Journal of Mathematics and Modeling in Finance (JMMF)
Vol. 3, No. 1, Winter & Spring 2023
Research paper

The fast algorithm for computing all steady states in overlapping generations models

Alexey Zaytsev[1]

[1] RANEPA, Moscow, Russian Federation
zaytsev-av@ranepa.ru

Abstract:
Modern research often requires the use of economic models with multiple agents that interact over time. In this paper we research overlapping generations models, hereinafter OLG. In these models, the phenomenon of the multiplicity of long-term equilibrium may arise. This fact proves to be important for the theoretical justification of some economic effects, such as the collapse of the market and others. However, there is little theoretical research on the possibility of multiple equilibria in these models. At the same time, the works that exist are devoted to models with only few periods. This is due to the fact that the complexity of algorithms that calculate all long-term equilibria grows too fast with realistically selected lifespan values. However, solutions of some OLG models after the introduction of additional variables can become polynomial systems. Thus it is possible to represent many long-term equilibria as an algebraic variety. In particular, the Gröbner basis method became popular. However, this approach can only be used effectively when there are few variables. In this paper we consider the task of finding long-term equilibrium in overlapping generations models with many periods. We offer an algorithm for finding the systems solutions and use it to investigate the presence of multiple solutions in realistically calibrated models with long-lived agents. We also examine these models for multiple equilibria using the Monte Carlo method and replicate previously known results using a new algorithm.

Keywords: OLG-models, multiplicity of equilibrium, Gröbner basis, polynomial systems
Classification: JEL Codes: C02, C32, D11, D58

1 Introduction

In this article we apply computational methods to realistically calibrated overlapping generations models. In particular, a new equilibrium finding algorithm has been proposed. We use it to search for multiple equilibrium in several OLG models.

[2]Corresponding author

Received: 22/07/2023 Accepted: 07/10/2023
https://doi.org/10.22054/jmmf.2023.74945.1096

OLG models are used in many areas of modern economics, such as macroeconomics and finance (ALtig et al., 2001; Auerbach and Kotlikoff, 1983; Auerbach and Kotlikoff, 1987; Summers, 1981). These models have improved over time, adding several regions, various commodities, and demographic changes, such as Kotlikoff et al. (2007). Recent work on climate change has included OLG models (Bovenberg and Heijdra, 1998; Fried et al., 2018; Kotlikoff et al., 2021).

Unfortunately, these models may have multiple equilibria. Kehoe and Levine (1990) provides a simple example of a three-period model in which there is a multiplicity of long-term equilibria. However, Kubler and Schmedders (2010) states that with a large number of agent lifetimes, the probability of multiplicity is small.

The presence of multiple equilibria in the economic system can explain important phenomena such as market collapses (Hong and Stein (2003) and Basak et al. (2006)). Although this phenomenon is well studied for financial markets, less is known for housekeeping models. Information on the possibility of multiple equilibrium is of practical importance. Thus, with two equilibriums, one is preferable to the other, and the behaviour strategy of economic agents can change according to their expectations.

Basiri et al. (2022) proposes an alternative algorithm for finding a Gröbner basis for a model with production and endogenous labor. It is proposed to first find a polynomial from a single variable, which lies in ideal generated by the equations for the model, and, using the shape lemma, to complete it to the Gröbner basis. This algorithm significantly speeds up the search for a solution, but still spends a lot of time (more than a minute) finding equilibrium, making it unsuitable for the study of the phenomenon of multiple solutions.

This article uses a new algorithm that works even faster. In particular, it uses the method of quickly finding all real roots of a polynomial with certain properties. The new algorithm can be used to find multiple solutions.

We first look at the simplest model and replicate the results of Kubler and Schmedders (2010) using a new algorithm. Next we consider the model with exogenous labor and replicate the results of Basiri et al. (2022). Our algorithm allows us to conduct Monte Carlo experiments and examine the model for multiple equilibrium. Finally, were looking at an extended model with endogenous labor.

This work is organized is organized as follows: The second chapter describes a simple model with a known example of multiple solutions. In the third, we propose a new algorithm for finding equilibrium that does not use Gröbner bases. In Part Four, a new algorithm for finding the roots of a polynomial from a single variable was devised to speed up computation, allowing solutions of polynomials of the desired type to be considered. The fifth examines the expanded initial model with exogenous labor and the manufacturing firm and transfers all results. Chapter Six discusses the extended specification for endogenous labor. The seventh part will give examples of how the algorithm works, including multiple equilibrium in a simple model. Chapter eight contains the results of the search for possible multiple solutions in all the models reviewed and the time in which the algorithm finds solutions. Chapter 9 contains a brief description of the work done and the results achieved.

2 Simple model

Consider this model. Discrete time extends from $-\infty$ to $+\infty$. At each time one agent is born that lives for n periods. In the ath period, each agent receives a donation of e_a. For simplicity and without loss of generality it can be assumed that there is no time discount in the model and the utility function is not time dependent. Then the agent that was born in the $t$period maximizes the utility function

$$U_t = \sum_{a=1}^{n} u(c_a(t+a-1))$$

where $c_a(t+a-1)$ is consumption of this agent at period $t+a-1$. Suppose utility is independent of time and rate.

Equilibrium is given by the equation

$$\begin{cases} \sum_{a=1}^{n} \bar{c_a}(t) - e_a = 0 \\ (\bar{c_1}(t), \bar{c_2}(t+1), \dots, \bar{c_n}(t+n-1)) \in \underset{c(t),\dots,c(t_+n-1)}{\arg\max} \ U_t(c(t),\dots,c(t+n-1)) \\ \end{cases} \tag{1}$$

In this case, the maximum is subject to $\sum_{a=1}^{n} p(t+a-1)(c(t+a-1)-e_a) = 0$. Its consistent with the fact that agents trade with each other.

To solve this system you need to find an infinite number of prices and consumption. We cannot do this with our methods. Instead, we can look for long-term equilibrium.

The long-term status adds two conditions:

$$\begin{cases} \frac{p_{t+1}}{p_t} & = q \\ \bar{c_a}(t) & = c_a \end{cases} \tag{2}$$

There are two types of long-term equilibrium: monetary (at $q = 1$) and real ($q \neq 1$). We will be interested in the number of real solutions.

