Practical Dynamic Programming: An Introduction

Associated programs

dpexample.m: deterministic

dpexample2.m: stochastic

Outline

1. Specific problem: stochastic model of accumulation from a DP perspective
2. Value Function Iteration: An Introduction
3. Value Function Iteration: An Example
4. Value Function Iteration: Discretization
5. The curse of dimensionality
6. The importance of “initial conditions”
7. Adding uncertainty

1. Stochastic growth model from a DP perspective

- Consider a model in which the production function is subject to shocks to total factor productivity

$$y_t = a_t f(k_t)$$

- These productivity shocks are assumed to be governed by a Markov process, so that

$$a(\varsigma_t)$$

$$prob(\varsigma_{t+1} = \varsigma) = \gamma(\varsigma_t)$$

and additional restrictions are imposed so that productivity is stationary (a common initial assumption but one that seems counterfactual given Solow-style accounting).

Accumulation of capital

- Endogenous state evolution equation in the sense of our earlier presentation of DP
- Control is consumption

$$k_{t+1} = k_t + g(k_t, a_t, c_t)$$
$$= k_t + a_t f(k_t) - \delta k_t - c_t$$

Objective

- Discounted utility from consumption (with infinite horizon as special case)

$$General : E_0 \sum_{t=0}^{T} \beta^t u(k_t, x_t, c_t)$$

$$Specific : E_0 \sum_{t=0}^{T} \beta^t u(c_t)$$

Explicit horizon

- To allow consideration of iterative schemes in theory and computation
- To allow for finite horizon economies
- Requires some additional notation in terms of keeping track of time (lose time invariance because optimal behavior depends on “how long is left”)

2. Value Function Iteration

- We now describe an iterative method of determining value functions for a series of problems with increasing horizon
- This approach is a practical one introduced by Bellman in his original work on dynamic programming.

Review of Bellman's core ideas

- Focused on finding "policy function" and "value function" both of which depend on states (endogenous and exogenous states).de
- Subdivided complicated intertemporal problems into many "two period" problems, in which the trade-off is between the present "now" and "later".
- Working principal was to find the optimal control and state "now", taking as given that latter behavior would itself be optimal.

The Principle of Optimality

- “An optimal policy has the property that, whatever the state and optimal first decision may be, the remaining decisions constitute an optimal policy with respect to the state originating from the first decisions”—Bellman (1957, pg. 83)

Following the principle,

- The natural maximization problem is

$$\max_{c_t, k_{t+1}} \{u(c_t) + \beta E_t V(k_{t+1}, \varsigma_{t+1}, T-t)\}$$

$$s.t. \;\; k_{t+1} = k_t + g(k_t, a(\varsigma_t), c_t)$$

where the right hand side is the current momentary objective (u) plus the consequences (V) of for the discounted objective of behaving optimally in the future given that there are h=T-t periods to go (h is length of “horizon”).

Noting that time does not enter in an essential way other than through the horizon

- We can write this below as (with ' again meaning next period)

$$\max_{c,k'}\{u(c)) + \beta EV(k', \varsigma', h') \mid (k, \varsigma)\}$$

$$s.t. \;\; k' = k + g(k, a(\varsigma), c)$$

- So then the *Bellman equation* is written as (with the understanding that h=h'+1)

$$V(k, \varsigma, h) = \max_{c,k'}\{u(c) + \beta EV(k', \varsigma', h') \mid (k, \varsigma)\}$$

$$s.t. \;\; k' = k + g(k, a(\varsigma), c)$$

After the maximization

- We know the optimal policy and can calculate the associated value, so that there is now a Bellman equation of the form

$$V(k,\varsigma,h) = \{u(c(k,\varsigma,h))) + \beta E V(k + g(k,a(\varsigma),c(k,\varsigma,h)),\varsigma',h') \mid (k,\varsigma)\}$$

- Note that the policy function and the value function both depend on the states and the horizon.
- The Bellman equation provides a mapping from one value function (indexed by h'=h+1) to another (indexed by h)

