Macroeconomics III

Value Function Iteration in Stochastic Environment

Diego Rodrigues

SciencesPo
diego.desousarodrigues@sciencespo.fr

Fall 2023

Stochastic Dynamic Programming

Suppose consumer maximizes

$$V\left(k_0, z_0\right) = \max_{\{c_t, k_{t+1}\}_{t=0}^{\infty}} = E_0\left[\sum_{t=0}^{\infty} \beta^t u\left(c_t\right)\right], 0 < \beta < 1$$

s.t.

$$k_{t+1} + c_t \leq z_t f\left(k_t\right)$$

$$z_{t+1} = \rho z_t + \epsilon_{t+1} \quad , \rho \in (0,1), \epsilon \sim F\left(0, \sigma_\epsilon^2\right)$$

$$k_0 > 0 \quad \text{given and ,}$$

$$c_t \geq 0$$

- Note: u(.) and f(.) satisfy standard properties.
- At period t agents know the realization of the shock z_t.

Optimal Growth Problem with Uncertainty

- We can show that:

$$V\left(k_0, z_0\right) = \max_{0 \leq k_1 \leq z_0 f(k_0)} \left\{u\left(z_0 f\left(k_0\right) - k_1\right) + \beta E_0\left[V\left(k_1, z_1\right)\right]\right\}$$

Optimal Growth Problem with Uncertainty

- We can show that:

$$V\left(k_0, z_0\right) = \max_{0 \leq k_1 \leq z_0 f(k_0)} \left\{u\left(z_0 f\left(k_0\right) - k_1\right) + \beta E_0\left[V\left(k_1, z_1\right)\right]\right\}$$

- Since the problem repeats itself in every period we have:

$$V(k, z) = \max_{0 \leq k' \leq z f(k)} \left\{u\left(z f(k) - k'\right) + \beta E\left[V\left(k', z'\right)\right]\right\}$$

- We can show that:

$$V\left(k_0, z_0\right) = \max_{0 \leq k_1 \leq z_0 f(k_0)} \left\{u\left(z_0 f\left(k_0\right) - k_1\right) + \beta E_0\left[V\left(k_1, z_1\right)\right]\right\}$$

- Since the problem repeats itself in every period we have:

$$V(k, z) = \max_{0 \leq k' \leq z f(k)} \left\{u\left(z f(k) - k'\right) + \beta E\left[V\left(k', z'\right)\right]\right\}$$

- Notice that z is not our choice variable, since it is exogenous. So, we again have the **stochastic version of the Euler Equation**:

$$u'\left(c_t\right) = \beta E\left[u'\left(c_{t+1}\right)\left(1 + z_{t+1} f'\left(k_{t+1}\right) - \delta\right)\right]$$

Markov Process

Recall we have the following definition for a Markov Process:

Markov Process

A stochastic variable z_t follows a **first-order Markov chain** if for all $k \geq 1$ for each $i = 1, ..., n$:

$$\text{Prob}\left[z_t = \hat{z}/z_{t-1}, z_{t-2}, \ldots, z_{t-k}\right] = \text{Prob}\left[z_t = \hat{z}/z_{t-1}\right]$$

We can then characterize a $n-$ dimensional Markov Process by the state space $z \in \mathcal{Z} = \{z_1, ..., z_n\}$ and by the nXn transition matrix P, where:

$$P_{ij} = \text{Prob}\left[z_{t+1} = z_j/z_t = z_i\right], \text{ notice that } \sum_{j=1}^{n} P_{ij} = 1.$$

Model with uncertainty

Now consider the model is given by;

$$
\begin{aligned}
V\left(k_0\right) &= \max E_0 \sum_{t=0}^{\infty} \beta^t \ln\left(c_t\right) \\
k_{t+1} &= A z_t k_t^{\alpha} - c_t \\
k_0 &: \text{ given} \\
z_t &: \text{ stochastic shock}
\end{aligned}
$$

Model with uncertainty

Now consider the model is given by;

$$
\begin{aligned}
V\left(k_0\right) &= \max E_0 \sum_{t=0}^{\infty} \beta^t \ln\left(c_t\right) \\
k_{t+1} &= A z_t k_t^{\alpha} - c_t \\
k_0 &: \text{ given} \\
z_t &: \text{ stochastic shock}
\end{aligned}
$$

There are **two state variables** in this model: k_t and z_t.

Model with uncertainty

Now consider the model is given by;

$$V(k_0) = \max E_0 \sum_{t=0}^{\infty} \beta^t \ln(c_t)$$

$$k_{t+1} = A z_t k_t^{\alpha} - c_t$$

$$k_0 : \text{ given}$$

$$z_t : \text{ stochastic shock}$$

There are **two state variables** in this model: k_t and z_t.
What is z_t?

- It can be an **i.i.d. shock**.
- It can be a **Markov chain**.
- It can be an **autoregressive process** (in this case there is need to approximate).