Applied research often uses the utility function $u(c) = \frac{c^{1-\sigma}}{1-\sigma}$ for $0 < \sigma < 1$ and the logarithmic function for $\sigma = 1$. Under these conditions, the system takes the form of:

$$\begin{cases} c_{a+1}^{\sigma}q - c_a^{\sigma} = 0, & a = 1, \dots, n-1 \\ \sum_{a=1}^{n} q^{a-1}(c_a - e_a) = 0 \\ \sum_{a=1}^{n}(c_a - e_a) = 0 \end{cases} \tag{3}$$

Indeed, the system has a solution at $q = 1$ and $c_a = (\frac{1}{n}\sum_{a=1}^{n} e_a)$. To find other solutions, we will replace $w = q^{\frac{1}{\sigma}}$. After the replacement, the system will appear as:

$$\begin{cases} c_{a+1}w - c_a = 0, \quad \text{a=1,\dots,n-1} & \text{(4a)} \\ \sum_{a=1}^{n}(c_a - e_a) = 0 & \text{(4b)} \\ \sum_{a=1}^{n} w^{\sigma(a-1)}(c_a - e_a) = 0 & \text{(4c)} \end{cases}$$

The solutions of this system (if you remove $w = 1$) are the long-term equilibrium of the model.

Note that the rate of root discovery depends on how many nonzero coefficients a polynomial has. Therefore, one should look for the most closed form of a given polynomial of all possible. In this case, the polynomial is expressed as the sum in which the number of members depends on the number of periods. This is due to the fact that e_a can be any real number, so the number of parameters of the system is linearly dependent on the number of periods.

3 Pipeline for finding solution

In this model, as in all the studies examined in this paper, the problem of finding a solution to a polynomial version of equations is reduced to finding the roots of one polynomial. Consider how this works in this case. Using 4*a*, we get equations for consumption in each period:

$$c_a = c_n w^{n-a} \tag{5}$$

By framing 5 in 4*b*, we express c_n through e_a:

$$\sum_{a=1}^{n}(c_n w^{n-a} - e_a) = 0$$

By converting this equation using the formula of the sum of the geometric progression and multiplied by $1 - w$, we get:

$$c_n(1 - w^n) = (\sum_{a=1}^{n} e_a)(1 - w) \tag{6}$$

Now you can substitute 5 and 6 in 4c:

$$\sum_{a=1}^{n} w^{\sigma(a-1)}((\sum_{a=1}^{n} e_a)(1-w)w^{n-a} - e_a) = 0$$

By using transformations similar to 6, we can make the equation appear

$$w^{\sigma-1}(1-w)(1-w^{(\sigma-1)n})(\sum_{a=1}^{n} e_a) - \sum_{a=1}^{n} w^{a\sigma}(1-w^n)(1-w^{\sigma-1})e_a = 0 \qquad (7)$$

The solutions of this polynomial can be extended to the solution of the entire system, namely, using intermediate equations 6 and 5. Indeed, knowing the roots of 7, you can substitute them in 6 and get the value of c_n, and then substitute it in 5. Thus, by reducing the system to one polynomial, we have saved information about the whole system through intermediate steps.

The resulting algorithm will look like this:

Reduce the problem to a system of polynomial equations, adding auxiliary variables if necessary.

Convert the system to one polynomial from one variable.

Find the real roots of this polynomial and weed out those that have no economic meaning (for example, those that have negative consumption).

Using intermediate calculations from step 2, recover the remaining variables.

4 Finding polynomial roots

The challenge is to find real solutions to the polynomial. In this case, it is desirable to think of an algorithm that will depend on the number of non-zero coefficients at most linearly.

This algorithm is based on Carvalho (2017), modifying it to solve a specific problem. The main idea of his work is that real roots can be found recursively, using knowledge of derivative roots. Namely, if there is a $f(x)$ polynomial, then between the roots of the derivative it is monotonically decreasing or increasing. Then, if there is a root on such a site, it can be quickly found using, for example, binary search or Newtons method. Thus, to calculate the roots of a polynomial, one must first calculate all its derivatives until the degree of the derivative is 1. Then you can easily find the root, and therefore you can continue to search for roots of derivatives of greater degree.

This algorithm has several drawbacks, the main one being a large number of recursion iterations. If the polynomial degree is 1000, it will look for the roots of 1000 polynomials. In our case, most coefficients are zeros. Then, if you use the standard implementation of polynomials as an array of coefficients, there will be a lot of nonnegative zeros stored in the memory. Instead, you can use the implementation of a polynomial as a dictionary, where the k integer key corresponds to degree, and the value is a factor at x^k.

If you want to calculate the value of a polynomial, for example at point 10, the computer will have to work with numbers of the order of 10^{1000}. Moreover, if you take a derivative of such a polynomial many times, the last polynomial will have coefficients of about 1000!, which makes computing on the computer even longer.

We propose a modification of this algorithm that eliminates these problems. First, one can use the fact that a polynomial has most coefficients equal to zero. In particular, most derivatives have a free coefficient equal to 0. The roots of such a polynomial are easier to find, because among them there is a zero root.

Lets say, when we take the derivative again, we have a polynomial of $f(x)$ with a zero free coefficient. Then instead of its roots one can look for the roots of the polynomial $\frac{f(x)}{x^k}$, where k is a multiplicity of zero It has a non-zero free coefficient, which eliminates unnecessary iterations. Then you can add the zero root and we get the roots we need.

Second, for binary search it is not so much the value in a point that is important as its sign. Therefore, it is possible to find the roots of the polynomial on the monotonous region knowing only if the value of the polynomial at a point is positive or negative. This can be used as follows: let us calculate the value of the $f(x)$ polynomial at $|a| > 1$. Instead, you can calculate the values of the rational function $\frac{f(x)}{x^{deg(f)}}$. Note that large values do not have problems with numbers which absolute values are less than 1.

dThird, to find the roots of a polynomial, you can multiply it by any constant and the solutions do not change. In this way, it is possible to avoid the problem of excessively high coefficients by keeping the senior coefficient equal to one at each step.

By applying all the above improvements to the algorithm in practice, we will get an algorithm whose speed is sufficient for our purposes.

5 Model with exogenous labor

First, consider a simpler model with exogenous labor. This model was considered in the work of Basiri et al. (2022).

