MONTE CARLO SIMULATION OF MONETARY TRANSACTIONS: A SIMPLE MODEL FOR WEALTH DISTRIBUTION

FYS3150: COMPUTATIONAL PHYSICS

TRYGVE LEITHE SVALHEIM
SEBASTIAN G. WINTHER-LARSEN
GITHUB.COM/GREGWINTHER

Abstract. We build a simple agent-interaction model in order to simulate wealth distribution by taking inspiration from the field of econophysics. The model can account for effects of saving (λ) and agent preference to trade only with agents of similar wealth (α) and agents they have dealt with before (γ). We find that the upper tails of the wealth distributions can only be adequately explained by simple Pareto distribution: $w_m \propto m^{-1-\lambda-\alpha-\gamma}$. We suggest some prospective future additions to the model; inclusion of time propagation dynamics, a system of several economies connected through varying degrees of trade, and functionality that takes welfare and social benefits into account.

Contents

Date: December 9, 2016.

1. Introduction

This is a first venture into the field of econophysics for us. We use concepts from statistical mechanics to build a model consisting of agents allowed to conduct monetary transactions with one another. This study is based on the previous work of Patriarca et al. (2004)[1] and Goswami and Sen (2014)[2]. We start by giving a very brief introduction to the simplest of macroeconomic models, justifying the use of a simplification to an agent-interaction model. Thereafter we show how the agent model works. It is built up piece by piece and takes into account the effects of saving, and the preference for an agent to conduct a transaction with another agent of similar wealth and one the agent has dealt with before. We then give a thorough description of how the model can be implemented using Monte Carlo methods in C++ [1]. The program is relatively straightforward to implement, except perhaps for the algorithm that breaks the transaction loop if a steady state is reached prematurely. This functionality speeds up the simulations tremendously. After that we swiftly present all results, accounting for all the different effects. What follows is a somewhat lengthy discussion of the model's shortcomings and suggestions for future developments. We end this article with some summary remarks.

2. Theoretical Background

2.1. The simplest model for an economy. Arguably, the most famous equation in macroeconomics is the "autarky identity"[2]

$$Y = C + G + I, \tag{1}$$

where Y is income, C is consumption, G is government spending and I is investment. The best-known, but not necessarily the best, measure of income Y is the Gross Domestic Product (GDP)[3]. Consumption C is the monetary value of all goods and services purchased in the private sector, while government spending G is the consumption of the government. Finally, investment I is the sum of private and public saving.

Equation 1 is autarkic because all interactions with other economies are excluded from the expression. There are no terms representing exports and captial inflow, for instance. We are dealing with a *closed* economy, alternatively the entire world as a whole. Moreover, let's assume that the economy we are studying is a peaceful anarchy, without a governing authority of any sort, in effect setting $G = 0$. To begin with, we will also forgo the agents the ability to save such that the worth of every individual, or agent, in the economy must be spent at once. Equation 1 is reduced to $Y = C$, everything one agent spends is the income of another.

2.2. Monetary transactions. To simulate monetary transactions in our model economy we employ the framework introduced in Patriarca et al.[1]. We assume there are N agents that exchange money in pairs (i, j). We assume also that all agents start with the same amount of money $m_0 > 0$. For every period an arbitrary pair of agents are picked at randomly and we let them conduct business, i.e. a transaction takes place between them. Money is conserved during the transaction such that

$$m'_i + m'_j = m_i + m_j, \tag{2}$$

where the left-hand side is the agent i and j's updated wealth and the right-hand side represents the amount of money agents i and j had before the transaction.

[1]All program source code is available online. Follow the github url on the title page
[2]Any introductory text on macroeconomics will give a thorough elaboration, e.g. Gärtner.
[3]GDP = GNP (Gross National Product) in autarky.

The exchange is done via a random reassignment factor ϵ, such that

$$m_i' = \epsilon(m_i + m_j) \tag{3}$$

$$m_j' = (1 - \epsilon)(m_i + m_j) \tag{4}$$

No agent will ever have negative wealth, that is $m \geq 0$. Moreover, because of the conservation law in equation 2, the system eventually reaches an equilibrium state given by a Gibbs distribution

$$w_m = \beta e^{-\beta m}, \quad \beta = \frac{1}{\langle m \rangle}, \tag{5}$$

where $\langle m \rangle$ is the expected wealth, for which the arithmetic mean is an unbiased estimator. This implies that after an equilibrium has been reached the majority of agents is left with lower wealth than they had initially and the number of rich agents exponentially decrease.