3. An example

- Suppose that the utility function is u(c) = log(c), that the production function is Cobb-Douglas with capital parameter α and there is complete depreciation (δ=1). These assumptions allow us to find the policy and value functions as loglinear equations
- Then the accumulation equation is

$$k' = a(\varsigma)k^{\alpha} - c$$

One period problem

- Suppose that “there is no tomorrow” so that the value of future capital is zero and thus it is optimal to choose k’=0.
- Then, the optimal policy and associated value function are

$$c(k,\varsigma,1) = a(\varsigma)k^{\alpha}$$

$$v(k,\varsigma,1) = u(c) = \log(c)$$

$$= \log a(\varsigma) + \alpha \log k$$

Two period problem

- Bellman Equation

$$V(k,\varsigma,2) = \max_{c,k'}\{\log(c) + \\ + \beta E(\log(a') + \alpha \log(k')\}$$

$$s.t \qquad k' = ak^{\alpha} - c$$

- Optimal policy from Euler equation

$$0 = \frac{1}{c} - \beta \frac{\alpha}{ak^{\alpha} - c}$$

$$c(k,\varsigma,2) = \frac{1}{1+\beta\alpha} a(\varsigma) k^{\alpha} \qquad k'(k,\varsigma,2) = \frac{\beta\alpha}{1+\beta\alpha} a(\varsigma) k^{\alpha}$$

Under optimal policies

$$
\begin{aligned}
V(k,\varsigma,2) = \{&\log(\frac{1}{1+\beta\alpha}\, a(\varsigma)k^{\alpha}) + \beta E(\log(a(\varsigma\,')) \mid \varsigma \\
&+\beta\alpha \log(\frac{\beta\alpha}{1+\beta\alpha}\, a(\varsigma)k^{\alpha})\} \\
&= E(\log(a(\varsigma\,')) \mid \varsigma + \beta\alpha \log(a(\varsigma)) \\
&+(1+\beta\alpha)\alpha \log(k) \\
&+\log(\frac{1}{1+\beta\alpha}) + \beta\alpha \log(\frac{\beta\alpha}{1+\beta\alpha})
\end{aligned}
$$

A candidate solution

$$v(k,\varsigma,h) = \theta_h \log(k) + g_h(\varsigma) + \eta_h$$

$$v(k,\varsigma,h+1) = \max_{c,k'}\{\log(c) + \beta E[\theta_h \log(k') + g_h(\varsigma') + \eta_h] \mid (k,\varsigma)\}$$

$$s.t. \quad k' = a(\varsigma)k^{\alpha}$$

Finding the optimal policy and value function

$$EE: 0 = \frac{1}{c} - \frac{\beta\theta_h}{ak^\alpha - c}$$

$$\Rightarrow c(k,\varsigma,h+1) = \frac{1}{1+\beta\theta_h} a(\varsigma)k^\alpha$$

$$\Rightarrow v(k,\varsigma,h+1) = \{\log(\frac{1}{1+\beta\theta_h} a(\varsigma)k^\alpha)$$

$$+ \beta E[\theta_h \log(\frac{\beta\theta_h}{1+\beta\theta_h} a(\varsigma)k^\alpha) + g_h(\varsigma') + \eta_h]\}$$

$$= [\alpha(1+\beta\theta_h)\log(k)]$$

$$+ [(1+\beta\theta_h)\log(a(\varsigma) + \beta E g_h(\varsigma') \mid \varsigma$$

$$+ [\log(\frac{1}{1+\beta\theta_h}) + \beta\theta_h \log(\frac{\beta\theta_h}{1+\beta\theta_h}) + \beta\eta_h]$$

Comments

- The more importance (value) is attached to future capital, the smaller is consumption's share of output
- One value function which is loglinear in k' generates another that is loglinear in k.
- A recursion governs the capital coefficient, which can be used to show that it increases with the horizon

$$\theta_{h+1} = [\alpha(1+\beta\theta_h)]$$
$$= \alpha[1+(\alpha\beta)+...+(\alpha\beta)^h]$$

Comments

- Other more complicated recursions govern the other coefficients (these are harder to solve, since the difference equations are more complicated).
- Illustrates general idea of calculating a sequence of value functions and associated policy functions.