The model has a discrete time that runs from $-\infty$ to ∞. There is one firm involved in the model which consumtion good is produced according to the Cobb-Douglas formula

$$Y_t = K_t^\alpha L_t^{1-\alpha} + (1-\delta)K_t$$

where Y_t denotes a firms output after price normalization, and K_t and L_t denotes capital and labor. Share capital is *alpha*, depreciation is *delta*. Then profit maximization conditions take the form of

$$\alpha K_t^{\alpha-1} L_t^{1-\alpha} = r_t + \delta$$

and

$$(1-\alpha) K_t^{\alpha} L_t^{-\alpha} = w_t$$

where r_t and w_t denote interest rates and wages respectively.

Also in the model there are households, each of which lives N periods. At the same time, the household that appeared in the period with the number t maximizes the utility function

$$U_t(c) = U(c) = \sum_{i=1}^{N} \beta^i v(c_{t+i-1,i})$$

Provided

$$k_{t+1,i} = (1+r_t)k_{t,i-1} + w_t l_i - c_{t,i}$$

where $c_{t,i}$, $k_{t,i}$ and $l_{t,i}$ denote household consumption, savings, and labor for $i$the lifetime of t. We assume that households do not value labor and normalize

$$\sum_{i=1}^{N} l_i = 1$$

It is important for our task to be able to characterize the solution of the system by polynomial equations, so it will be convenient to focus on the utility function

$$v(c) = \begin{cases} -c^{-\gamma} & \gamma > 0 \\ \ln(c) & \gamma = 0 \end{cases}$$

It is also necessary to consider the condition of supply and demand equilibrium for capital

$$K_t = \sum_{i=1}^{N} k_{t,i}$$

As with the small model, we will investigate the long-term equilibrium of the model. We also assume that all model parameters $(\alpha, \beta, \gamma, \delta, l_1, \ldots, l_N)$ are rational numbers. Then it is possible to write a system of equations, the solutions of which are the long-term equilibrium of the model.

$$\begin{cases} c_i & =\mathrm{k}_{i-1}(1+r)+wl_i-k_i \quad (i=1,\dots N) & \text{(8a)} \\ c_i^{-\gamma-1} & =\beta(1+r)c_{i+1}^{-\gamma-1} \quad (i=2,\dots N) & \text{(8b)} \\ 1+r & =\mathrm{K}^{\alpha-1}\alpha+1-\delta & \text{(8c)} \\ w & =\text{(1-}\alpha)K^{\alpha} & \text{(8d)} \\ K & =\sum_{i=1}^{N}k_i & \text{(8e)} \\ k_0 & =\mathrm{k}_N=0 & \text{(8f)} \end{cases}$$

We need a polynomial system, so well make some replacements. First, suppose that $\alpha=\frac{m}{n}$, where m and n are natural numbers. Secondly, we will introduce a new variable S, such that $Sn=K$. This ratio allows you to write $K^{\alpha-1}=S^{m-n}$ and finally multiplying 8c by S^{m-n} we get a polynomial equation. And third, enter a p variable such that $p^{\gamma+1}=\beta(1+r)$. After these changes, we get a polynomial system.

$$\begin{cases} c_i=k_{i-1}(1+r)+wl_i-k_i \quad (i=1,\dots N) & \text{(9a)} \\ c_i=pc_{i-1} \quad (i=2,\dots N) & \text{(9b)} \\ p^{\gamma+1}=\beta(1+r) & \text{(9c)} \\ K=S^n & \text{(9d)} \\ S^{n-m}(r+\delta)=\alpha & \text{(9e)} \\ w=(1-\alpha)S^m & \text{(9f)} \\ K=\sum_{i=1}^{N}k_i & \text{(9g)} \\ k_0=k_N=0 & \text{(9h)} \end{cases}$$

For our purposes it will be convenient to consider the case when household labor does not change from period to period ($l_i=\frac{1}{N}$) Lets write up a plan. From 9*b* we withdraw that

$$c_i=c_1p^{i-1} \tag{10}$$

Substituting 10 and 9*g* in the sum of all the equations from 9*a* and applying the formula for the sum of geometric progression, you get

$$\frac{p^N-1}{p-1}c_1=Kr+w=(S^nr+w) \tag{11}$$

Now you can multiply the equation by $p-1$ and get a polynomial

$$(p^N-1)c_1=(p-1)(S^nr+w) \tag{12}$$

Now lets consider the weighted sum of equations from 9*a* where the equation with the number i has the weight $(1+r)^{N-i}$. Then you can again use 10, the geometric progression formula twice and multiply by the required polynomial to get

$$((1+r)p-1)(p^N-(1+r)^N)c_1 = w((1+r)^N-1)(p-(1+r))/N \tag{13}$$

Now you can multiply this equation by $N(p^N-1)$ and use 12 and get

$$[N(S^n r+w)(p^N-1)(p-1)((1+r)p-1)(p^N-(1+r)^N)-w(p-(1+r))((1+r)^N p^N-1)]=0 \tag{14}$$

You can also express r and w from $9e$ and $9f$ and get a polynomial

$$\begin{split}S^m[N(\delta\alpha\beta-p^\gamma+\beta(1-\delta))(p^N-1)(p-1)(p^{(\gamma+1)}-\beta)(\beta^N p^N-p^{\gamma N})-\\ \alpha(p^\gamma-\beta(1-\delta))(\beta p-p^\gamma)(p^{(\gamma+1)N}-\beta^N)]=0\end{split} \tag{15}$$

S was defined so that the solution of $S=0$ made no economical sense, so you can simplify this to a 1-variable polynomial

$$\begin{split}N(\delta\alpha\beta-p^\gamma+\beta(1-\delta))(p^N-1)(p-1)(p^{(\gamma+1)}-\beta)(\beta^N p^N-p^{\gamma N})-\\ \alpha(p^\gamma-\beta(1-\delta))(\beta p-p^\gamma)(p^{(\gamma+1)N}-\beta^N)=0\end{split} \tag{16}$$

Using the algorithm described in chapter three, we can find solutions.

