer" who gives money to the seller. Here, we describe this interaction as a mutual exchange of goods/services, so instead of one person winning and the other losing, there is an exchange for mutual benefit. From this and the results from Patriarca [3], the final distribution of the wealth should be in the form of

$$w_m = \beta \exp\left(-\beta m\right)$$

When we included savings into the equation (pun intended), the only difference is that some amount of money is being withheld from the transaction. Now, instead of the total amount of money $m_+ m_j$ being exchanged is $(1-\lambda)(m_i + m_j)$. Now the amount of money that each individual has after the transaction has occurred is

$$\begin{aligned} m_i' &= \lambda m_i + \epsilon(1-\lambda)(m_i + m_j) \\ m_j' &= \lambda m_j + (1-\epsilon)(1-\lambda)(m_i + m_j) \end{aligned}$$

which can be written more conveniently in the form

$$\begin{aligned} m_i' &= m_i + \delta m \\ m_j' &= m_j - \delta m \\ \delta m &= (1-\lambda)(\epsilon m_j - (1-\epsilon) m_i) \end{aligned}$$

Here, ϵ is still a randomly generated uniform number, but λ is now a user set input denoting the amount of money that is being withheld during the transaction. Here, we expect the results to follow a normal (bell-curve) distribution as discovered by Patriarca et al. [3].

The two other cases we considered were more complex in meta terms, but computationally no different. The first of these two cases is the consideration of "nearest neighbor" interactions. Here we say that the probability for two agents to interact scales as the absolute value of the difference between wealths

$$p \propto |m_i - m_j|^{-\alpha}$$

where α is a parameter that we can toggle. This says that the closer two agents are in wealth, the more likely they are to have a financial event. This is then compared to another random number generated within the transaction loops. If the probability value is larger than the uniform random number we generate within the loop, then the transaction occurs using the same functionality as the basic transaction model where some amount, ϵ, of the total money is exchanged. This is in accordance with Householders algorithm. If the agents have the exact same wealth, then the probability of interaction is said to have occurred definitively (p=1). This can be amended to also model simulations that include savings, but the outcome is still the same as dictated by the literature as following $w_m \propto m^{-1-\alpha}$ [4]. The larger the value of α, the closer agents have to be in wealth for there to be a significant probability of transactions occurring.

The second of these interactions we accounted for previous interactions. The probability now scales as

$$p_{ij} \propto |m_i - m_j|^{-\alpha} \left(c_{ij} + 1\right)^{\gamma}$$

where c_{ij} denotes the number of previous interactions between agents i and j and γ is another user set parameter with a similar meaning to α in the previous case. The larger the value of γ, the more the probability is weighted by previous interactions.

The final product of these two cases will be a comparison to the famous "Pareto Tails" found by Goswami and Sen [4].

2 Equilibrium

In the literature, there is a lot of discussion about the equilibrium state or steady state distribution. This is the point where additional financial transactions do not yield a change in the distribution of the wealth of the society. Another way to think of this is that, like a statistical mechanical system, the society has reached its more probable distribution.

The literature discusses many ways of deciding when equilibrium is reached. These range from looking at when the standard deviation goes to a steady states value (net fluctuation about the mean value) [3] or looking at when the differences in the distributions goes to zero by comparing datasets [4]. We made an assumption that the number of transactions that we had the society performed was larger than the number of transactions required for the system to reach whatever equilibrium condition could have been set. This was for multiple reasons. For one, it was simpler to implement computationally (a while loop that terminates after some set number of transactions allowing for easy plotting), and also it reduced the likelihood of error in improperly determining an equilibrium condition. Since beyond equilibrium, the distributions do not change, then additional transactions beyond the equilibrium condition are assumed to have no affect on our results. It also reduces CPU time and load because it eliminates the need to check some calculated value against previous calculations through each iteration. The goal is always elegant and fast programming, so this measure was taken accordingly.

Therefore, we still are enacting an equilibrium condition: equilibrium is achieved in some number of transactions that we assume is less than the number of transactions we dictate the society conduct. Further work could be done in providing a more analytic expression in accordance to the literature.