Infinite horizon problem

$$v(k,\varsigma) = \theta \log(k) + g(\varsigma) + \eta$$

$$v(k,\varsigma) = \max_{c,k'} \{\log(c) + \beta E[\theta \log(k') + g(\varsigma') + \eta] \,|\, (k,\varsigma)\}$$

$$s.t. \quad k' = a(\varsigma)k^{\alpha}$$

$$EE: 0 = \frac{1}{c} - \frac{\beta\theta}{a(\varsigma)k^{\alpha} - c} \Rightarrow c = \frac{1}{1+\beta\theta} a(\varsigma)k^{\alpha}$$

Value function implication

$$v(k,\varsigma) = \theta \log(k) + g(\varsigma) + \eta$$

$$\begin{aligned} BE: v(k,\varsigma) &= \alpha(1+\beta\theta)\log(k) \\ &\quad + (1+\beta\theta)\log(a(\varsigma)) + \beta E g(\varsigma') \mid \varsigma \\ &\quad + \log\left(\frac{1}{1+\beta\theta}\right) + \beta\theta \log\left(\frac{\beta\theta}{1+\beta\theta}\right) + \beta\eta \end{aligned}$$

$$\Rightarrow \theta = \alpha(1+\beta\theta) = \frac{\alpha}{1-\alpha\beta}$$

Comments

- Value function of infinite horizon is limit of finite horizon functions as h goes to infinity.
- We can see this for capital coefficients using standard formula for geometric sum (other coefficients are more complicated)
- Example suggests that one strategy for approximating the infinite horizon value function is to construct the sequence of finite horizon value functions (this would be very inefficient here, though, because it is so easy to compute the infinite horizon function)

Comments (contd)

- Such approximation strategies motivate the study of the properties of the Bellman equation as a mapping on a space of (value) functions.
- Economics of problem: distribution of uncertainty foes not affect saving (accumulation) in this setting for reasons related to Sandmo's discussion of offsetting substitution and income effects of rate of return uncertainty.

4. Discretization

- Let’s suppose that there are only a finite set of capital levels which can be chosen, rather that treating the level of capital as a continuous variable.
- Similarly, we assume that productivity can only take on a finite set of values
- Let’s suppose that consumption is continuous and, to simplify the discussion below, let’s think of substituting in the accumulation constraint, so that there is a reduced form objective (as in our discussion of the calculus of variations)

- Discrete capital and productivity “grids”

$$k : \kappa_1 < \kappa_2 < ....\kappa_m$$

$$a : \eta_1 < \eta_2 < ....\eta_n$$

- Reduced form objective

$$z(k, k', a) = u(af(k) + (1-\delta)k - k')$$

- Hence: we can form a (big) list of z numbers: utility for all different k,a,k’ triplets
- Similarly, the value function is just a list of numbers giving value for each discrete state.

DP iterations under certainty

- Start with given value function v(k)
- In each state k, find the k’ value that maximizes $z(k,k')+\beta Ev(k')$ and compute the associated value $v(k)=\max\{z(k,k')+\beta Ev(k')\}$
- Collecting up results across states produces a new v(k)
- Continue until change in v is small.

Example

- Look at iterative scheme in practice using simple MATLAB program
- Model parameters