6 Model with endogenous labor

We will expand the OLG model to include a more natural endogenous condition. Some equations remain the same:

$$\begin{cases}\alpha K_t^{\alpha-1}L_t^{1-\alpha}=r_t+\delta & \text{(17a)}\\ (1-\alpha)K_t^\alpha L_t^{-\alpha}=w_t & \text{(17b)}\\ k_{t+1,i}=(1+r_t)k_{t,i-1}+w_t l_i-c_{t,i} & \text{(17c)}\\ K_t=\sum_{i=1}^{N}k_{t,i} & \text{(17d)}\end{cases}$$

Now labor is endogenous, so:

$$L=\sum l_{t,i}$$

Note that in the exogenous model this condition was also fulfilled, but was not an equation, since l_is were not variables. Also, households themselves now choose how much they work in each period, so negative utility from labor must be added to the utility function.

$$U(c,l) = \sum_{i=1}^{N} \beta^i \left(\frac{c_i^{1-\theta} - 1}{1-\theta} - \nu \frac{l_i^{1+\gamma}}{1+\gamma}\right)$$

Since we are looking for long-term equilibrium, the system looks like:

$$\begin{cases} c_i = k_{i-1}(1+r) + wl_i - k_i \quad (i = 1, \ldots, N) & \text{(18a)} \\ \beta(1+r)c_i^{-\theta} = c_{i-1}^{-\theta} \quad (i = 2, \ldots, N) & \text{(18b)} \\ l_i^{-\gamma} = \beta(1+r)l_{i-1}^{-\gamma} \quad (i = 2, \ldots, N) & \text{(18c)} \\ \nu c_i^{\theta} l_i^{\gamma} = w \quad (i = 1, \ldots, N) & \text{(18d)} \\ 1 + r = \alpha K^{\alpha-1} L^{1-\alpha} + 1 - \delta & \text{(18e)} \\ w = (1-\alpha)K^{\alpha} L^{-\alpha} & \text{(18f)} \\ K = \sum_{i=1}^{N} k_i & \text{(18g)} \\ L = \sum_{i=1}^{N} l_i & \text{(18h)} \\ k_0 = k_N = 0 & \text{(18i)} \end{cases}$$

Make the replacements:

$$S^n = \frac{K}{L}$$

and

$$p^{\theta\gamma} = \beta(1+r)$$

Then the equation system will take the form:

$$\begin{cases} c_i = k_{i-1}(1+r) + wl_i - k_i \quad (i = 1, \ldots, N) \\ c_i = p^\gamma c_{i-1} \quad (i = 2, \ldots, N) \\ p^\theta l_i = l_{i-1} \quad (i = 2, \ldots N) \\ p^{\theta\gamma} = \beta(1+r) \\ K = S^n L \\ \nu c_i^\theta l_i^\gamma = w \\ S^{n-m}(r+\delta) = \alpha \\ w = (1-\alpha)S^m \\ K = \sum k_a \\ L = \sum l_a \\ k_0 = k_N = 0 \end{cases} \tag{19}$$

We can simplify the system as in the previous chapter:
Lets get rid of consumption like in 12

$$\frac{p^{\gamma N} - 1}{p^\gamma - 1} c_1 = Kr + wL = L(S^n r + w) \tag{20}$$

$$(p^{\gamma N} - 1)c_1 = (p^\gamma - 1)L(S^n r + w) \tag{21}$$

You can also get rid of the labor:

$$(p^\theta - 1)L = (p^{\theta N} - 1)l_N \tag{22}$$

Now it is possible to link labor and remaining consumption in one equation derived similarly to 13

$$((1+r)p^\theta - 1)(p^{\gamma N} - (1+r)^N)c_1 = wl_N((1+r)^N p^{\theta N} - 1)(p^\gamma - (1+r)) \tag{23}$$

Lets combine the three previous equations into one and get:

$$\begin{split} L[(S^n r + w)(p^{\theta N} - 1)(p^\gamma - 1)((1+r)p^\theta - 1)(p^{\gamma N} - (1+r)^N) - \\ w(p^{\gamma N} - 1)(p^\theta - 1)(p^\gamma - (1+r))((1+r)^N p^{\theta N} - 1)] = 0 \end{split} \tag{24}$$

Now, similarly to the last step in the previous chapter, lets express r and w:

$$\begin{split} LS^m[(p^{\theta\gamma} - \beta(1-\delta) - \delta\alpha\beta)(p^{\theta N} - 1)(p^\gamma - 1)(p^{\theta(\gamma+1)} - \beta)(\beta^N p^{\gamma N} - p^{\theta\gamma N}) - \\ (1-\alpha)(p^{\theta\gamma} - \beta(1-\delta))(p^{\gamma N} - 1)(p^\theta - 1)(\beta p^\gamma - p^{\theta\gamma})(p^{\theta(\gamma+1)N} - \beta^N)] = 0 \end{split} \tag{25}$$

We obtain the required polynomial in one variable.

7 Examples

We have implemented the algorithm for finding roots of polynomials in Python 3.10.9. You can look at source code on https://github.com/SellBound/polysolve. We can randomly generate economic parameters for each model. After that we used formulas for polynomials in one variable for each model, computed them and found its roots. We can also find all other variables and test solutions for economic relevance. Below are examples of how the algorithm works.

7.1 Simple model

To begin with, here is an example of finding multiple equilibrium in a simple model.

N	σ	e_1	e_2	e_3	e_4	e_5
5	5	0.09693	0.88628	0.7565	0.38374	0.31883

Table 1: Simple model parameters

We will find equilibrium in the model with the parameters specified in the table 1. By substituting these parameters into the formula for a polynomial in one variable, we get 26

$$\begin{aligned} &-0.3188329x^{29} + 2.761111x^{25} - 2.507194x^{24} + 0.06491200000000003x^{20} \\ &\quad - 0.37276x^{19} + 0.37276x^{15} - -0.129776x^{14} + 0.129776x^{10} + 0.789343x^{9} \\ &\qquad - 3.231625x^{5} + 2.539216x^{4} - 0.096934 \end{aligned} \tag{26}$$

Using the new algorithm, we can find all the real roots of the polynomial above. However, it is not yet a root of a whole system. We can try to continue with this root until the whole system is solved, but there may be problems. First, when using the algorithm, we sometimes multiplied many terms from our system by other polynomials. The roots of these polynomials appear in the solution, but we should not take them into consideration. Secondly, negative solutions make no economic sense, so we forget them too. Third, in the process of calculating the complete solution, negative values of consumption, labor and other indicators may appear. These cases dont suit us either. Here and further by filtering we will understand the selection of roots according to the above criteria. After eliminating the redundant, there will be three solutions:

In this table, U represents the total utility in all periods.