It is easy to see that the logarithm of the Gibbs measure in equation 5 yields a linear equation

$$\ln(w_m) = \ln(\beta) - \beta m \tag{6}$$

2.3. **Pareto distributions.** How the income and wealth of a population is distributed has been studied for a long time. Already in 1897, Vilfredo Pareto, sought to describe both wealth and income distribution by power law probability distributions[3]. Pareto showed that the higher end of the distribution of money follows a distribution

$$w_m \propto m^{-1-\nu}, \quad \nu \in [1, 2]. \tag{7}$$

Pareto's "80-20 rule", that 20% of the population controls 80% of the wealth, corresponds to a certain value of ν. Today, a whole class of distributions similar to the one in equation 7 are named after Pareto[4].

2.4. **Transactions and savings.** We are can expand upon the model by introducing a savings rate λ. The savings rate is defined as a fraction of an agent's wealth that does not partake in a transaction for each cycle. One can gather from the macroeconomic identity in equation 1 that income must still be the same and the transaction conservation law in equation 2 still holds. The updated wealth of agents i and j after a transaction becomes

$$m_i' = \lambda m_i + \epsilon(1 - \lambda)(m_i + m_j) \tag{8}$$

$$m_j' = \lambda m_j + (1 - \epsilon)(1 - \lambda)(m_i + m_j) \tag{9}$$

one can rewrite these expressions to

$$m_i' = m_i + \delta m$$
$$m_j' = m_j - \delta m$$

where

$$\delta m = (1 - \lambda)(\epsilon m_j - (1 - \epsilon)m_i) \tag{10}$$

2.5. **Economic inequality and social friction.** "The first historical series of income distribution statistics became available with the publication in 1953 of Kuznet's monumental Shares of Upper Income Groups in Income and Savings. Kuznet's series dealt with only one country (the United Stetes) over a period of thirty-five years (1913-1948)" (Piketty & Goldhammer, 2014, Kindle locations 276-280[5]). Ever since that time there has been wealth and income inequality to a greater or lesser degree. According to the PolitiFact the top 400 richest Americans in 2011 "[had] more wealth than half of all Americans combined"[6].

One of the possible reasons for continued wealth and income inequality that has risen in popularity the last few years is the process of wealth concentration. This is a process by which, under certain conditions, newly created wealth concentrates in the possession of already-wealthy individuals or entities. Those who already have wealth have the means to invest in newly created sources and structures of wealth. Piketty argues that the fundamental force for wealth divergence is the usually greater return of capital than economic growth (Piketty & Ganser, 2014 p. 284 Table 12.2[5]).

To incorporate the social rigidities described above, we will make two further additions to the model. These additions build mostly on the work of Goswami and Sen[2]. Firstly, we will make it more likely for two agents to conduct a transaction if they have similar wealth. Second, we will make it more likely for two agents to make a transaction if they have made a transaction before. We define a probability

$$p_{ij} \propto |m_i - m_j|^{-\alpha}(c_{ij}+1)^{\gamma}, \tag{11}$$

where the first factor $|m_i - m_j|^{-\alpha}$ relates to wealth similarity and the second factor $(c_{ij}+1)^{\gamma}$ relates to previous transactions. A relatively large difference between m_i and m_j should translate to a lower probability. The strength of this effect is governed by the exponent α, a larger value decreases the probability of a trade further. The variable c_{ij} is simply the number of times agent i and j have conducted transactions in the past. The number 1 is added in order to ensure that if they have not interacted earlier they can still interact. The strength of this effect is governed by the exponent γ.

3. Algorithm & Implementation

All programming code used in this project is written in C++ and is publicly available at github by following the link on the title page. The program consists mainly of two files, main.cpp and functions.cpp. main.cpp contains the set-up of the program and is where the number of agents, initial wealth, number of transactions, number of simulations and all parameters (like λ, α and γ) are specified. The main part of the program however, is in functions.cpp which in addition to the trade (...) [4] function contains the function output (...) [5] which writes data to file.

3.1. The trade (...) function. This function runs a specified number of Monte Carlo cycles, where the actual transactions take place. For each iteration two random agents are picked, then some logic follows which will determine if the two agents are allowed to go into business. This is a very important part of the program and is therefore listed here:

```
while ((pow(fabs(m_i - m_j), -alpha)*
    pow(previous_transactions+1, gamma) < random_number)
    || (agent_i == agent_j)) {

  // Pick new agents ...
  agent_i = (int) rand() % N;
  agent_j = (int) rand() % N;

  // ... find wealth of these ...
  m_i = agents(agent_i);
```

[4] The functions have several arguments: trade(int N, int no_of_transactions, arma::vec (& agents), double lambda, double alpha, double gamma). To include all the arguments in the text would have decreased readability and they are replaced with dots.

