Economics 817 **Winter 2004**

Advanced Macroeconomic Theory II **Hiroyuki Kasahara**

Homework 2

(Due: January 29 by 5 pm)

Note: Study groups *discussing* the problems are strongly encouraged. But write your own answers.

1 Dynamic Programming under Certainty

Consider the one-sector optimal growth problem:

$$\max_{\{c_t,k_{t+1}\}_{t=0}^{\infty}} \sum_{t=0}^{\infty} \beta^t U(c_t)$$
$$s.t. \quad c_t + k_{t+1} \le f(k_t), \quad t = 0, 1, ...$$
$$k_0 \quad \text{given.}$$

where $\beta \in (0,1)$; $U : \mathbf{R}_+ \to \mathbf{R}$ is a utility function that is continuous, strictly increasing, strictly concave, and twice differentiable; $f : \mathbf{R}_+ \to \mathbf{R}_+$ is a production function that is continuous, strictly increasing, strictly concave, and twice differentiable with $f(0) = 0$, $\lim_{k\to 0} f'(k) > 1$, and $\lim_{k\to\infty} f'(k) < 1$.

The corresponding Bellman equation is:

$$V(k) = \max_{k' \le f(k)} U[f(k) - k'] + \beta V(k'). \tag{1}$$

Define an operator by the right hand side of the Bellman equation (1):

$$(TV)(k) = \max_{k' \le f(k)} U[f(k) - k'] + \beta V(k'). \tag{2}$$

Let V^* be the fixed point of (2): $TV^* = V^*$.

1. Argue that the above assumptions on U and f imply that we can only consider the set of maintainable capital stocks $[0, \bar{k}]$, where $\bar{k}$ satisfies $f(\bar{k}) = \bar{k}$. [Note that $f(k) < k$ for all $k > \bar{k}$ by $f'(k) < 1$ for $k > \bar{k}$.]

2. Prove (i) there exists a unique fixed point of the operator (2) and (ii) the unique fixed point V^* is a *bounded and continuous* function. [Replicate the proof of Theorem 4.6 in Stokey and Lucas in your

own words. For (i), show that the Blackwell's sufficient condition is satisfied. For (ii), show that if V_n is bounded and continuous, then $V_{n+1} = TV_n$ is also bounded and continuous (i.e., the operator (2) maps the space of bounded continuous functions—denoted by $C(X)$—into itself). For the preservation of continuity, use the Theorem of the Maximum.]

3. Prove that V^* is (i) non-decreasing and (ii) strictly increasing. [Replicate the proof of Theorem 4.7 in Stokey and Lucas in your own words.]

4. Prove that V^* is (i) weakly concave and (ii) strictly concave.[Replicate the proof of Theorem 4.8 in Stokey and Lucas in your own words.]

5. Prove that there exists an unique solution to $\max_{k' \leq f(k)} U[f(k) - k'] + \beta V^*(k')$ for each k. [Hint: If you add a concave function to a strictly concave function, then you will get a strictly concave function.]

6. Prove that V^* is differentiable on $(0, \bar{x})$. [Replicate the proof of Theorem 4.11 in Stokey and Lucas in your own words.]

7. (Monotone Optimal Policy) Let $g(k) \in argmax_{x \leq f(k)} U[f(k) - x] + \beta V^*(x)$. Prove that $g(k)$ is strictly increasing in k.

2 Numerical Dynamic Programming under Certainty

This exercise is based on Ch.12.5 of Judd (1998). Please refer to pp.424-428 of Judd (1998). It is very important to add your comments generously so that Jeremy and I can understand your code. Unless it's really obvious, add comments on every smaller segments of code, or possibly every line, in this homework.

Consider the Bellman equation corresponding to the one-sector optimal growth problem:

$$V(k) = \max_{k' \leq f(k)} U[f(k) - k'] + \beta V(k'). \tag{3}$$

We choose:

$$U(c) = \ln c \tag{4}$$

$$f(k) = Ak^{\alpha} \tag{5}$$

so that we have a closed form solution to the policy function: $k_{t+1} = \alpha\beta A k_t^{\alpha}$.

By approximating the continuous state space of k by a finite set $K = \{k_1, ..., k_n\}$, we approximate the Bellman equation by

$$\tilde{V}(i) = \max_{j=1,2,\ldots,n} U[f(k_i) - k_j] + \beta \tilde{V}(j), \tag{6}$$

where $\tilde{V}(i)$ represents the ith element of a "n by 1" vector. Namely, the continuous function $V(k)$ that maps $[0, \bar{k}]$ into $\mathbf{R}$ is approximated by a function which maps $\{1, 2, ..., n\}$ into $\mathbf{R}$. The space of functions that maps $\{1, 2, ..., n\}$ into $\mathbf{R}$ together with a metric $d_\infty(\tilde{V}, \tilde{W}) \equiv \max_{i=1,2,...,n} |\tilde{V}(i) - \tilde{W}(i)|$ is a complete metric space since it's just n-dimentional Euclidean space. The right hand side of (6) defines an operator that maps n-dimentional Euclidean space into itself.[1]

We are going to solve this discretized Bellman equation using the "value iteration" method using computers.

Set the parameter values to: $\beta = 0.96$, $\alpha = 0.25$. Also, $A = (\alpha\beta)^{-1}$, implying that the steady state capital level is 1.

Our discretization will be $k_i = 0.8 + (i-1)\kappa$. We choose $\kappa = 0.01$ with $n = 41$ so that $k_1 = 0.8$ and $k_n = 1.2$.