This shows that the algorithm, among others, can find multiple solutions. So in two equilibrium there is deflation (for example, in the first decision prices are reduced by 10 times per period). In these models, consumption increases over time. In a single example of $q > 1$, inflation between periods exceeds 50 percent. In this equilibrium, consumption decreases.

w	q	c_1	c_2	c_3	c_4	c_5	U
0.65238	0.11816	0.17439	0.26731	0.40975	0.62807	0.96274	-330.0354
0.88619	0.54655	0.37806	0.42661	0.48140	0.54322	0.61298	-29.08206
1.09631	1.58367	0.58214	0.53100	0.48435	0.44180	0.40298	-25.90597

Table 2: Long-term equilibrium

The only optimal equilibrium is achieved only in the third case with a positive interest rate. In this case, the first equilibrium is the most suboptimal and should be avoided.

7.2 Model with exogenous labour

Now lets look at an example of finding long-term equilibrium in a model with exogenous labor. We generate random parameters:

N	σ	β	δ	α	γ
70	4	0.9780	0.0560	0.5348	7

Table 3: Parameters of the model with exogenous labour

The table 3 shows the parameters of the model from which we will look for equilibrium. Using the algorithm, we find the polynomial in one variable:

$$\begin{aligned} &-0.005999586708825199x^{631} + 1x^{577} - 0.9933554817275747x^{576} \\ &- 1.9370923749882019x^{569} + 1.9244594692916281x^{568} + 0.9376348296629071x^{561} \\ &- 0.931635242954082x^{560} - 0.2115421881527618x^{87} + 0.2129477840873316 4x^{86} \\ &+ 0.40702726232947406x^{79} - 0.40969965484325477x^{78} - 0.19581099215433925x^{71} \\ &+ 0.19708015785473637x^{70} - 0.001405595934569846x^{16} + 0.0013747486647801611x^{9} \\ &+ 0.0012976438490005303x^{8} - 0.0012691657003971093 = 0 \end{aligned} \tag{27}$$

Note that this polynomial has degree 631; for a standard root algorithm, this would take a long time to solve. But it has only 17 non-zero coefficients, which makes it much easier to find using the new algorithm. You can now find the real roots and filtering out the economically meaningless. There remains one solution with $p = 1.0196$ and $r = 0.1939$, the results can be found in the table 5.

The results show that during the lifetime of the agent, its consumption increases by 4.5 times. The capital chart is in the form of an inverted letter U. In this case, the peak of savings is reached in the period with number 58 and takes a value of 2.5 times more than the peak consumption.

7.3 Model with endogenous labor

Now lets look at an example of finding long-term equilibrium in an endogenous model. Generate random parameters (table 4).

N	σ	β	δ	α	γ	θ	ν
70	5	0.9827	0.0813	0.4	3	4	9651

Table 4: Parameters of the model with endogenous labor

Using the algorithm, we find the polynomial from one variable (the equation 28).

$$\begin{aligned}
&0.6x^{1358} - 0.6x^{1354} - 0.5895953281731733x^{1349} - 0.5416398185960283x^{1346} \\
&+ 0.5895953281731733x^{1345} + 0.5416398185960283x^{1342} + 0.5322471776613056x^{1337} \\
&- 0.5322471776613056x^{1333} - 1x^{1151} + 0.4x^{1148} + 0.6x^{1144} + 1.5242986988846505x^{1139} \\
&- 0.39306355211544897x^{1136} + 0.39306355211544897x^{1135} - 1.5242986988846505x^{1132} \\
&- 0.5322471776613056x^{1127} - 0.38624739000403563x^{1123} + 0.9184945676653412x^{1120} \\
&+1x^{871} - 1x^{868} - 0.9347033707114772x^{859} + 0.9347033707114772x^{856} - 0.9826588802886222x^{855} + \\
&0.9826588802886222x^{852} + 0.9184945676653412x^{843} - 0.9184945676653412x^{840} + \\
&0.2938967785528849x^{521} - 0.2938967785528849x^{518} - 0.2747063095546261x^{509} + \\
&0.2747063095546261x^{506} - 0.288800279333211x^{505} + 0.288800279333211x^{502} + \\
&0.2699425945551685x^{493} - 0.2699425945551685x^{490} - 0.2938967785528849x^{241} + \\
&0.11755871142115396x^{238} + 0.17633806713173092x^{234} + 0.4479864771545527x^{229} \\
&- 0.11552011173328444x^{226} + 0.11552011173328441x^{225} - 0.44798647715455264x^{222} \\
&- 0.15642573090852271x^{217} - 0.11351686364664579x^{213} + 0.2699425945551685x^{210} \\
&+ 0.17633806713173092x^{28} - 0.17633806713173092x^{24} - 0.1732801675999266x^{19} - \\
&0.15918619782134166x^{16} + 0.1732801675999266x^{15} + 0.15918619782134166x^{12} \\
&+ 0.15642573090852271x^{7} - 0.15642573090852271x^{3}
\end{aligned} \tag{28}$$

This polynomial has degree 1358; applying standard methods to it is useless, but our algorithm does it in a fraction of a second.

Now you can find the real roots and filter out the economically worthless ones, and you get one solution that you can find in the table 6.

Lets analyze the obtained results: First, a realistic interest rate of 4.2%. Second, during the lifetime of the agent, consumption increased by 50%, and hil labor decreased by 40%. Third, the capital curve takes the form of a parabola with branches down with a peak at period 43 and a value that is 5 times higher than peak consumption.

8 Monte-Carlo experiments

8.1 Simple model

For the simplest model, the results depend on the parameters N and σ.

The table 7 provides information on how long it took to solve $1,000$ models with parameters e_i generated evenly randomly from 0 to 1. As can be seen from the table, results depend on a combination of N and σ, and may also depend on parity. But still, its enough to run a lot of tests in a relatively short time.

Note that our algorithm is not designed to work with small models, so classical methods can work faster. However, this does not change the fact that the difference between these models is less obvious, so the algorithm can be applied to small models as well.

8.2 Model with exogenous labor

For the exogenous model, 100 simulation tests were performed. At the same time, the parameters of α, β and δ were generated randomly from a uniform distribution on a segment from 0 to 1. The γ was a random natural number from 5 to 20 and σ from 3 to 10.