[5] output (...) = output(int N, arma::vec agents, std :: string filename).

```
  m_j = agents(agent_j);

  // ... update previous transactions ...
  previous_transactions = C(agent_i, agent_j);

  // ... and update the random comparison number
  random_number = (double) rand() / RAND_MAX;
```

One can see that the the argument of the while loop in the program listing above contains equation 11 which is compared against a random number and an equality test that will return True if agents i and j are the same agent. This part of the code makes sure that a new pair of agents are randomly picked until we are satisfied that the agents are not the same and that they are close enough in wealth and have good enough relations to go forward with a transaction. After everything is all-right and we have a good pair of agents a transaction takes place:

```
// Random value of transaction b/w 0 and 1
double epsilon = (double) rand() / RAND_MAX;

// Transaction takes place, but some is saved (lambda)
double delta_m = (1 - lambda)*
  (epsilon*m_j - (1 - epsilon)*m_i);
agents(agent_i) += delta_m;
agents(agent_j) -= delta_m;

// Register that a transaction has taken place
C(agent_i, agent_j) += 1;
C(agent_j, agent_i) += 1;
```

The program listing above should be straight-forward. Notice that the transaction is registered in matrix C by incrementing element i, j and j, i by 1.

The last part of the trade (...) function is not particularly important for the quality of the data that a simulation will provide, but vastly increase the speed of the program. This last functionality breaks the for loop within the trade (...) function when a steady state is reached. Listed in its entirety here:

```
double var = arma::var(agents);
cumVarBlock += var;
cumVar2Block += var*var;

// Enter here at the correct place: end of block
if (i % blockSize == 0) {
  // Variance of a block "averaged" over the block
  double avgVar  =  cumVarBlock / blockSize;
  double avgVar2 = cumVar2Block / blockSize;

  // Variance of a block of variance
  double varVarBlock = avgVar2 - avgVar*avgVar;

  // Check if variance is low enough
  if ((fabs(avgVarBlockOld - avgVar) / fabs(avgVarBlockOld)
          < 0.2) &&
    (fabs(varVarBlock - varVarBlockOld) / fabs(
        varVarBlockOld) < 0.5)) {
```

```
    // Telling a user that an equilibrium has been reached
    std::cout << "Equilibrium reached at transaction no. "
       << i << std::endl;

    // Done! For now.
    break;

    } else {
      avgVarBlockOld = avgVar;
      varVarBlockOld = varVarBlock;
    }
  // Reset cumulative variance
  cumVarBlock  = 0;
  cumVar2Block = 0;
```

This code section does some calculations on the results after a certain amount of transactions have taken place, an amount predetermined by blockSize. The average variance of the block is computed as well as the average square variance of the block size. Then the "variance of the variance" is computed. These measures are compared against the measures from the previous block. The relative size of the average variance must be less than 20% and the relative size of the variance of the variance must be less than 50% of the measures from the previous block. If both these logical statements evaluate to True, the for loop of the trade (...) function is broken, terminating the current simulation. We found that for most simulations an equilibrium is reached quite quickly, usually when between 40000 and 120000 transactions have taken place between $N = 1000$ agents. This speeds up the time for a simulation to finish tremendously, compared with the initially, and rather naively, picked 10^7 iterations. In order to find correct values for blockSize as well as when the variance measures are "low enough" the behaviour of the variance measures were meticulously studied through several trial runs of the function.

3.2. **Example main().** The parameters of the simulation is set up in the main() function. When all parameters are chosen the following for loop is almost everything that is needed

```
// Start simulations
for (int i = 0; i < simulations; i++) {

  // Print progress
  std::cout << "Simulation no. " << i << std::endl;

  // Assign intitial wealth to all agents
  agents.fill(m0);

  // Trade!
  trade(N, transactions, agents, lambda, alpha, gamma);

  // Sort and add equilibrium wealth to total
  totagents += arma::sort(agents);
```

One can see that what must be specified is the number of simulations (simulations), initial wealth (m0), number of agents (N), number of transactions (transactions), an array containing the wealth of every agent (agents), savings rate (lambda),