3 Basic Transactions

The first step of this project is to model basic financial transactions made by agents. We assume the following:

- Each agent starts off with the same nonzero amount of wealth m_0
- The total wealth within the society is relatively conserved [1]
- No agent gets into debt, i.e. no negative wealths

We make the agents by creating an array of some user set size and assigning every cell in the array to some numerical value that is equal to the amount of money that that "person" (index) has. This way, we can keep track of the individual via the indices and the wealth based on the values of the cells.

In each iteration, a random number generator chooses two agents A_i and A_j assigned by indices i and j to trade. A_i then trades a randomly generated portion ϵ of its wealth to A_j. We repeat these iterations $10^6 - 10^7$ times and show that the wealth distribution follows the Pareto Distribution discussed in 1.2. Most of the wealth is concentrated among a few agents, while the most agents are left with a small amount of the wealth. This is consistent with real income distributions. A summary of some of these results are offered in Figure 1.

[1]See 6.1

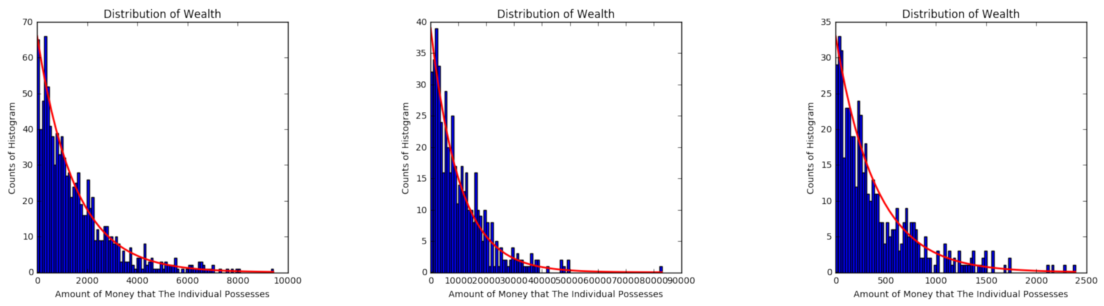
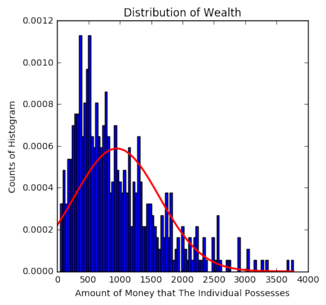


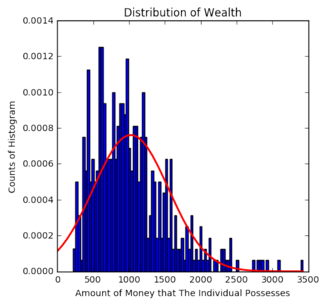
Figure 1: The wealth distribution of three trials. The bars are counts for each wealth "bin", while the fit is an exponential decay. These are for various amounts of agents with different amounts of starting wealth. The fit verifies that our model obeys the Pareto Efficiency Law.

4 Introduction of Savings

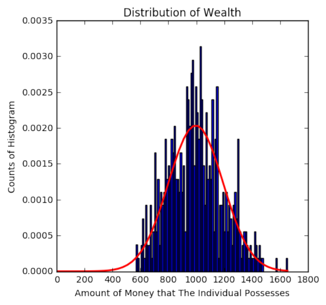
We expanded this model to introduce savings. We use a predetermined parameter λ that an agent saves before each transaction, described in 1.2. Our result for the basic model was made with $\lambda = 0$. We ran trials for $\lambda = 0.25$, 0.5, 0.9 at $N = 500$, 1000 agents. The trials for $N = 500$ and 1000 were nearly identical; the main difference was that higher N values take longer to equilibrate and form a sharper normal distribution for higher savings values. Because the transaction value was fixed, our final result for $N = 500$ is closer to equilibrium. The trials for changing λ show that the resulting income distribution shifts from a Pareto Distribution to more of a Gaussian Distribution as λ is increased. This agrees with the results of Patriarca et al. [3]. The summary of results for our modeling is shown in Figure 2.