The table 8 contains information on how long it took to solve 100 models in the t_{exo} column.

Basiri et al. (2022) describes an algorithm that finds equilibrium in a 70-period model. It takes the algorithm 70 seconds simply to find the real roots of the polynomial in one model and even more time to find a complete solution. At the same time our model in 2 seconds finds solutions to 100 models and time does not depend on the number of periods. It can be concluded that the new algorithm works much more efficiently than the existing solutions.

8.3 Model with endogenous labor

Now lets look at the algorithm results for the endogenous problem.

The parameters of σ, α, β, δ and γ were randomly generated in the same way as in the exogenous model. θ was generated in the same way as γ, and ν with a $\frac{1}{2}$ probability was x or $\frac{1}{x}$, where x is a random amount taken from a uniform distribution of $[0.1]$.

The table 8 in the t_{endo} column contains information on how long it takes the algorithm to calculate the equilibrium in 100 models.

Note that the given problem is solved almost an order of magnitude slower, because the polynomial whose roots need to be found has more nonzero coefficients. There is another plus of our algorithm: it doesnt depend on the number of periods, because the kind of multiplication does not depend on N.

8.4 Study models for multiple equilibria

Multiple equilibrium tests were also performed. For a simple model, there were already results describing the probability of multiple equilibrium, which were described in Kubler and Schmedders (2010). The new algorithm replicated the results of this work. Tests were also performed on two models with labor. α, β, γ, δ parameters were randomly sampled in the same way as exogenous models. The θ parameter was sampled just like the γ. And the ν parameter was generated this way: first we got a random number of x from a uniform distribution from 0 to 1 and then with a probability of $\frac{1}{2}$ ν took the value of x or $\frac{1}{x}$. More than 50,000 iterations were made for both models, but no multiplicity was found, which gives hope that there is no multiplicity of equilibrium in realistically calibrated models.

9 Conclusion

In this paper we described a new algorithm for finding long-term equilibrium in OLG models. In the process, a method was devised to quickly find the real roots of the polynomial. The new algorithm gave a significant increase in the speed of work, which allowed the study of economic models with the Monte Carlo method. The new algorithm replicated results relating to the existence of examples of multiplicity in models with few periods and without labor. In particular, it was observed that only one of these results was an effective solution. Exo- and endogenous models have also been investigated for the existence of examples of multiplicity. After many iterations, none were found. This may mean that in realistically calibrated models, a multiplicity of long-term equilibria is unlikely.

10 Appendix

$S = 1.0388$	$w = 1.1156$	$c_{23} = 0.0219$	$k_{23} = 0.0461$	$c_{47} = 0.0349$	$k_{47} = 0.1181$
$K = 5.1326$	$k_0 = 0$	$c_{24} = 0.0224$	$k_{24} = 0.0486$	$c_{48} = 0.0356$	$k_{48} = 0.1214$
$c_1 = 0.0143$	$k_1 = 0.0016$	$c_{25} = 0.0228$	$k_{25} = 0.0512$	$c_{49} = 0.0363$	$k_{49} = 0.1245$
$c_2 = 0.0146$	$k_2 = 0.0032$	$c_{26} = 0.0233$	$k_{26} = 0.0537$	$c_{50} = 0.037$	$k_{50} = 0.1275$
$c_3 = 0.0149$	$k_3 = 0.0049$	$c_{27} = 0.0237$	$k_{27} = 0.0564$	$c_{51} = 0.0378$	$k_{51} = 0.1305$
$c_4 = 0.0152$	$k_4 = 0.0066$	$c_{28} = 0.0242$	$k_{28} = 0.0591$	$c_{52} = 0.0385$	$k_{52} = 0.1332$
$c_5 = 0.0155$	$k_5 = 0.0084$	$c_{29} = 0.0247$	$k_{29} = 0.0618$	$c_{53} = 0.0392$	$k_{53} = 0.1357$
$c_6 = 0.0158$	$k_6 = 0.0101$	$c_{30} = 0.0251$	$k_{30} = 0.0646$	$c_{54} = 0.04$	$k_{54} = 0.1379$
$c_7 = 0.0161$	$k_7 = 0.0119$	$c_{31} = 0.0256$	$k_{31} = 0.0674$	$c_{55} = 0.0408$	$k_{55} = 0.1398$
$c_8 = 0.0164$	$k_8 = 0.0138$	$c_{32} = 0.0261$	$k_{32} = 0.0703$	$c_{56} = 0.0416$	$k_{56} = 0.1413$
$c_9 = 0.0167$	$k_9 = 0.0157$	$c_{33} = 0.0266$	$k_{33} = 0.0733$	$c_{57} = 0.0424$	$k_{57} = 0.1422$

$c_{10} = 0.0171$	$k_{10} = 0.0176$	$c_{34} = 0.0272$	$k_{34} = 0.0763$	$c_{58} = 0.0432$	$k_{58} = 0.1425$
$c_{11} = 0.0174$	$k_{11} = 0.0195$	$c_{35} = 0.0277$	$k_{35} = 0.0793$	$c_{59} = 0.0441$	$k_{59} = 0.1419$
$c_{12} = 0.0177$	$k_{12} = 0.0215$	$c_{36} = 0.0282$	$k_{36} = 0.0824$	$c_{60} = 0.0449$	$k_{60} = 0.1404$
$c_{13} = 0.0181$	$k_{13} = 0.0235$	$c_{37} = 0.0288$	$k_{37} = 0.0855$	$c_{61} = 0.0458$	$k_{61} = 0.1378$
$c_{14} = 0.0184$	$k_{14} = 0.0256$	$c_{38} = 0.0293$	$k_{38} = 0.0886$	$c_{62} = 0.0467$	$k_{62} = 0.1337$
$c_{15} = 0.0188$	$k_{15} = 0.0277$	$c_{39} = 0.0299$	$k_{39} = 0.0918$	$c_{63} = 0.0476$	$k_{63} = 0.1279$
$c_{16} = 0.0192$	$k_{16} = 0.0299$	$c_{40} = 0.0305$	$k_{40} = 0.0951$	$c_{64} = 0.0486$	$k_{64} = 0.1201$
$c_{17} = 0.0195$	$k_{17} = 0.0321$	$c_{41} = 0.0311$	$k_{41} = 0.0983$	$c_{65} = 0.0495$	$k_{65} = 0.1098$
$c_{18} = 0.0199$	$k_{18} = 0.0343$	$c_{42} = 0.0317$	$k_{42} = 0.1016$	$c_{66} = 0.0505$	$k_{66} = 0.0965$
$c_{19} = 0.0203$	$k_{19} = 0.0366$	$c_{43} = 0.0323$	$k_{43} = 0.1049$	$c_{67} = 0.0515$	$k_{67} = 0.0797$
$c_{20} = 0.0207$	$k_{20} = 0.0389$	$c_{44} = 0.033$	$k_{44} = 0.1083$	$c_{68} = 0.0525$	$k_{68} = 0.0586$
$c_{21} = 0.0211$	$k_{21} = 0.0412$	$c_{45} = 0.0336$	$k_{45} = 0.1116$	$c_{69} = 0.0535$	$k_{69} = 0.0323$
$c_{22} = 0.0215$	$k_{22} = 0.0436$	$c_{46} = 0.0343$	$k_{46} = 0.1149$	$c_{70} = 0.0546$	$k_{70} = 0$