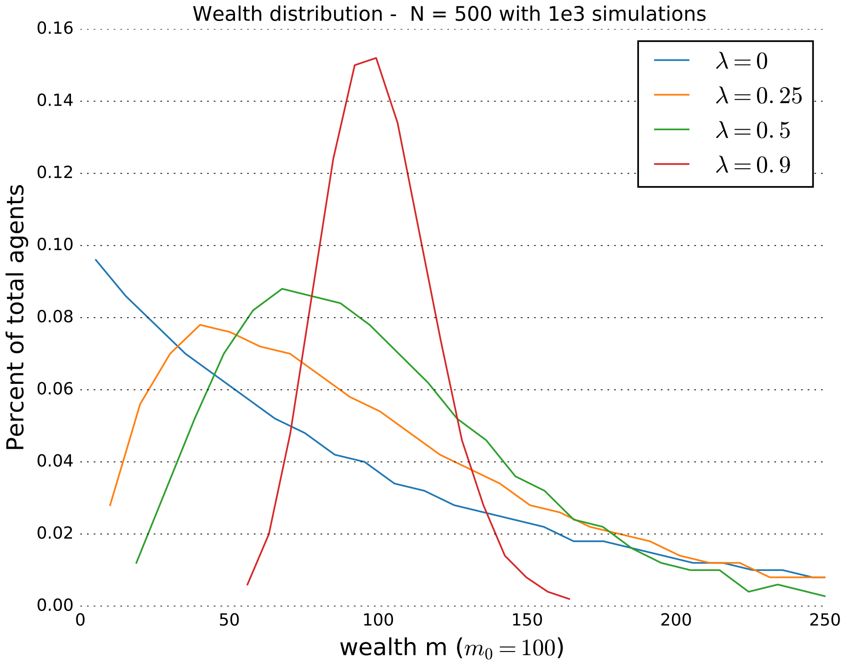


FIGURE 1. Wealth distribution for $N = 500$ agents after 1000 simulations at different savings rates $\lambda \in \{0, 0.25, 0.5, 0.9\}$.

parameter governing weight of wealth similarity (alpha) and parameter governing weight of past transactions (gamma).

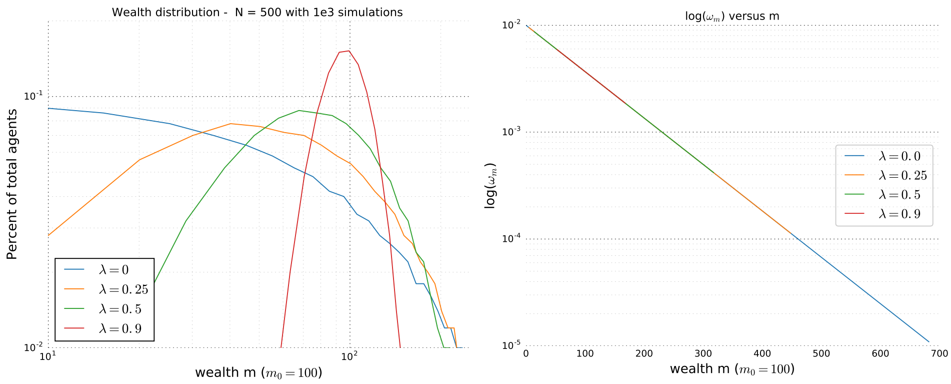
4. RESULTS

Figure 1 shows plot of different wealth distributions after 1000 simulations for $N = 500$ agents. We see that that without savings, the distribution shows a decreasing number of agents as wealth increases. As the savings rate λ is allowed to increase, the distribution moves to the right and eventually peaks at the initial wealth $m_0 = 100$ when $\lambda = 0.9$. This plot, were it not for the colors, could be an exact copy of figure 1 in Patriarca et al.[1]. Without savings the distribution appears as a inverse exponential function which is the same class of function as the Gibbs distribution (equation 5), and the Pareto distribution (equation 7). However, these traits disappear from the wealth distributions once savings are allowed.

A further study of the wealth distributions from figure 1 is visualized in figure 2. Figure 2a shows the same graphs as in figure 1, but on with logarithmic scales on both axes. Figure 2b the wealth values evaluated by equation 5 and plotted with logarithmic y-axis. Observe that all all the wealth distributions give straight lines. This makes sense, as the logarithm of the Gibbs distribution yields a first degree polynomial (equation 6).

