Theory of Income I Computational Problem Set 2
Fall 2020

Prepared by
Chase Abram *

November 22, 2020

1 Motivation

This assignment is meant to be a hands-on exposure to the most basic heterogeneous-agent consumption-savings model with production. My hope is that by the end you are familiar with the basic components of a heterogeneous-agent model, and can see how it differs from a representative-agent approach.

There are a lot of moving pieces in these models, and asking you to code this whole thing from scratch seems a bit insensitive to your time, particularly for those of you who are not planning to further pursue macro. So I am providing `Julia` code with most of the "hard" parts of the process done, and **you just have to fill in a few crucial functions** which are used to solve the household and firm problems. If you are working in `Julia`, this will be easy: just fill in the missing bits. If you choose to work in `Python`, you will then need to translate the remainder of my code into `Python` as well[1], in order to get the results at the end (Again, you don't have to actually code that much, so probably the least effort approach is to just learn enough of `Julia` to fill in my code). As a final note, you are completely free to use as little or as much of my code as you want to get results, regardless of the language you use[2]. If you have a better way to write the model (which certainly exists) and want to give it a try, go for it!

2 Grading

Again, this is not meant to be a time-consuming assignment. Focus on understanding the model and how the algorithms work more than the exact mechanics of making the code perfect. I don't anticipate you spending more than a handful of hours on this, unless you want to dig deeper and learn more (in which case you can spend endless hours playing with it). So mostly this will be graded on completion/effort[3], since the actual coding does not strictly fall under the Core curriculum.

*Send questions/corrections to abram@uchicago.edu.

[1] Since I am giving the `Julia` version, I also do not care if you collaborate in writing the parts already given to you for `Python`.

[2] But you can't grab other code that is not your own. No using the QuantEcon model!

[3] And can hopefully provide a tiny bit of grade buffer stock (meta-joke intended).

3 Setup

Our version of the (Aiyagari, 1994) model will not look exactly like the one in his paper, but it is meant to be slightly simpler to follow. Please note this, as it will slightly change some of the equations from the paper (and final supply/demand diagram).

3.1 Households

We assume a continuum of households each maximizing their own present discounted utility. They rent out their capital to firms, which pay r, and their capital depreciates at rate δ. They also supply one unit of labor inelastically to firms at wage w, but they draw labor productivity z each period, so their effective labor is z and labor income is wz. Households' savings behaviors are liquidity-constrained by $a_t \geq 0$ (we could choose the "natural" borrowing limit instead, but this is simpler and we don't lose any of the insight). So they solve

$$\max \sum_{t=0}^{\infty} \beta^t u(c_t)$$
$$c_t + a_{t+1} \leq r a_t + (1-\delta) a_t + w z_t$$

Let $Q(z' \mid z)$ denote the measure describing the probability of drawing z' given the current draw is z, so that in principle the labor productivity process may be Markovian. We will assume this is a stationary process, otherwise things get wonky. Then the Bellman equation for the household is

$$\begin{aligned}
V(a, z) &= \max_{a' \in \Gamma(a,z)} \left\{ u(ra + (1-\delta)a + wz - a') + \beta \int V(a', z') \mathrm{d}Q(z' \mid z) \right\} \\
\Gamma(a, z) &= [0, ra + (1-\delta)a + wz]
\end{aligned}$$

3.2 Firms

Firms[4] produce output for a unit price, and decide how much capital and labor to demand. Their technology is CRS

$$F(K, N) = K^{\alpha} N^{1-\alpha}$$

so capital and labor demand are summarized by

$$r = \alpha K^{\alpha-1} N^{\alpha-1}$$
$$w = (1-\alpha) K^{\alpha} N^{-\alpha}$$

[4] We may consider a representative firm if we wish. It makes no difference since tech is CRS.

3.3 Equilibrium

Our problem is stationary, because the stochastic process is stationary. It is also recursive, because the problem today can be written in terms of the problem tomorrow. We are also letting individuals make their own choices, so everything will be competitive. Thus our relevant equilibrium concept is a **stationary recursive competitive equilibrium.**

Both households and firms take the wage and rental rate as given. Hence an equilibrium will be a set $\{r, w, N, K\}$ (we could also include household choice $g(a, z)$ and implied stationary distribution $\Phi(a, z)$) such that

(i) Given r and w, households solve their problem (their Bellman equation holds)

(ii) Given r and w, firms choose K and N

(iii) Aggregate household choices for factor supply equal firm demand:

$$K = \int a \mathrm{d}\Phi(a, z)$$
$$N = \int z \mathrm{d}\Phi(a, z)$$

where Φ is the implied stationary distribution over assets and labor productivity implied by the solution to the households' problem.

4 Preliminary Maths

(a) Where is the heterogeneity in the model? (One line)

(b) In the market for capital, which side is more difficult to calculate, capital supply or capital demand? (One line)

(c) Why do households save in this model? How much do households save if we set Q such that z is constant for all time? [Hint: You might want to consider Deaton (1991) or Deaton and Laroque (1992).]

(d) Predict: Will households tend to supply more or less capital (save more or less) if the labor productivity process is more or less persistent[5]? (One or two lines)

5 Coding

No need to write anything for this part. All that is required is completing the code segments.

(a) Complete `update_V!(V,Vnew,g,A,Z,Zmat,r,w,beta,delta)`, which updated `Vnew` as the output from a single Bellman iteration on `V`. `V` is the value function over the discretized domain of $n_a \times n_z$ possible states, `Vnew` is the value function being updated, `g` is the households' asset choice policy (maps the coordinates of an asset and shock pair to the coordinate of an asset choice), `A` is the asset domain, `Z` is the labor productivity domain, `Zmat` is the transition matrix for the productivity process, `r` is the rental rate, `w` is the wage rate, `beta` is the discount rate, and `delta` is the depreciation rate. Use $u(c) \equiv \ln(c)$.

[5] A random walk is completely persistent, an iid process is not at all persistent

(b) Complete `state_dist_update!(g, state_dist,state_distnew, Zmat)`, which updates `state_distnew` using a given distribution `state_dist`, policy `g`, and shock transition matrix `Zmat`.

(c) Complete `agg_K`, `agg_N`, `rental`, `wage`, and `demand_K`.

6 Code Interpretation

No need to do any coding here, just show that you can understand what each algorithm is doing.

(a) How does `supply_K` work?

(b) How does `steady_state` work? In particular, how is the bisection updating working? Why not just update directly, instead of bisecting?

7 Results

(a) Provide the output equilibrium variables and plots at the bottom of the code for each of the three parameterizations of `Zmat`.

(b) What is special with regards to the persistence of the first `Zmat` parameterization? How does it compare to the other two parameterizations? (One or two lines)

(c) Consider the outputs from the three steady-state runs. What do you conclude about how the persistence of the shock process translates into equilibrium rental rate and capital? What is the economic interpretation?

(d) The plots seem to go wild/vertical when r is sufficiently high. Can you interpret what makes this r special, and why this jump to the upper boundary of our asset domain occurs?

References

Aiyagari, Rao S. (1994). "Uninsured idiosyncratic risk and aggregate saving". In: *The Quarterly Journal of Economics* 109.3, pp. 659–684.