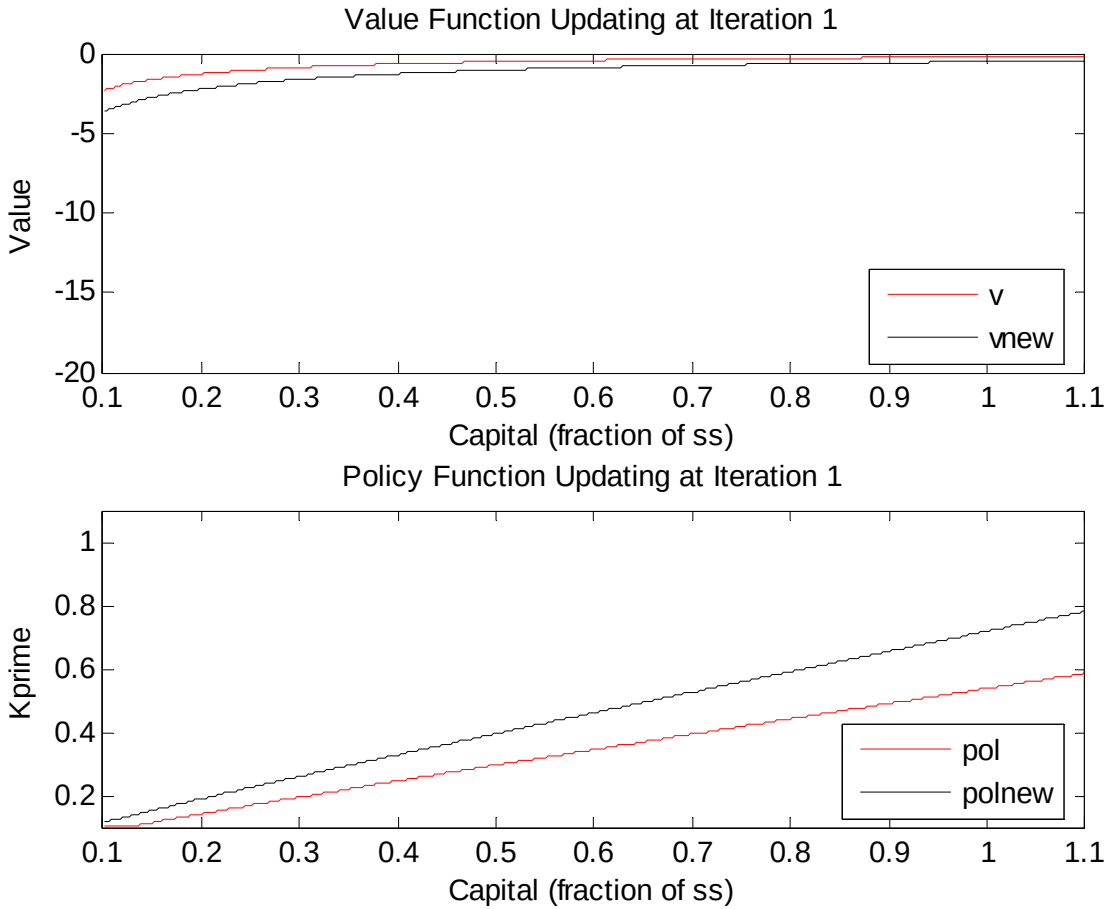
$$\beta = 1/(1+.06)$$

$$u(c) = \frac{1}{1-\sigma} c^{1-\sigma} \quad \text{with } \sigma = 3$$

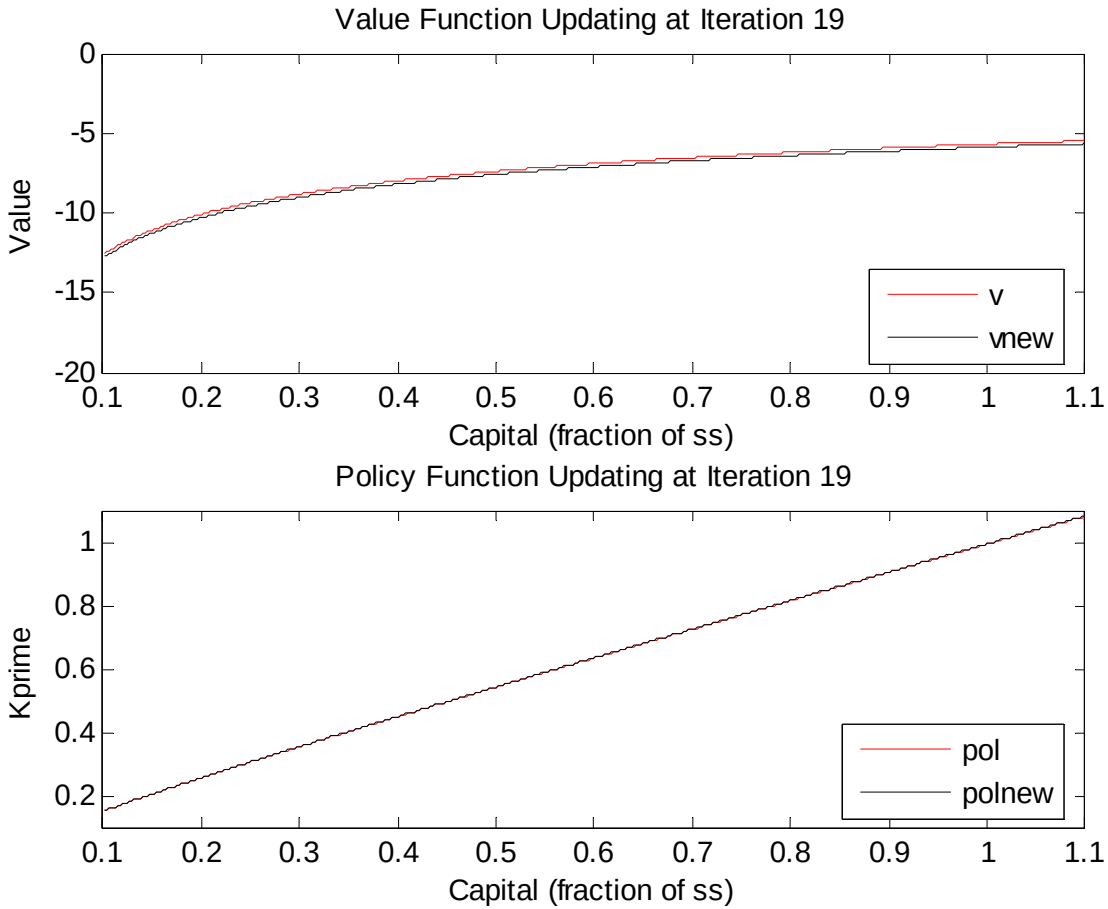
$$y = ak^{\alpha} \quad \text{with } a = 1 \text{ and } \alpha = 1/3$$