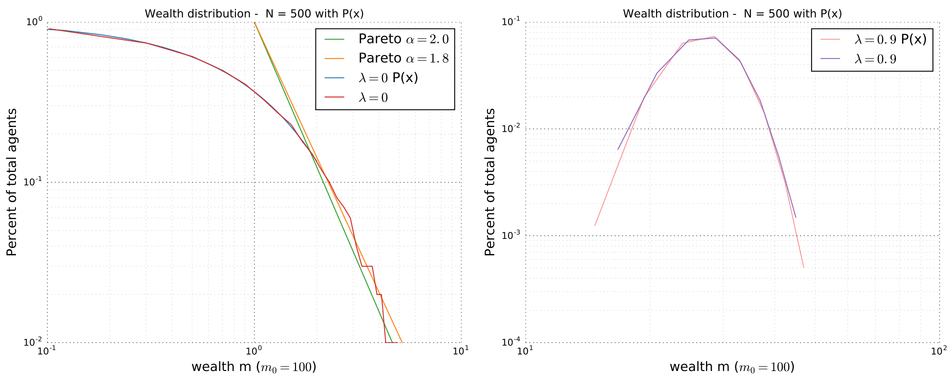
Figure 3a shows two different Pareto distributions with $\nu = 1.8$ and $\nu = 2$ (equation 7) in the same plot as a simulated wealth distribution where $\lambda = 0$ (no savings). We see that they fit adequately with the higher end of the distribution, but not perfectly. The simple Pareto distribution becomes a straight line because of the logarithmic axes in the plot. The same figure also contains a parametrization of the wealth distribution [6]. Figure 3b also shows such a parametrization, but for

[6] This is done in the way Partriarca et al. (2004)[1] does. A recipe is shown in appendix A



(A) (B)

FIGURE 2. Logarithmic plot of wealth distribution from figure 1 is shown in figure 2a. Gibbs measure (equation 5) of every data point from the simulations plotted with logarithmic y-axis in figure 2b.



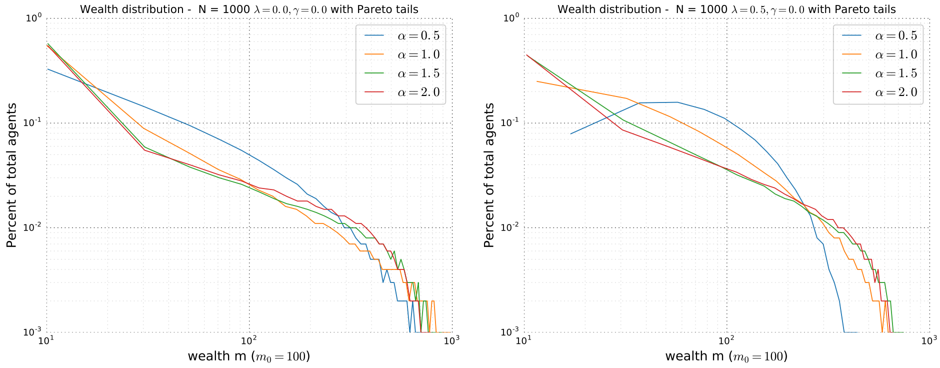
(A) (B)

FIGURE 3. Parametrization of two different wealth distributions, $\lambda = 0$ in figure 3a and $\lambda = 0.9$ in figure 3b. Method for parametrization is found in appendix A. For $\lambda = 0$ two Pareto power law fits are also shown in figure 3a.

a higher savings rate $\lambda = 0.9$. The parametrisations are indicated by a label $P(x)$. We see that the parametrisation provides a very good fit. The Pareto distribution only works for the right tail of the distribution and only to a certain extent.

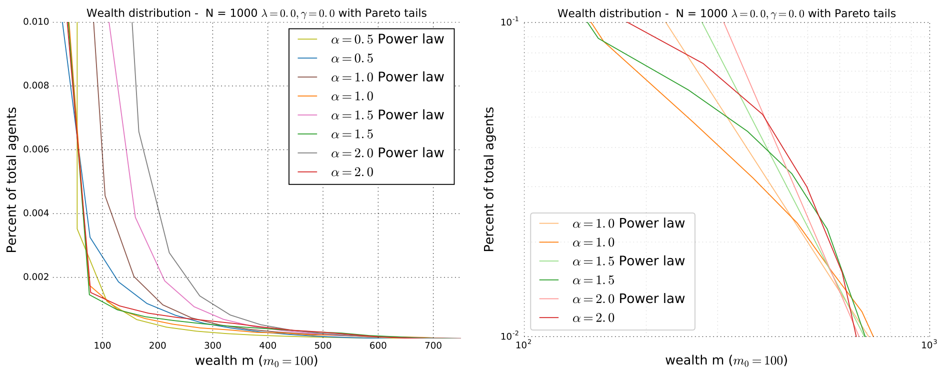
The effect of "nearest neighbour" interactions are summarized in figures 4 and 5. Figure 4 shows logarithmic plots of simulations for different values of α from equation 11. The parameter for previous transactions γ is set to zero for now. The only difference between the subfigures in figure 4 is that in figure 4a $\lambda = 0$ and in figure 4b $\lambda = 0.5$, colloquially with and without saving respectively. Figure 4 shows similar results to figure 5 in Goswarmi and Sen (2014)[2].