Computing the model using a Markov Chain

Assume

$$z_t = \begin{cases} z_1 \\ z_2 \end{cases},$$

with transition matrix

$$\Pi = \begin{bmatrix} \pi_{11} & \pi_{12} \\ \pi_{21} & \pi_{22} \end{bmatrix},$$

where $\pi_{i1} + \pi_{i2} = 1$ (the row sum up to one).

In what follows we will discuss the steps to solve for the Value Function Iteration.

Algorithm to Value Function Iteration

1. Define the parameters, get the steady state and make the grid (around the steady state).

Algorithm to Value Function Iteration

1. Define the parameters, get the steady state and make the grid (around the steady state).
2. Use a method (i.e., Tauchen method) to obtain the values of z and the transition matrix Π.

Algorithm to Value Function Iteration

1. Define the parameters, get the steady state and make the grid (around the steady state).
2. Use a method (i.e., Tauchen method) to obtain the values of z and the transition matrix Π.
3. Compute 2 matrices for consumption, conditional on the shock, i.e., $C_i = Az_ik^{\alpha} - k^{'}$.

Algorithm to Value Function Iteration

1. Define the parameters, get the steady state and make the grid (around the steady state).
2. Use a method (i.e., Tauchen method) to obtain the values of z and the transition matrix Π.
3. Compute 2 matrices for consumption, conditional on the shock, i.e., $C_i = Az_i k^{\alpha} - k^{'}$.
4. Compute the first value functions like before, by using:

$$TV_1 = (\max \{U_1 + \beta * \text{zeros}(m, 1) * \text{ ones } (1, m)\})'$$
$$TV_2 = (\max \{U_2 + \beta * \text{zeros}(m, 1) * \text{ ones } (1, m)\})'$$

Algorithm to Value Function Iteration

1. Define the parameters, get the steady state and make the grid (around the steady state).
2. Use a method (i.e., Tauchen method) to obtain the values of z and the transition matrix Π.
3. Compute 2 matrices for consumption, conditional on the shock, i.e., $C_i = Az_i k^\alpha - k'$.
4. Compute the first value functions like before, by using:

$$TV_1 = (\max\{U_1 + \beta * \text{zeros}(m,1) * \text{ ones } (1,m)\})'$$
$$TV_2 = (\max\{U_2 + \beta * \text{zeros}(m,1) * \text{ ones } (1,m)\})'$$

5. Two stopping criteria: while check1 $>$ 0.0001 OR check2 $>$ 0.0001:

$$TV_1 = (\max\{U_1 + \beta * (\pi_{11}V_1 + \pi_{12}V_2) * \text{ ones } (1,m)\})'$$
$$TV_2 = (\max\{U_2 + \beta * (\pi_{21}V_1 + \pi_{22}V_2) * \text{ ones } (1,m)\})'$$

Algorithm to Value Function Iteration

6. Compute two policy functions (one for each state z).

Algorithm to Value Function Iteration

6. Compute two policy functions (one for each state z).
7. Simulate the economy many times to get statistics.

Algorithm to Value Function Iteration

6. Compute two policy functions (one for each state z).
7. Simulate the economy many times to get statistics.

- Notice that to simulate, we will need to create a function that generates randomly the two states;
- This can be generated easily by two states, but it can be challenge in case we have a lot of them.

Additional steps

- **Check that the policy functions is not constrained by the discrete state space.** If k' is equal to the highest or the lowest value of capital in the grid for some i, relax the bounds of k and redo the value function iteration.
- **Check the error tolerance is small enough.** If a small reduction in the tolerance level leads to large changes in the value function or in the policy function, then the tolerance is too high.
- **Check whether or not the grid is large enough.** If a change in the grid leads to a substantial difference in the result maybe the grid is too sparse.
- **A good initial guess for the value function can reduce the computational time.**

Value Function Iteration: some comments

1. **Advantages.**
   - Always work. It is stable, so that it converges to the true solution.
2. **Disadvantages.**
   - It is slow. It suffers form the "curse of dimensionality".
   - The main problem in the value function iteration is the maximization part.

Howard's Improvement

Idea

Decision rules converge faster than value functions and maximization is time-consuming. We keep the policy function fixed.

Howard's Improvement

Idea

Decision rules converge faster than value functions and maximization is time-consuming. We keep the policy function fixed.

Howard's improvement reduces the number of times we update the policy function relative to the number of times we update the value function.

1. Guess V_0; then use the operator:

$$V_1 = T[V_0] = \max_{0 \leq k' \leq f(k)} \left\{ u\left(f(k) - k'\right) + \beta V_0\left(k'\right) \right\}$$

   to find V_1 and $k' = g(k)$.

   - Then for some finite $n_h \in \{1, 2, ..., N_h\}$ iterate:

$$V_1^{n_h} = u(f(k) - h(k)) + \beta V_1^{n_h - 1}(g(k))$$

   .

2. Check if $V_0 \approx V_1^{N_h}$.
3. If not, repeat (1) until $V_n \approx V_{n-1}^{N_h}$.

Value Function Iteration: code

NOW WE WILL SEE TOGETHER A CODE SOLVING THE VALUE FUNCTION ITERATION IN A STOCHASTIC ENVIRONMENT