$$\delta = .1$$

Early iteration



Later iteration



Lessons: Specific and General

- Value function to converge more slowly than policy function: welfare is more sensitive to horizon than behavior.
  - Has motivated work on “policy iteration” where one starts with an initial policy (say, a linear approximation policy); evaluates the value function; finds a new policy that maximizes $z(k,k')+\beta Ev(k')$ and then iterates until the policy converged.
  - Has motivated development of “acceleration algorithms” Example: evaluate value under some feasible policy (say constant saving rate policy) and then use this as the starting point for value iteration
- Policy function looks pretty linear.
  - Has comforted some in use of linear approximation methods
  - Has motivated others to look for “peicewise linear” policy functions.

5. The Curse of dimensionality

- We have studied a model with one state which can taken on N different values.
- Our “brute force” optimization approach involves calculating z(k,k’) at each k for each possible k’ and then picking the largest one. Hence, we have to compute z(k,k’) a total of N*N times.
- If we have a “good” method of finding the optimal policy in at each point k, such as using properties of the objective, then the problem might only involve NM computations (with M<N being the smaller number of steps in the maximization step).
- Suppose that there were two states, each of which can take on N different values. We would have to find an optimal state evolution at each of N*N points and a blind search would require evaluating z at N*N points – all of the possible actions -- for each at each state. A more sophisticated search would require (N*N*M), as we would be doing M computations at each of N*N points in a state “grid”.

Curse

- Even if computation is maximially efficient (M=1, so that one does only one evaluation of Z at each point in a state grid), one can see that this state grid gets “large” very fast: if there are j states then there are N^j points in the grid: with 100 points for each state and 4 state variables, one is looking at a grid of 10^8 of points.
- Discretization methods are not feasible for complicated models with many states. Has motivated study of many other computational procedures for dynamic programming.

6. The importance of initial conditions

- We started the example dynamic programs with a future v=0 and with $v(k,a,0)=u(af(k)+(1-\delta)k)$.
- Theorems in the literature on dynamic programming guarantee the convergence of value function iteration – under specified conditions on u,f,a – from any initial condition. In this sense, the initial condition is irrelevant. Such theorems will be developed in Miao's "Economic Dynamics" course in the spring.
- However, the theorems indicate that for a large number of iterations, results from any two starting points are (i) arbitrarily close to each other and (ii) arbitrarily close to the solution for the infinite horizon case.
- In a practical sense, initial conditions can be critical. Imagine that, by accident or design, one started off with exactly the correct value function for the infinite horizon case, but you did not know the policy function. In this situation, there would only be one iteration, because the infinite horizon Bellman equation means that the "new" v will be equal to the "old" v.

7. Adding uncertainty

- Since we have discretized the state space, it is also natural to discretize the shock variable.
- The discretization is a Markov chain, with fixed probabilities of making transitions from one exogenous state to another.