Figure 5 shows the simulations for nearest neighbour interaction fitted to a Pareto distribution. The Pareto distribution should fit only for higher values of wealth, i.e. the tails of the distribution. Figure 5a show that the power law fittings (Pareto distributions) appear to have similar shapes to the results from the simulations. One can see, however, that the data from the simulations tend to drop quicker



(A) (B)

FIGURE 4. The effect of different propensity to trade with nearest neighbours, $\alpha \in \{0.5, 1.0, 1.5, 2.0\}$. Figure 4a excludes saving, $\lambda = 0$, figure 4b includes saving, $\lambda = 0.5$.



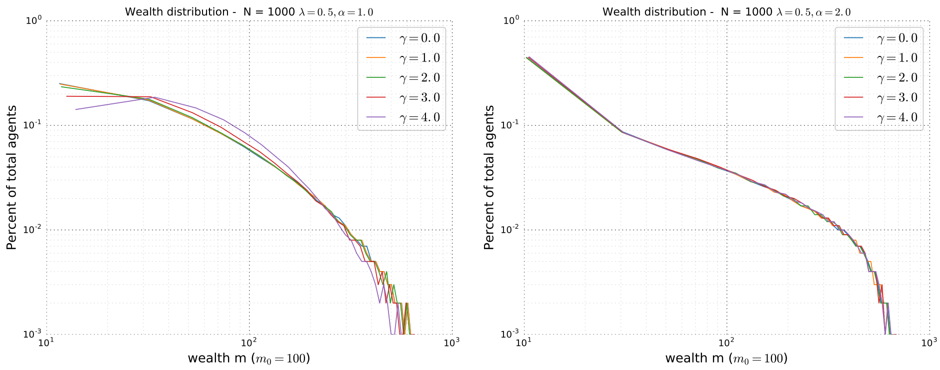
(A) (B)

FIGURE 5. Pareto power law fits of wealth distributions including propensity to trade with nearest neighbour, $\alpha \in \{0.5, 1.0, 1.5, 2.0\}$. Figure 5a is a linear plot, while figure 5b is double-logarithmic.

than their Pareto power law counterparts. Figure 5b shows a logarithmic version of figure 5a. Here the Pareto fittings are straight lines, as one would assume, but the data from the simulations apparently inhibit more complexity than the a simple power law relationship between wealth and frequency, but the fit is less horrible than it could have been.

The last effect to consider is the former transaction effect, in combination with savings and nearest neighbour interactions. This effect is illustrated in figure 6. This figure is an attempt to replicate figure 5 in Goswami and Sen (2014)[2]. Both subfigures show the wealth distribution for different values of γ (propensity to trade with agents you have traded with before), but figure 6a weighs neighbour interactions less strongly than figure 6b, with $\alpha = 1.0$ and $\alpha = 2.0$ respectively. Again, the Pareto distribution fit is okay, but not tremendously good. One can see that it fits best for the very tip of the tail.

Lastly, figure 7 show a power law fitting of the wealth distributions first presented in figure 6. The figure both includes ($\lambda = 0.5$) and excludes savings ($\lambda = 0$). The



(A) (B)

FIGURE 6. Wealth distributions when weighing of previous transactions has been taken into account, $\gamma \in \{0.0, 1.0, 2.0, 3.0, 4.0\}$. Figure 6a wealth similarity is weighed less strongly ($\alpha = 1$) than in figure 6b ($\alpha = 2.0$).