(a) $\lambda = 0.25$, N = 500



(b) $\lambda = 0.5$, N = 500



(c) $\lambda = 0.9$, N = 500

Figure 2: The income distributions of varying λ at N = 500. The income distribution resembles more of a Gaussian distribution at higher λ.

5 More Complex Problems

5.1 Nearest Neighbor

The basic framework of the nearest neighbor interaction is outlined in Section 1.2.

For this section, we had to redefine how we initialize agents. Typically, we would make an array with all of the same values (everyone starts with the same amount of money). In this scenario, if everyone started with the same amount of money, then the probability for interactions would always be equal to 1 (since we defined that in instances of exactly the same wealth transactions occur definitively), and nothing interesting would occur. So, for these advanced trials, we ran this using a uniform distribution to give everyone some amount of money within a given range.

We first begin by looking at the α=0 case. Here, we say that the probability of transactions occurring is equal to 1, so the transactions occur definitively as discussed earlier.

The trial for our $\alpha = 0$ case is shown in Figure 3. For the case where α=0, we get that the probability for interactions is 1, which means that all agents engage in transactions. We ran this trial for only 10^5 transactions due to computational load, so we do not see the same sort of peak in the exponential decay of the plot. We do still see that along the lines of this unit test, we still get the earlier produced results where our distributions follow the exponential $w_m \propto m^{-1-\alpha}$.

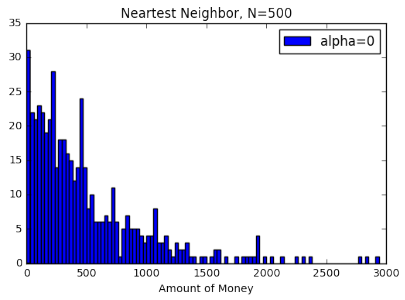


Figure 3: Nearest neighbor case for α=0. We can see that we still get the expected exponential distribution of wealth that we got in 3. This served as a unit test to ensure that our program would reproduce previously achieved results given the proper inputs.

Now, if we run this with $\alpha = 0.5, 1.0, 1.5, 2.0$, we get similar distributions for small α. As we increase the value of α, the likelihood of transactions begins to get smaller and smaller. This means that we gradually lose our exponential distribution and it becomes slightly more "flat". This is because only people who are extremely close in wealth have any significant probability of interacting. This is emphasized in Figure 4. The distributions are shown in figure 5 both with and without savings. The distributions looks somewhat different in terms of scaling, but overall we see the same general trend.

These results follow our expectations of the literature [4]. The more impressive results come with the recreation of the Pareto tails. See Section 5.2. From our wealth distribution data, we were also able to recreate the log plots seen in [4].

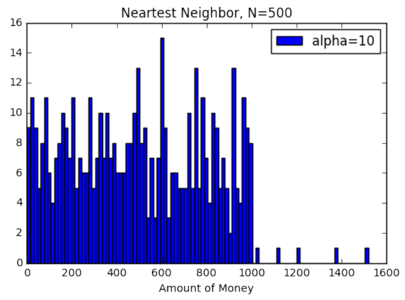
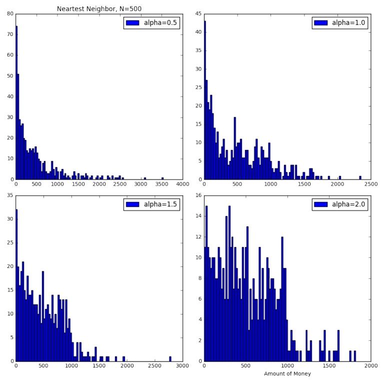
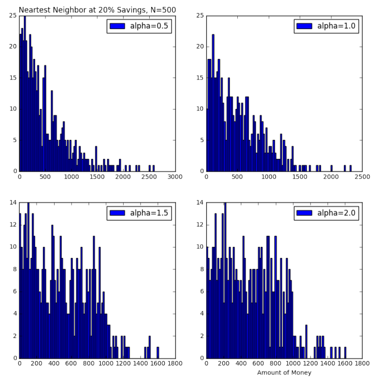


Figure 4: The nearest neighbor interaction with α=10. At large values of α, the distribution of wealth continues to become "flatter" and closer to the initial distribution of wealth due to the low probability of interactions for people of different wealths.