Transition matrix for Markov chain (elements are prob(a'|a)=π_{ij})

	Later: High	Later: Medium	Later: Low
Now: High	.5	.4	.1
Now: Medium	.2	.6	.2
Now: Low	.1	.4	.5

DP iterations with uncertainty

- At each grid point (k= κ_l ,a= η_i), the Bellman equation is

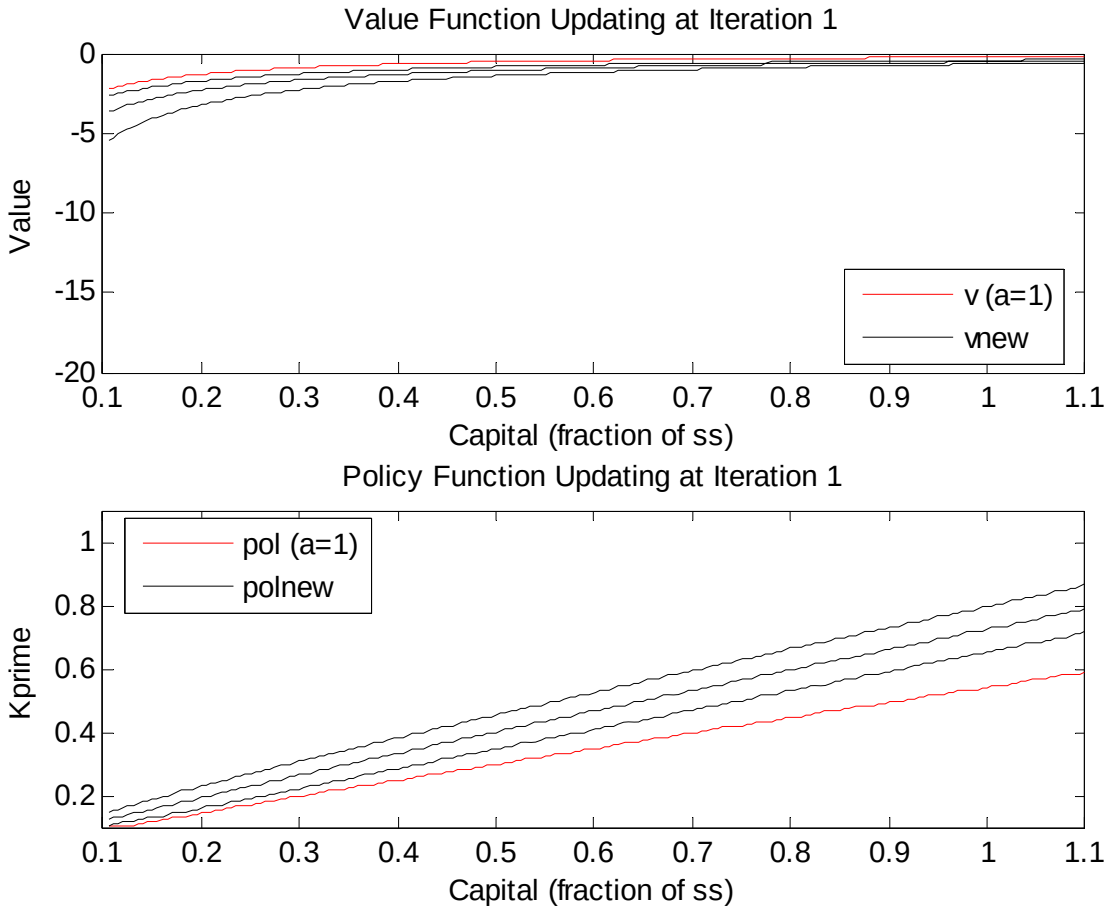
$$V(\kappa_l, \eta_i, h) = \max_{k'}\{z(\kappa_l, \eta_i, k') + \beta \sum_{j=1}^{J} \pi_{ij} EV(k', \eta_j')\}$$

- DP iterations with uncertainty require calculating the "expected value" as well as the prior computations discussed above.
- With a Markov chain, this is just adding up value functions after multiplying by future states.
- Grid for v now must depend on (k,a), which adds to curse of dimensionality.

Stochastic dynamic program

- Three value functions in graph: each of three values of “a”
- Three policy functions in graph: each of three values of “a”.
- No longer see “improvements” for all three: just the middle “a” function to avoid graphical clutter (it is the dashed line).

Early iteration



Later iteration