LSE Macroeconomics Summer Program

Part I: The Essentials

Instructor: Wouter J. Den Haan

Tuesday Assignment

Solving a model with time-varying risk premium with projection methods

1 Goal

The goal of this assignment is to show that solving a model with projection methods can actually be quite simple. There is only one somewhat tricky step, namely evaluating the conditional expectation using numerical integration.

2 Model

The solution to the Lucas Asset Pricing Model is characterized as follows

$$\begin{aligned} p_t &= \beta \mathrm{E}_t \left(\left(\frac{c_{t+1}}{c_t} \right)^{-\gamma} (d_{t+1} + p_{t+1}) \right) \\ c_t &= d_t \\ d_t &= \mu_d + \rho_d d_{t-1} + \varepsilon_t \\ \varepsilon_t &\sim N\left(0, \sigma^2\right) \end{aligned}$$

Of course, you can choose your own parameters, but to make our programs comparable consider

Parameter	μ_d	ρ_d	σ	β	γ
Value	0.1	0.9	0.1	0.9	3.

3 Preliminaries

3.1 Approximating function

You always have to take a stand on which function you want to approximate and with what kind of approximating function. In this assignment, you should approximate the price function using the nth-order polynomial

$$p_t(d_t) = \sum_{n=0}^{N} a_n d_t^n.$$

The Matlab function pfunc.m, included with the assignment, defines this function.

3.2 Grid

With projection methods, you have to create a grid for the state variables. In the Lucas Asset Pricing Model, d_t is the only state variable and we can calculate the mean and standard deviation. In this assignment, we construct a grid for d_t ranging from three standard deviations below the mean to three standard deviations above the mean.

3.3 Numerical integration

The tricky step is to evaluate the conditional expectation using numerical integration. But don't be scared of this. Just realize the following:

- The stochastic variable ε_{t+1} shows up three times inside the conditional expecation. Once because d_{t+1} depends on ε_{t+1}, once because c_{t+1} depends on d_{t+1} which in turn depends on ε_{t+1}, and once because p_{t+1} depends on d_{t+1} which in turn depends on ε_{t+1}.

- Numerical integration basically comes down to thinking of ε_{t+1} as a random variable with discrete

support with each realization having a probability. Remember from the slides that if $x \sim N\left(\mu, \sigma^2\right)$,

$$\mathrm{E}\left(h\left(x\right)\right) \approx \sum_{j=1}^{gh} \left(\frac{\omega_j}{\sqrt{\pi}} h\left(\mu + \sigma\sqrt{2}\zeta_j\right) \right)$$

where ω_j and ζ_j are the Gaussian Hermite weights, sort of like probabilities and nodes respectively.

4 Exercise 1: solve for p

1. Complete the external function that calculates the sum of squared Euler equation errors. This is found in the included Matlab file `errfunc.m`.

2. The main projection program is given in the file main.m. Understand the structure of this file. Note that it really does not do much more than construct a grid and call a minimization routine. Run this projection algorithm. Now you have solved your first projection problem!

3. Figure 1 plots the errors on the grid (when you are done you may want to play around a bit with grid size, polynomial order, etc to see how errors change).

4. Plot the policy function. Hint: you can use the pfunc.m function.

   !!! The coefficients used as input in pfunc.m must be ordered in the right way. The first element is the element corresponding to the highest order and the last element is the constant.

5 Risk premium and expected returns

The risk premium x_t is the difference between the expected rate of return $\mathrm{E}_t(r_{t+1})$ and the risk-free rate r_t^f. The two rates and the premium are given by the following equations

$$\begin{aligned} 1 &= \left(1 + r_t^f\right) \beta \mathrm{E}_t \left(\left(\frac{c_{t+1}}{c_t} \right)^{-\gamma} \right) \\ r_{t+1} &= \frac{d_{t+1} + p_{t+1}}{p_t} - 1 \\ x_t &= \mathrm{E}_t(r_{t+1}) - r_t^f. \end{aligned}$$

We could have solved for r_t^f, $\mathrm{E}_t(r_{t+1})$, and x_t simultaneously with p_t. But given the solution for p_t, it is very easy to solve for these objects. You are asked to approximate the expected risk premium function using the N^{th}-order polynomial

$$x_t(d_t) = \sum_{n=0}^{N} b_n d_t^n.$$

The Matlab function `xfunc.m` defines this function.

Before doing any calculations ask yourself whether r_t^f, $\mathrm{E}_t(r_{t+1})$, and x_t are increasing or decreasing in d_t.

6 Exercise 2: solve for expected returns

1. To solve for the values of b_n, i.e. the coefficients of the approximating function for the expected risk premium, you do *not* have to use a minimization routine. You can simply implement the following two steps:

   (a) Calculate r_t^f, $\mathrm{E}_t(r_{t+1})$, and x_t at each grid point, where r_t^f is defined by

$$r_t^f = \frac{1}{\beta \mathrm{E}_t \left(\left(\frac{c_{t+1}}{c_t} \right)^{-\gamma} \right)} - 1 \text{ and}$$

(b) find the coefficients b_n by running a projection (regression).

2. Plot the three objects as a function of d. Do the results correspond to what you expected?

3. Next, plot time series for the three objects to evaluate how much time-variation there is in these three objects.

4. How would the simulation for the risk premium look like if you would use first-order perturbation? And if you would use second-order perturbation?