TA Session: Computational Dynamic Programming

Econ 30400: Mathematical Methods for Economics

Levi Crews (Chicago)

September 2020

A simple consumption-saving problem

An agent with CRRA utility and an initial wealth endowment w_0 maximizes the lifetime value of her consumption stream $\{c_t\}$ discounted at rate β subject to an exogenous interest rate r:

$$v(w_0) = \max_{\{c_t\}} \sum_{t=0}^{\infty} \beta^t \frac{c_t^{1-\sigma}}{1-\sigma} \quad \text{s.t. } w_{t+1} = (1+r)(w_t - c_t)$$

Let's put this in recursive form

Sequential

$$v(w_0) = \max_{\{c_t\}} \sum_{t=0}^{\infty} \beta^t \frac{c_t^{1-\sigma}}{1-\sigma} \quad \text{s.t.} \;\; w_{t+1} = (1+r)(w_t - c_t)$$

Recursive

$$v(w) = \max_{c} \frac{c^{1-\sigma}}{1-\sigma} + \beta v(w') \quad \text{s.t.} \;\; w' = (1+r)(w - c)$$

This problem has a closed-form solution

Recall that a **solution** here is

- a **value function** $v(\cdot)$ that satisfies the Bellman equation, and
- a **policy function** $c(\cdot)$ that attains the maximum,

where both are functions of the state variable w.

We can solve this particular problem by the guess-and-verify method:

$$v(w) = v_0 w^{1-\sigma}$$
$$c(w) = c_0 w,$$

where

$$v_0 = \frac{(1+r)(1-c_0)}{\beta(1+r)(1-\sigma)c_0}$$
$$c_0 = 1 - \frac{(\beta(1+r))^{1/\sigma}}{1+r}.$$

This problem has a closed-form solution

Recall that a **solution** here is

- a **value function** $v(\cdot)$ that satisfies the Bellman equation, and
- a **policy function** $c(\cdot)$ that attains the maximum,

where both are functions of the state variable w.

We can solve this particular problem by the guess-and-verify method:

$$v(w) = v_0 w^{1-\sigma}$$
$$c(w) = c_0 w,$$

where

$$v_0 = \frac{(1+r)(1-c_0)}{\beta(1+r)(1-\sigma)c_0}$$
$$c_0 = 1 - \frac{(\beta(1+r))^{1/\sigma}}{1+r}.$$

But what if there's no closed-form solution?

By the **contraction mapping theorem**, we can compute v by **successive approximations**: the sequence

$$\{v_k\} \text{ s.t. } \begin{aligned} v_{k+1} &= Tv_k \\ v_0 &\in C(\mathbb{R}) \\ T &\equiv \text{ Bellman operator} \end{aligned}$$

converges to v at rate β

Four types of approximations:

1. Value function iteration
   a. exogenous grid search
   b. interpolation
2. Policy function iteration
   a. exogenous grid search
   b. endogenous grid search

But what if there's no closed-form solution?

By the **contraction mapping theorem**, we can compute v by **successive approximations**: the sequence

$$\{v_k\} \text{ s.t. } v_{k+1} = Tv_k$$
$$v_0 \in C(\mathbb{R})$$
$$T \equiv \text{ Bellman operator}$$

converges to v at rate β

Four types of approximations:

1. Value function iteration
   a. exogenous grid search
   b. interpolation
2. Policy function iteration
   a. exogenous grid search
   b. endogenous grid search

1a. Value function iteration (exogenous grid)

1. Discretize the state space: $W \equiv [w_1, \ldots, w_N]'$ for some large N
2. Guess any $v_0 = [v_0(w_1), \ldots, v_0(w_N)]'$ (**why? but are some better?**)
3. Compute $u(w_n, w_m) \equiv$ payoff when current state is w_n and next state is w_m
4. Compute $v_{k+1}(w_n) \equiv \max_{w_m \in W}[u(w_n, w_m) + \beta v_k(w_m)]$ for each $w_n \in W$
5. Stop when $||v_{k+1} - v_k|| < \epsilon$ for some tolerance level ϵ

Why it works: follows directly from contraction mapping theorem

Pros: easy to implement, fast for low-dim state space and small N

Cons: slow for high-dim state space and large N

1b. Value function iteration (interpolation)

1. Discretize the state space: $W \equiv [w_1, \ldots, w_N]'$ for some large N
2. Guess any $v_0 = [v_0(w_1), \ldots, v_0(w_N)]'$
3. Compute the function $\hat{v}_k$ by interpolating $\{v_k(w_n)\}_{w_n \in W}$
4. Compute $v_{k+1}(w_n) \equiv \max_c [u(c) + \beta \hat{v}_k((1+r)(w_n - c))]$ for each $w_n \in W$
5. Stop when $||v_{k+1} - v_k|| < \epsilon$ for some tolerance level ϵ

Why it works: follows directly from contraction mapping theorem

Pros: faster than grid search for high-dim state space and large N (**why?**)

Cons: extra work to interpolate

2a. Policy function iteration (exogenous grid)

1. Discretize the state space: $W \equiv [w_1, \ldots, w_N]'$ for some large N
2. Guess any $c_0 = [c_0(w_1), \ldots, c_0(w_N)]'$ (**why? but are some better?**)
3. Compute the function $\hat{c}_k$ by interpolating $\{c_k(w_n)\}_{w_n \in W}$
4. Compute $c_{k+1}(w_n) \equiv \arg_c\{u'(c) - \beta(1+r)u'[\hat{c}_k((1+r)(w_n - c))] = 0\}$
5. Stop when $||c_{k+1} - c_k|| < \epsilon$ for some tolerance level ϵ

Why it works: envelope theorem, contraction mapping theorem

Pros: faster than value function iteration

Cons: smaller scope (need concavity, differentiability)

2b. Policy function iteration (endogenous grid)

1. Discretize the state space: $W \equiv [w_1, \ldots, w_N]'$ for some large N
2. Guess any $c_0 = [c_0(w_1), \ldots, c_0(w_N)]'$
3. Compute $c_{k+1}(\tilde{w}_n) = (u')^{-1}[\beta(1+r)u'(c_k(w_n))]$ and $\tilde{w}_n = \frac{w_n}{1+r} + c_{k+1}(\tilde{w}_n)$
4. Interpolate values $\{c_{k+1}(w_n)\}_{w_n \in W}$
5. Stop when $||c_{k+1} - c_k|| < \epsilon$ for some tolerance level ϵ

Why it works: as above + Euler equation, budget constraint

$$c = (u')^{-1}[\beta(1+r)u'(c')], \qquad w' = (1+r)(w-c)$$

Pros: faster than exogenous grid because it's slow to solve for $c_{k+1}(w_n)$

Cons: even smaller scope (need u' invertible, too)