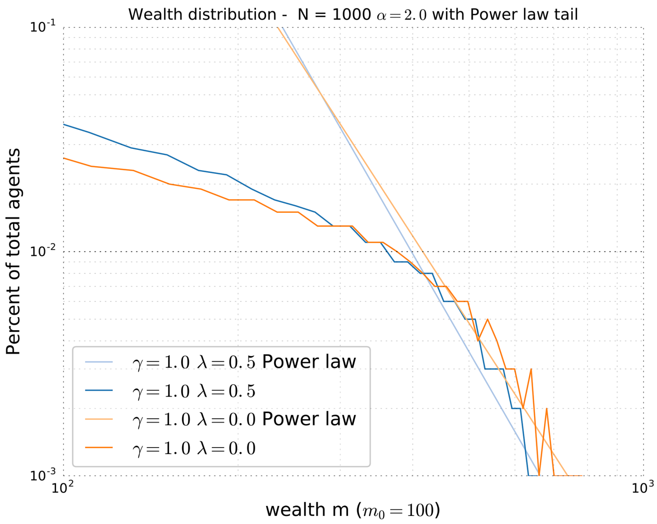


FIGURE 7. Power law fitting of wealth distribution including savings, nearest neighbhour- and previous transaction effects. The plot is logarithim and both includes ($\alpha = 0$) and excludes ($\alpha = 0.5$) savings.

plot is logarithmic and the Pareto power distribution is therefore presented as a straight line.

5. Discussion

Most part of this study was a replications of the work done by Patriaraca et al. (2004)[1] and Goswami and Sen (2014)[2] and we have more than sufficiently managed to do so, but the path to that point has regrettably involved a bit of guesswork. We found that both articles were somewhat lacking in procedural and algorithmic descriptions, which we find not to be consistent with sound science and the scientific method[7]. "[Just] as other information should be available to those who want to learn and understand, program source code is the only means for programmers to learn the art from their predecessors. It would be unthinkable for playwrights not to allow other playwrights to read their plays (...)" (Erik Naggum)[7]. That being said, we have confidence that our own results are very much comparable to both of of the studies that this study is based upon.

5.1. **The "savings rate" and time dynamics.** In this study we call the factor $\lambda \in [0, 1]$ the savings rate. It makes sense to employ it to the model described herein, as one would not assume that anyone would spend (potentially) their entire net worth on a single transaction. On the other hand, a savings rate would usually be defined as the fraction of income that an individual saves over a certain amount of time, for example over a year. The term "savings rate" as it is used in this model can therefore lead to some confusion, mostly because there is no concept of how time progresses throughout the simulation. The only thing that happens is a series of transactions which could have happened during any time span, and after a steady state is reached we have a snapshot of the monetary distribution, but no feel for the dynamics of the economy and how the situation may propagate through time. Such time dynamics functionality could be useful to see the time it takes for an economic of monetary policy to take effect, for instance.

5.2. **Future developments.** We found, after delving into the data, that the Parato distributions that fit the higher tails of the simulation distribution data best can be described by the following rule

$$w_m \propto m^{-1-\lambda-\alpha-\gamma} \tag{12}$$

where, of course, λ is the savings rate, α is the importance of wealth similarity and γ is the importance of previous transactions. This is a very neat result, but we have only confirmed that the Pareto distributions fits, and only adequately at the, in a visual manner. It would be good to confirm the explanatory power by some sort of regression analysis or a goodness-of-fit test. This was regrettably beyond the scope of our study.

A very important part of the social sciences, of which economics, is to try to have applications for the real world and help in a way to explain the behaviour of human beings. It would therefore be nice to compare the results of these simulations to real wealth distributions. As Piketty points out[5], data on wealth and income distributions have been available in the US since 1953. One study[8], in the Norwegian business magazine Kaptial uses the magazine's study of the 400 richest Norwegians to show that their wealth are indeed Pareto distributed, but what about the rest? We have simulated the distribution of the entire population here!

Norway is a special case because it is one of very few OECD[8] member countries that still have a wealth tax. These include Switzerland, Hungary, Spain and France[9]. In Norway, the wealth tax has a tremendous impact on high net worth individuals, making up 57.1% of the total tax burden of the 400 richest individuals[10]. It would therefore be beneficial to include tax effects in the model in some way, as different tax regimes will almost definitely have an impact of the monetary distribution.

We started the the theory section by stating a macroeconomic identity (equation 1). In it we included government spending, which necessarily is very important to

[7] We went as far as to e-mail some of the co-authors of the articles, asking (very nicely!) for descriptions of algorithms, software and methods. At the time of writing we are still without answer.