(a) Subplots for the nearest neighbor interactions as dictated by our algorithm. The values of alpha are listed on the plots. This is for N=500 and no savings.



(b) Subplots for the nearest neighbor interactions as dictated by our algorithm. The values of alpha are listed on the plots. This is for N=500 at 20% savings.

Figure 5: The above plots show the nearest neighbor wealth distribution for N=500 with and without savings.

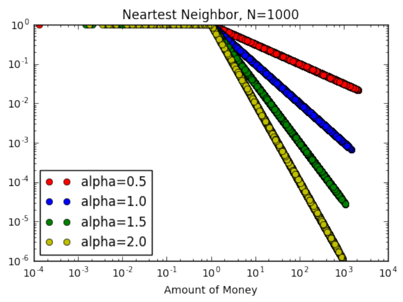
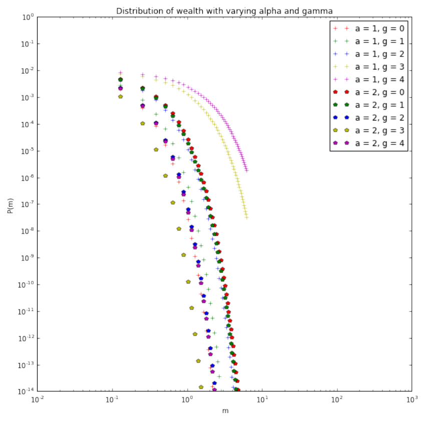


Figure 6: Recreation of log plots from Goswami and Sen. Verifies exponential distributions of wealth in society as dictated by $w_m \propto m^{-1-\alpha}$.

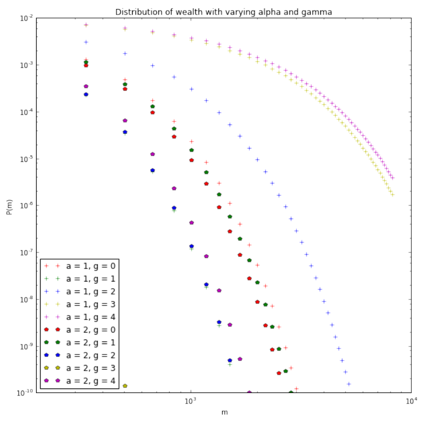
5.2 Previous Interactions with Nearest Neighbors

The last expansion to this program is to account for previous transactions made by agents. To model more realistic transactions, we added an additional parameter that makes agents more likely to trade with those similar in wealth and have a history of transactions together. We introduce the parameter γ which dictates how strongly these factors affect the likelihood of two agents trading. The higher the γ, the more an agent "values" past transactions and vicinity and wealth, making it more likely to trade with another agent that shares those attributes. The probability to make a transaction is discussed in the last section of 1.2. An analogous model was created by Goswami and Sen [4] and we used their results as a benchmark for our results.

We created a log-log plot the probability P(m) as a function of the money m. We saw that our results matched that of the literature [4]. In particular, the "tails" of the distribution matched the expected behavior: The higher the γ, the sharper the drop in P(m) as a function of m. In other words, the income disparity becomes more pronounced as γ is increased. These plots are shown in Figure 7.



(a) Probability P(m) as a function of the money m for the nearest neighbor and past transactions model.



(b) A zoomed in picture of the Pareto Distribution

Figure 7: The Pareto Distributions of our nearest neighbor and past transactions model. These behaviors match that of Goswami and Sen[4].

6 Discussion

6.1 Conservation of Money Numerical Precision

Based on how we defined the nature of these transactions, the total amount of money in the system should be conserved since proportions of the total money are being exchanged in each transaction.

