LSE Macroeconomics Summer Program
Part II: Heterogeneous Agents
Instructor: Wouter J. Den Haan

Tuesday Assignment
Solving a heterogeneous agent model using the KS-algorithm

Goal

In this assignment, we solve a model that is similar to the one in yesterday's assignment *except* that there is not only idiosyncratic, but also aggregate productivity shocks. The objective is to solve this model. The presence of aggregate shocks makes the problem a lot more difficult because now the cross-sectional distribution is time-varying which means that we have a (time-varying) state variable that is an infinite-dimensional object.

Overview

The model equations are given at the end of this document. The model is solved using the algorithm of Krusell and Smith (1998), which consists of three steps:

1. Start with the following law of motion for the capital stock:

$$K_t^a = b_0 + b_K K_{t-1}^a + b_Z \left(z_t - z\right), \tag{1}$$

   where K_t^a is the (end of period t) aggregate capital stock, z_t is productivity and z is the steady-state level of productivity.
   Given this perceived law of motion, the problem of the individuals can be solved with the same type of algorithms used to solve the corresponding partial equilibrium model. The only difference is that we have extra state variables, namely K_{t-1}^a and z_t. To simplify the programming part somewhat, we use 2nd-order perturbation to solve the individual problem. Given that there is typically quite a bit of uncertainty in models with idiosyncratic uncertainty, this may not always lead to accurate solutions. In that case it would be better to use a projection method or possibly a higher-order perturbation solution.[1] But for the structure of the complete algorithm it doesn't matter how one solves the individual problem, so we choose an algorithm that ensures you can finish the assignment relatively quickly.

2. Given the policy rules of the individuals obtained in step 1, conduct a Monte Carlo simulation for a panel of I agents. Each period aggregate the individual capital stocks to obtain a time series for aggregate capital K_t^a.

3. Run a regression to update the coefficients of the law of motion of K_t^a given in Equation (**??**).

[1] On Thursday we will show that higher-order perturbation solution may be problematic in terms of having odd shapes away from the steady state.

4. Go back to step 1 until the law of motion obtained in step #3 is (basically) the same as the law of motion used in step #1.

Thus, in step #1 we calculate the law of motion for the individual capital stock that is implied by the law of motion for K_t^a given in Equation (**??**). In steps #2 and #3, we take the individual policy rules as given and calculate the implied law of motion for K_t^a.

Assignment

The algorithm is partly programmed in the file `solve_model_agg_uncertainty.m`, which calls Dynare to solve the problem of the individuals (second-order perturbation). The Dynare file containing the individual problem is called `model_agg_uncertainty.mod`.

Main part of the assignment.

1. Complete the model part in `model_agg_uncertainty.mod`
2. Complete the missing parts in `solve_model_agg_uncertainty.m` indicated with XXX.

Additional exercise: Comparison with alternative solution & model

1. Next, write a program that solves the same model, but now using first-order perturbation instead of second-order.
2. Next, write a dynare program that solves a version of the model without individual uncertainty.
3. Compare properties of the outcomes of step #2, #3, & #4. That is the two numerical solutions to the model with aggregate and idiosyncratic uncertainty and the model without idiosyncratic uncertainty. In particular, look at
   (a) the volatility of individual consumption. Is consumption a lot more volatile in the economy with idiosyncratic risk?
   (b) what is the mean level of *aggregate* capital? Do agents build a bigger buffer stock of capital in the economy with idiosyncratic risk?
   (c) calculate the the IRF of K_t^a in the three economies.

Using Dynare Policy Rule Coefficients

Dynare stores the coefficients of the policy rules inside the oo_ structure. Unfortunately, what is stored is not exactly equal to the more natural representation that Dynare displays on the screen; for example, the ordering may be different and for second-order some coefficents are multiplied by 1/2). To help you out, we have provided a program "get_policy_rule_coefs.m" that will put the policy rule coefficients in a matrix "decision" in exactly the order that Dynare writes them to the screen. In principle, there is nothing you have to do in the program, but take a look at what this program does anyway.

Some tips on get_policy_rule_coefs.m

- As you can see, you have to set a couple variables before running the program (like the perturbation order).
- This program allows YOU to choose which variables to put into the Matlab matrix decision and in which order.
- So if you list k as the first variable, then its policy rule coefficients will appear in the first column of decision. This may or may not be different from where Dynare puts it.
- Since there are not that many variables in this program, you may want to use the same ordering as Dynare has chosen. But it is up to you. Since you only need capital, you may also choose to only list k, so that the matrix decision will be just a column vector.
- Alternatively, you can use the function provided to generate the matrix decision. This does not change anything regarding the use of global memory.

!!!! Important comment about the global statement The variables created by Dynare like oo_ are only available in the main program. This means that they will not be available inside a function EVEN if Dynare is run inside that function. This is why you need to add global M_ oo_ at the beginning of both the main program and the function in which you run Dynare. If you run Dynare in the main program, then you would not need this global statement.

Model

Rental rate on capital and wage rate:

$$\begin{aligned} r_t &= \alpha z_t \left(K^a_{t-1}\right)^{\alpha-1}, \\ w_t &= (1-\alpha) z_t \left(K^a_{t-1}\right)^{\alpha}. \end{aligned}$$

Euler equation individual agent:

$$\frac{1}{c_{i,t}} + \zeta_2 - \zeta_1 \exp\left(-\zeta_0 k_{i,t}\right) = E_t\left[\frac{\beta}{c_{i,t+1}}\left[r_{t+1} + 1 - \delta\right]\right],$$

where $\zeta_2 = \zeta_1 \exp\left(-\zeta_0 \overline{k}\right)$ and $\overline{k}$ is the steady state level of individual capital holdings. Budget constraint individual agent:

$$c_{i,t} + k_{i,t} = r_t k_{t-1} + w\left(1 + e_{1,i,t}\right) + (1-\delta) k_{i,t-1}.$$

Productivity process:

$$z_t = 1 - \rho + \rho z_{t-1} + e_{2,t}.$$

Innovations:

$$\begin{aligned} e_{1,i,t} &\sim N(0, \sigma_1), \\ e_{2,t} &\sim N(0, \sigma_2). \end{aligned}$$