Global Solutions and Income Fluctuation Problem

Chase Coleman

July 6, 2017

NYU Stern

Introduction to Income Fluctuation Problem

Income Fluctuation Problem (IFP)

Foundation for modern heterogeneous agent macro

Partial equilibrium model of a single consumer's problem

- Wages exogenously given
- Asset return exogenously given

Income Fluctuation Problem (IFP)

Single consumer with preferences

$$\max_{\{c_t(s^t)\}_{t=0}^{\infty}} E_0 \left[\sum_{t=0} \beta^t u(c_t(s^t)) \right]$$

Markov stochastic process, $\{s_t\}$

Receives stochastic income each period, $y(s_t)$

Income can either be

- Consumed, $c_t(s^t)$ or
- Saved in risk-free asset, $a_t(s^t)$, with return r

Recursive Formulation

Problem can be "Bellmanized" with (a_t, s_t) as the state variables

$$V(a_t, s_t) = \max_{a_{t+1}} u(c_t) + \beta E\left[V(a_{t+1}, s_{t+1}) | s_t\right]$$

Subject to

$$c_t = (1+r)a_t + y(s_t) - a_{t+1}$$

Solving the IFP

Solving the IFP

What does it mean to "solve" the IFP?

Want to characterize

- Policy function: $a^*(a_t, s_t)$
- Value function: $V(a_t, s_t)$

Three Iterative Algorithms

Will introduce three global solution methods.

All three are variants on value function iteration

1. Discretized Value Function Iteration
2. Interpolated Value Function Iteration
3. Envelope Condition Method

Discretized Value Function Iteration

1. Choose grids on state variables: $\mathcal{A} \times \mathcal{S}$
2. Initial guess of value function: $V_0(a,s)\ \forall(a,s) \in \mathcal{A} \times \mathcal{S}$
3. Given V_j, for each $(a_t, s_t) \in \mathcal{A} \times \mathcal{S}$
    3.1 Find $a_{t+1} \in \mathcal{A}$[1] such that

    $$a_{t+1} \in \arg\max u(y(s_t) + (1+r)a_t - a_{t+1}) + \beta E\left[V_j(a_{t+1}, s_{t+1})\right]$$

    3.2 Update policy function

    $$a^*(a_t, s_t) = a_{t+1}$$

    3.3 Update value function

    $$V_{j+1}(a_t, s_t) = u(c_t) + \beta E\left[V_j(a_{t+1}, s_{t+1})\right]$$

4. If $d(V_{j+1}, V_j) < \varepsilon$ then done, otherwise return to 3

[1] Just evaluate at each grid point and choose the max. Very computationally inexpensive

Interpolated Value Function Iteration

1. Choose grids on state variables: $\mathcal{A} \times \mathcal{S}$
2. Initial guess of value function: $V_0(a, s)\ \forall (a, s) \in \mathcal{A} \times \mathcal{S}$
3. Create functional approximation of V_j using interpolation routines: $\hat{V}_j$
4. Given $\hat{V}_j$, for each $(a_t, s_t) \in \mathcal{A} \times \mathcal{S}$
   4.1 Using optimization routines, find
   $$a_{t+1} \in \arg\max u(y(s_t) + (1 + r)a_t - a_{t+1}) + \beta E\left[\hat{V}_j(a_{t+1}, s_{t+1})\right]$$
   4.2 Update policy function
   $$a^*(a_t, s_t) = a_{t+1}$$
   4.3 Update value function
   $$V_{j+1}(a_t, s_t) = u(c_t) + \beta E\left[\hat{V}_j(a_{t+1}, s_{t+1})\right]$$
5. If $d(V_{j+1}, V_j) < \varepsilon$ then done, otherwise return to 3

Envelope Condition Method

1. Choose grids on state variables: $\mathcal{A} \times \mathcal{S}$
2. Initial guess of value function: $V_0(a, s)\ \forall (a, s) \in \mathcal{A} \times \mathcal{S}$
3. Create functional approximation of V_j using interpolation routines: $\hat{V}_j$
4. Given $\hat{V}_j$, for each $(a_t, s_t) \in \mathcal{A} \times \mathcal{S}$
    4.1 Use envelope condition to get $c_t^* = (u')^{-1}\left(\frac{\partial \hat{V}_j(a_t, s_t)/\partial a_t}{1+r}\right)$
    4.2 Update policy function

$$a^*(a_t, s_t) = y(s_t) + (1+r)a_t - c_t^*$$

    4.3 Update value function

$$V_{j+1}(a_t, s_t) = u(c_t^*) + \beta E\left[\hat{V}_j(a_{t+1}, s_{t+1})\right]$$

5. If $d(V_{j+1}, V_j) < \varepsilon$ then done, otherwise return to 3

Comparisons

Thoughts on different algorithms:

- Discretized VFI:
  - Pros: Easy to code
  - Cons: Infeasible for big problems
- Interpolated VFI:
  - Pros: Very general, solves slightly bigger problems
  - Cons: Optimization routines are a drag in terms of speed, infeasible for big problems
- Envelope Condition Method:
  - Pros: Very fast, frequently can be used in some other models to get a "closed form" for certain choice variables
  - Cons: Careful that derivatives are sufficiently nice, does not work for all models

Example

Example

Will walk through example of discretized value function together

Practice problem will require solving a perturbation (or two) of the income fluctuation problem using other algorithms