Table 5: Long-term equilibrium in model with exogenous labor

$p = 1.002$	$r = 0.0422$	$S = 1.2163$	$w = 1.3131$	$L = 56.2174$	$K = 398.3248$
$c_1 = 1.0464$	$l_1 = 1.0431$	$k_1 = 0.3233$	$c_{36} = 1.2893$	$l_{36} = 0.7897$	$k_{36} = 8.3663$
$c_2 = 1.0527$	$l_2 = 1.0348$	$k_2 = 0.6431$	$c_{37} = 1.297$	$l_{37} = 0.7834$	$k_{37} = 8.4512$
$c_3 = 1.0589$	$l_3 = 1.0266$	$k_3 = 0.9593$	$c_{38} = 1.3047$	$l_{38} = 0.7772$	$k_{38} = 8.5238$
$c_4 = 1.0653$	$l_4 = 1.0185$	$k_4 = 1.2719$	$c_{39} = 1.3125$	$l_{39} = 0.7711$	$k_{39} = 8.5836$
$c_5 = 1.0717$	$l_5 = 1.0104$	$k_5 = 1.5807$	$c_{40} = 1.3204$	$l_{40} = 0.765$	$k_{40} = 8.63$
$c_6 = 1.0781$	$l_6 = 1.0024$	$k_6 = 1.8857$	$c_{41} = 1.3283$	$l_{41} = 0.7589$	$k_{41} = 8.6626$
$c_7 = 1.0845$	$l_7 = 0.9945$	$k_7 = 2.1866$	$c_{42} = 1.3362$	$l_{42} = 0.7529$	$k_{42} = 8.6807$
$c_8 = 1.091$	$l_8 = 0.9866$	$k_8 = 2.4834$	$c_{43} = 1.3442$	$l_{43} = 0.7469$	$k_{43} = 8.6837$
$c_9 = 1.0975$	$l_9 = 0.9788$	$k_9 = 2.776$	$c_{44} = 1.3522$	$l_{44} = 0.741$	$k_{44} = 8.6711$
$c_{10} = 1.1041$	$l_{10} = 0.971$	$k_{10} = 3.0642$	$c_{45} = 1.3603$	$l_{45} = 0.7352$	$k_{45} = 8.6421$
$c_{11} = 1.1107$	$l_{11} = 0.9633$	$k_{11} = 3.3478$	$c_{46} = 1.3685$	$l_{46} = 0.7293$	$k_{46} = 8.5961$
$c_{12} = 1.1173$	$l_{12} = 0.9557$	$k_{12} = 3.6267$	$c_{47} = 1.3767$	$l_{47} = 0.7236$	$k_{47} = 8.5324$
$c_{13} = 1.124$	$l_{13} = 0.9482$	$k_{13} = 3.9008$	$c_{48} = 1.3849$	$l_{48} = 0.7178$	$k_{48} = 8.4503$
$c_{14} = 1.1307$	$l_{14} = 0.9406$	$k_{14} = 4.1699$	$c_{49} = 1.3932$	$l_{49} = 0.7121$	$k_{49} = 8.3489$
$c_{15} = 1.1375$	$l_{15} = 0.9332$	$k_{15} = 4.4338$	$c_{50} = 1.4015$	$l_{50} = 0.7065$	$k_{50} = 8.2276$
$c_{16} = 1.1443$	$l_{16} = 0.9258$	$k_{16} = 4.6923$	$c_{51} = 1.4099$	$l_{51} = 0.7009$	$k_{51} = 8.0854$
$c_{17} = 1.1511$	$l_{17} = 0.9185$	$k_{17} = 4.9453$	$c_{52} = 1.4183$	$l_{52} = 0.6954$	$k_{52} = 7.9214$
$c_{18} = 1.158$	$l_{18} = 0.9112$	$k_{18} = 5.1925$	$c_{53} = 1.4268$	$l_{53} = 0.6898$	$k_{53} = 7.7348$
$c_{19} = 1.165$	$l_{19} = 0.904$	$k_{19} = 5.4338$	$c_{54} = 1.4353$	$l_{54} = 0.6844$	$k_{54} = 7.5247$