We ran a few scripts to determine if this was true and the results were troubling. Figure 8 shows that for the same simulation, the total money in the system is not being conserved and in different ways for the different simulations. The causes for this stem from our previous issues with numerical precision. Even with rounding all of our money values to 2 decimal places, rounding the random numbers to 2 places, and being careful to truncate our decimals at the correct values, Python has a systematic error with random number generation and lack of numerical precision. As a result, some randomly generated numbers can lead to jumps in the total amount of money in the system. Granted, they are negligible, but still present. The small magnitude of the variation allows us to assume that money is conserved throughout a simulation. Future work could be done to alleviate the issue of loss of numerical precision within our model in order to preserve "true" conservation.

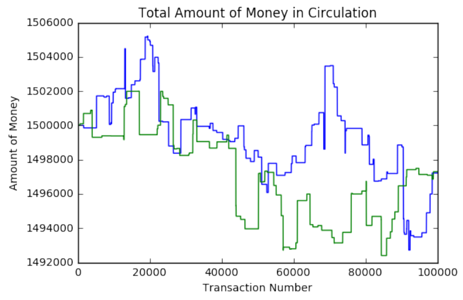


Figure 8: Plot of total money in a society. The same instance was modeled (N=1500, starting money=$ 1000, transactions=10^7), and the blue and green lines show the respective results for total money in the system versus transaction number.

6.2 Future Work

Our simple model can be greatly expanded to model the messier and more complex real world.

A huge component would be implementing a system of taxation. We can do a simple sales tax model by taking away a percentage of money exchanged during each transaction and transferring it to a 3rd party that acts as our government. Another way of ~~stealing~~ taking away this money would be to take a percentage of money that each agent already has after a set amount of iterations. This government can subsequently redistribute this money among the population. One would predict that the income distribution would be more even assuming the society doesn't collapse.

Another additional implementation is to remove the assumption that wealth is conserved. Real world economies prosper or collapse, and an individual's wealth changes based on this. Accounting for this would make for a more realistic macroscopic simulation.

We could also include some mechanism where when individuals save money, they can earn interest on that money up until their next transaction. This would stem from the lack of conservation of money because we would be introducing new money into the system. This is more along the lines of real world situations and would be interesting to model.

Regardless of our decisions for future work, more sophisticated computational machines would be needed to handle the amount of floating point operations required for thorough analysis such as credit cards, taxation, saving interest, debt, etc. since our feeble laptop computers operate in a range where these types of models for large scale simulations could take hours. Improved equipment and implementation of more efficient languages (C++) could allow us to investigate more complex simulations and perhaps conduct new research into Econophysics.

7 Conclusions

We were able to use Monte Carlo methods of random number generation to simulate financial transactions for base scenarios including exchange of goods and services and savings. Using Householder's algorithm, we were also able to use random number generation to compare the likelihood of events for probability dependent transactions such as the nearest neighbor and previous interaction scenarios. Our results match the results made by the literature and we provided some possible considerations for future work to be done in Econophysics with similar modeling techniques.

Supplementary figures can be found with all working codes in our Github repository [6].

References

[1] Econophysics. Wikipedia. `https://en.wikipedia.org/wiki/Econophysics`.

[2] Pareto, Vilfredo. *Course on Political Economy.* `http://www.institutcoppet.org/2012/05/08/cours-deconomie-politique-1896-de-vilfredo-pareto`. 1896.

[3] Patriarca, Marco, et al. *Gibbs versus non-Gibbs distributions in money dynamics.* `http://www.sciencedirect.com/science/article/pii/S0378437104004327`. Physica A. Volume 340, Issues 13, 1 September 2004, Pages 334339.

[4] Goswami, Sanchari. Sen, Parongama. *Agent based models for wealth distribution with preference in interaction.* `http://www.sciencedirect.com/science/article/pii/S0378437114006967`. Physica A. Volume 415, 1 December 2014, Pages 514524.

[5] Hjorth-Jensen, Morten. Github for PHY480. `https://github.com/CompPhysics/ComputationalPhysicsMSU`.

[6] Github Repository for All Files. `https://github.com/nathawkins/PHY480MSU/tree/master/project4`.