1. Write a computer program that numerically solve the fixed point of the discretized Bellman equation (6) using the "Value Function Iteration Algorithm":

   (a) Set the initial guess $V^0 = \mathbf{0}_n$, where $\mathbf{0}_n$ is an "n by 1" vector where each element is a scalar 0.

   (b) After l value function iterations, we have a value function, $\tilde{V}^l$. Given $\tilde{V}^l$, compute

   $$\tilde{V}^{l+1}(i) = \max_{j=1,2,...,n} U[f(k_i) - k_j] + \beta \tilde{V}^l(j),$$

   for $i = 1, 2, ..., n$.

   One way to do this is to use a "do-loop" twice. For $i = 1$, compute $U[f(k_1) - k_j] + \beta\tilde{V}^l(j)$ for $j = 1, 2, ..., n$ and store their values in an "n by 1" vector. Then, find the max among those values. If the mth element is the max, then $\tilde{V}^{l+1}(1) = U[f(k_1) - k_m] + \beta\tilde{V}^l(m)$. This is an "inner" loop. Repeat this for $i = 2, 3, ..., n$ ("outer loop') to get $\{\tilde{V}^{l+1}(2), \tilde{V}^{l+1}(3), ..., \tilde{V}^{l+1}(n)\}$ and store them in an "n by 1" vector; this is a function $\tilde{V}^{l+1} = T(\tilde{V}^l)$, where T is an operator defined by the right hand side of (6).

   In practice, avoid loops as much as possible in Matlab, Gauss, and Ox since they slow your program; instead, use vector or matrix operations. In this context, first create an "n by n" utility matrix, say $\mathbf{U}$, with the (i, j)th element given by $\mathbf{U}(i, j) \equiv U[f(k_i) - k_j]$. Second, compute an "n by n" matrix $\tilde{W}^{l+1} = \mathbf{U} + \beta\mathbf{1}_n \bigotimes \tilde{V}^{l'}$, where $\mathbf{1}_n$ is an "n by 1" vector of ones, $'$ is a transpose,

[1] Can you prove that the fixed point of this operator satisfies (i) strictly increasing and (ii) strictly concave? Can you prove the monotonicity of the policy function?

and $\bigotimes$ is the Kronecker product.[2] Finally, for each *row*, take a max across columns to compute an "n by 1" vector $\tilde{V}^{l+1}$. You can do this using only one command.[3]

(c) Repeat the recursive computation of Step (b) until $\max_{i=1,2,...,n} |V^l(i) - V^{l+1}(i)| < \epsilon/(1-\beta)$, where we set $\epsilon = 0.001$.[45] Let $\tilde{V}^*$ be the computed fixed point.

(d) Compute the policy function, P, which is an "n by n" matrix of which the (i,j)th element is 1 if $j = j^*(i) \equiv argmax_{j' \in \{1,2,...,n\}} U[f(k_i) - k'_j] + \beta \tilde{V}^*(j')$, 0 otherwise.[6]

2. We can think that the value function is the total sum of discounted value of utility under the optimal policy function. To see this, first, compute that the utility function under the policy function P, denoted by $\mathbf{U}^P$, which is an "n by 1" vector with the ith element equal to $U[f(k_i) - k_{j^*(i)}]$ where $k_{j^*(i)}$ is the optimal choice for the next capital stock given the current state is k_i.[7] Next, compute $\mathbf{U}^P + \beta P \tilde{V}^*$ and check whether it is equal to $\tilde{V}^*$ or not. Explain briefly why $\tilde{V}^* = \mathbf{U}^P + \beta P \tilde{V}^*$ must hold.

3. Compute $[I - \beta P]^{-1} \mathbf{U}^P$ and check if it is equal to $\tilde{V}^*$ or not. Explain briefly why $\tilde{V}^* = [I - \beta P]^{-1} \mathbf{U}^P$ must hold. [Hint. From the previous question, we have an expression $\tilde{V}^* = \mathbf{U}^P + \beta P \mathbf{U}^P + \beta^2 P^2 \mathbf{U}^P + \beta^3 P^3 \mathbf{U}^P + ...$]

4. Plot the graph of the computed policy function with k_i on the x-axis and $k_{j^*(i)}$ on the y-axis.[8] Also plot k_i against $\alpha\beta A k_i^\alpha$ on the same graph to see whether the discretized Bellman equation (6) approximates the Bellman equation (3) well.

---

[2] In Matlab, you can use the command $W = U + betakron(ones(n,1), V_0')$.

[3] In Matlab, you can use the command $V_1 = \max(W, 2)$.

[4] In Matlab, write something like:

$beta = 0.96$; $epsilon = 0.001$;

$tol = epsilon/(1 - beta)$;

$V_0 = zeros(n, 1)$;

$metric = 1$;

while $metric < tol$

....

$metric = max(V_0 - V_1)$;

$V_0 = V_1$;

end

where V_0 represents the previous value function and V_1 is the current value function.

[5] You can actually set $\epsilon = 0$ in this context since the state space is finite.

[6] Hint: in Matlab, use "max" command. Type "help max."

[7] Matlab: $sum(P.*U, 2)$ will return the "n by 1" vector, $\mathbf{U}^P$.

[8] Type "help plot" in Matlab.

5. Try the same program with (i) $\kappa = 0.001$ with $n = 401$ and (ii) $\kappa = 0.1$ with $n = 5$ and plot, all of them together including the case of $\kappa = 0.01$ with $n = 41$, the graph of the computed policy function with k_i on the x-axis and $k_{j^*(i)}$ on the y-axis.