$c_{20} = 1.1719$	$l_{20} = 0.8968$	$k_{20} = 5.6688$	$c_{55} = 1.4439$	$l_{55} = 0.679$	$k_{55} = 7.2899$
$c_{21} = 1.1789$	$l_{21} = 0.8897$	$k_{21} = 5.8975$	$c_{56} = 1.4526$	$l_{56} = 0.6736$	$k_{56} = 7.0296$
$c_{22} = 1.186$	$l_{22} = 0.8827$	$k_{22} = 6.1195$	$c_{57} = 1.4612$	$l_{57} = 0.6683$	$k_{57} = 6.7426$
$c_{23} = 1.1931$	$l_{23} = 0.8757$	$k_{23} = 6.3346$	$c_{58} = 1.47$	$l_{58} = 0.663$	$k_{58} = 6.4277$
$c_{24} = 1.2002$	$l_{24} = 0.8687$	$k_{24} = 6.5425$	$c_{59} = 1.4788$	$l_{59} = 0.6577$	$k_{59} = 6.0839$
$c_{25} = 1.2074$	$l_{25} = 0.8619$	$k_{25} = 6.743$	$c_{60} = 1.4876$	$l_{60} = 0.6525$	$k_{60} = 5.7099$
$c_{26} = 1.2146$	$l_{26} = 0.855$	$k_{26} = 6.9358$	$c_{61} = 1.4965$	$l_{61} = 0.6473$	$k_{61} = 5.3045$
$c_{27} = 1.2219$	$l_{27} = 0.8483$	$k_{27} = 7.1206$	$c_{62} = 1.5055$	$l_{62} = 0.6422$	$k_{62} = 4.8662$
$c_{28} = 1.2292$	$l_{28} = 0.8416$	$k_{28} = 7.297$	$c_{63} = 1.5145$	$l_{63} = 0.6371$	$k_{63} = 4.3937$
$c_{29} = 1.2365$	$l_{29} = 0.8349$	$k_{29} = 7.4648$	$c_{64} = 1.5235$	$l_{64} = 0.6321$	$k_{64} = 3.8856$
$c_{30} = 1.2439$	$l_{30} = 0.8283$	$k_{30} = 7.6235$	$c_{65} = 1.5326$	$l_{65} = 0.6271$	$k_{65} = 3.3404$
$c_{31} = 1.2514$	$l_{31} = 0.8217$	$k_{31} = 7.773$	$c_{66} = 1.5418$	$l_{66} = 0.6221$	$k_{66} = 2.7565$
$c_{32} = 1.2589$	$l_{32} = 0.8152$	$k_{32} = 7.9127$	$c_{67} = 1.551$	$l_{67} = 0.6172$	$k_{67} = 2.1323$
$c_{33} = 1.2664$	$l_{33} = 0.8088$	$k_{33} = 8.0423$	$c_{68} = 1.5603$	$l_{68} = 0.6123$	$k_{68} = 1.4659$
$c_{34} = 1.274$	$l_{34} = 0.8023$	$k_{34} = 8.1613$	$c_{69} = 1.5696$	$l_{69} = 0.6074$	$k_{69} = 0.7558$
$c_{35} = 1.2816$	$l_{35} = 0.796$	$k_{35} = 8.2695$	$c_{70} = 1.579$	$l_{70} = 0.6026$	$k_{70} = 0$

Table 6: Long-term equilibrium in model with endogenous labor

N	t_{exo}	t_{endo}
5	2.040	13.720
10	1.680	13.151
15	2.037	13.409
20	1.659	13.127
25	2.144	14.543
30	1.822	13.968
35	2.130	14.533
40	1.822	13.632
45	2.106	14.737
50	1.768	13.802
55	2.127	14.270
60	1.797	13.922
65	2.121	14.383
70	1.775	13.334

Table 8: Time spent for finding equilibrium in models with exo- and endogenous labor 100 times

N	σ	t
3	3	1.886
3	4	2.101
3	5	3.300
3	6	3.323
3	7	3.751
4	3	4.943
4	4	2.367
4	5	5.454
4	6	7.020
4	7	5.733
5	3	6.258
5	4	4.522
5	5	4.029
5	6	4.447
5	7	12.910
3	10	2.555
3	15	4.032
3	20	2.638

Table 7: Time spent for finding equilibrium in simple model 1000 times

Acknowledgement

The article was prepared as part of the research work of the state task of RANEPA..

Bibliography

[1] Altig A.,Auerbach A.,Kotlikoff L., 2001, Simulating fundamental tax reform in the United States *American Economic Review*, **91 (3)**, 574-595.

[2] Auerbach A.,Kotlikoff L., 1987, Evaluating fiscal policy with a dynamic simulation model, *The American Economic Review*, **77(2)** 49-55.

[3] Auerbach A.,Kotlikoff L., National savings, economic welfare, and the structure of taxation, 1983, *Behavioral simulation methods in tax policy analysis*, 459-498

[4] Basak S.,Cass D.,Manuel J.,Pavlova A., 2006, Multiplicity in General Financial Equilibrium with Portfolio Constraints, *Institute for Economic Research (PIER) Working Paper Series.*

[5] Basiri A.,Riahi M,,Kubler F.,Rahmany S., 2023, Efficient Calculation of All Steady States in Large-Scale Overlapping Generations Models, *Journal of Mathematics and Modeling in Finance*, https://doi.org/10.22054/jmmf.2023.71545.1083

[6] Becker T.,Weispfenning V., 2012, Gröbner bases: a computational approach to commutative algebra, *Springer Science & Business Media*, **141**.

[7] Bovenberg L.,Heijdra B. 1998, Environmental tax policy and intergenerational distribution, *Journal of Public Economics*, **67**, 1-24.

[8] Carvalho O., 2017, A simple recursive algorithm to find all real roots of a polynomial, *Researchgate.*

[9] Fried S,,Novan K.,Peterman W., 2018, The distributional effects of a carbon tax on current and future generations, *Review of Economic Dynamics*, **30**, 30-46.

[10] Hong H.,Stein J., 2003, Differences of Opinion, Short-Sales Constraints, and Market Crashes, *Review of Financial Studies*, **16**, 487-525.

[11] Kehoe, T.,Levine D., 1990, The economics of indeterminacy in overlapping generations models, *Journal of Public Economics*, **42**, 219-243.

[12] Kotlikoff L.,Kubler F.,Polbin A.,Sachs J.,Scheidegger S., 2021, Making carbon taxation a generational win win, *International Economic Review*, **62(1)**, 3-46.

[13] Kotlikoff L.,Smetters K.,Walliser J., 2007, Mitigating America's demographic dilemma by pre-funding social security, *Journal of monetary Economics*, **54**, 247-266.

[14] Kubler F.,Schmedders K., 2010, Tackling multiplicity of equilibria with Gröbner bases, *Operations research*, **58(4-part2)**, 1037-1050.

[15] Kubler F.,Schmedders K., 2010, Uniqueness of steady states in models with overlapping generations, *Journal of the European Economic Association*, **8(2-3)**, 635-644.

[16] Summers L., 1981, Capital Taxation and Accumulation in a Life Cycle Growth Model, *The American Economic Review*, **71**, 533-544.

How to Cite: Alexey Zaytsev[1], *The fast algorithm for computing all steady states in overlapping generations models*, Journal of Mathematics and Modeling in Finance (JMMF), Vol. 3, No. 1, Pages:203–222, (2023).