FYS3150 - Project 5
Financial Modelling

Daniel Heinesen, Gunnar Lange

November 23, 2016

Abstract

HEI

Contents

1 Introduction

2 Theoretical models

In this section we introduce the theoretical model necessary to model financial agents interacting among each other.

2.1 Modelling the interaction between two financial agents

We will begin with a simple model for the interaction of two agents. Assume that we have picked, at random, two agents i and j, each with wealth given by m_i and m_j. We will then model the interaction between this pair by drawing a random number, $\epsilon \in [0, 1]$, from a uniform distribution, and redistributing the wealth among the two according to the equations:

$$m_1' = \epsilon(m_1 + m_2) \tag{1}$$

$$m_2' = (1 - \epsilon)(m_1 + m_2) \tag{2}$$

Notice that the total wealth is conserved in this interaction, seeing as:

$$m_1' + m_2' = \epsilon(m_1 + m_2) + (1 - \epsilon)(m_1 + m_2) = m_1 + m_2$$

Thus we are only redistributing the wealth in the economy.

2.2 Implementing savings in the economy

We expand our model by including the possibility of our agents retaining a certain amount of money at every transaction. This is modelled by a parameter β, which describes the fraction of money saved by each agent in the transaction. Our models can then be written as:

$$
\begin{aligned}
m_i' &= \lambda m_i + \epsilon(1-\lambda)(m_i + m_j) \\
m_j' &= \lambda m_j + (1-\epsilon)(1-\lambda)(m_i + m_j)
\end{aligned}
\tag{3}
$$

Note that the total money is still conserved, seeing as:

$$m_i' + m_j' = \lambda m_i + \epsilon(1-\lambda)(m_i+m_j) + \lambda m_j + (1-\epsilon)(1-\lambda)(m_i+m_j) = \lambda(m_i+m_j) + (1-\lambda)(m_i+m_j) = m_i+m_j$$

2.3 Analytic solutions

3 Methods

3.1 Implementing the Monte Carlo simulation

To ensure that we get good results, we run multiple simulations, and average the final distribution in each of these simulations. This is summarized in Pseudo-code below:

```
num=zeros(shape=bins)

for i in range(no_of_experiments):
        for k in range(no_of_transactions):
                i=random.uniformint(0, N)
                j=random.uniformint(0, N)
                eps=random.uniform(0,1)
                money_i=m[i]
                money_j=m[k]
                m[i]=eps*(money_i+money_k)
                m[k]=(1-eps)*(money_i+money_k)
        for bin in bins:
                for agent in m:
                        if agent in bin:
                                num[bin]+=1
```

4 Results

5 Discussion

6 Conclusion