[8] Organisation for Economic Co-operation and Development. Colloquially defined as a "club for rich countries".

an economy. Depending on the relative size of the government, the spendings of the government could impact wealth distribution through welfare. Even though poor individuals may not have a high personal wealth it may not be necessary for them, because of the security that a welfare state provides. This would in turn affect savings rate in an economy. A huge "social security net" could lead to lower savings and vice versa.

In a modern, open economy one inevitably must trade with other economies. The macroeconomic identity in equation 1 can be expanded to net exports $X - M$, where X is exports and M is imports

$$Y = C + G + I + (X - M). \tag{13}$$

It would be interesting to see what opening the economy up for trade would do for wealth distribution. According to the Stolper-Samuelson theorem, opening an economy to trade should make unskilled workers producing traded goods in a high-skilled economy worse off as trade increases[11]. Advocates of protectionism usually use such theories to underline their argumentation. Building a model consisting of several "countries" with increasing amount of trade could be beneficial in this debate, especially as we have seen an increase in protectionist political rhetoric in recent times.

6. Summary Remarks

In this study we have used two academic articles in the field of econophysics[1][2] as a basis to build a simple model that mimics monetary distribution in a simple economy consisting of agents that are allowed to make monetary transactions with one another. The model is implemented in C++ employing Monte Carlo methods. The model is gradually expanded upon to include savings and social rigidities like wealth similarity bias and former transaction bias. We found that the wealth distribution seems to follow something similar to a Gibbs distribution for almost all simulations, with a very few number of high net worth individuals. The only situations where the simulations resulted in a wealth distribution with a relative amount of equality is with a very high savings rate($\lambda = 0.9$). We also found the upper tail of the wealth distribution to be close to, but not perfectly fitting, a Pareto distribution as has been the case since 1897[3]. This is studied visually, but remains to be statistically tested. Lastly, we suggest some possible future additions to the model, most importantly having several economies at once in a simulation with different amounts of trade, including a some sort of time dynamics in the model, and adding functionality that mimics a government that can provide welfare and social benefits.

References

[1] Patriarca, M., Chakraborti, A., & Kaski, K. (2004). Gibbs versus non-Gibbs distributions in money dynamics. *Physica A: Statistical Mechanics and its Applications*, 340(1), pp. 334-339.

[2] Goswami, S., & Sen, P. (2014). Agent based models for wealth distribution with preference in interaction. *Physica A: Statistical Mechanics and its Applications*, 415, pp. 514-524.

[3] Pareto, V. (1897). *Cours d'Économique politique*, Lausanne: Ed. Rouge.

[4] Arnold, B. C. (1983). *Pareto Distributions.* International Co-operative Publishing House

[5] Piketty, T., & Goldhammer, A. (2014). *Capital in the twenty-first century*, Cambridge: Harvard University Press.

[6] Moore, M., (March 7, 2011). The Forbes 400 vs. everybody else. *michaelmoore.com*. Retrieved December 8. 2016.

[7] Pharr, M., & Humphreys, G. (2010). *Physically Based Rendering*, Amsterdam: Elsevier.

[8] Aas, K. (2010). Stor formue - flaks eller dyktighet? *Kapital* 18, p. 30.

[9] OECD (2016). *OECD.stat* extract. `stats.oecd.org`. Retrieved December 9. 2016.

[10] Lieng, E., & Sveen, P. B. (2016). *katternivået til Norges rikeste : en analyse av skattebyrden i 2014 og formuesskattens regelendringer mellom 2006 og 2014*. Master Thesis. `http://hdl.handle.net/11250/2403838`. Retrieved December 9. 2016.

[11] Stolper, W. F., & Samuelson, P. A. (1941). Protection and real wages. *Economic Studies*. Oxford Journals. 9 (1), pp. 58-73.

Appendices

A. Parametrization from Patriarca et al.

A corresponding exact solution of an income distributions for a generic value of λ with $0 < \lambda < 1$ is provided in Patriarca et al.[1]. Fir one employs the reduced variable

$$x = \frac{m}{\langle m \rangle} \tag{14}$$

the agent money in units of average money $\langle m \rangle$ and the parameter

$$n(\lambda) = 1 + \frac{3\lambda}{1-\lambda}. \tag{15}$$

The money distributions, for arbitrary values of λ, are well fitted by the function

$$P_n(x) = a_n x^{n-1} e^{-nx} \tag{16}$$

where a_n is a normalization factor shown to be

$$a_n = \frac{n^n}{\Gamma(n)} \tag{17}$$