5

Dynamic Programming: A Less Technical Approach

In mathematics you don't understand things. You just get used to them.

—Johann von Neumann

In precisely built mathematical structures, mathematicians find the same sort of beauty others find in enchanting pieces of music, or in magnificent architecture. There is, however, one great difference between the beauty of mathematical structures and that of great art. Music by Mozart, for instance, impresses greatly even those who do not know musical theory; the cathedral in Cologne overwhelms spectators even if they know nothing about Christianity. The beauty in mathematical structures, however, cannot be appreciated without understanding of a group of numerical formulae that express laws of logic. ... Accordingly, I once believed that without numerical formulae, I could never communicate the sweet melody played in my heart.

—Kiyosi Itô, *Kyoto Lecture, 1998*

1 Introduction

Dynamic programming is another method for solving dynamic optimization problems. It becomes an important tool in macroeconomic literature, and has some appealing features in its solution strategies. It is especially suitable for solving problems under uncertainty, and because of its recursive nature computer simulation is easily done when open form solution is hardly to be obtained.

The basic idea of dynamic programming is briefly introduced in Section 2. Section 3 shows three methods of how to derive first order conditions. An example gives more details in Section 3.4. Then Section 4 discusses the computational methods for solutions under various occasions. Section 5.1 and 5.2 extend the methods for problems under uncertainty and continous time, hoping to unveil the tip of iceberg, i.e. the rich applications of dynamic programming, to readers.

This chapter is written in a non-technical way in the sense that it only tells readers how to apply dynamic programming to solve economic problems, without involving the concrete mathematics. However, a brief introduction of functional analysis in Appendix A.2 shows that what we have done throughout this chapter is indeed theoretically sound.

2 Basic Idea

Consider a general discrete-time optimization problem

$$\begin{aligned} &\max_{\{c_t,k_{t+1}\}_{t=0}^{+\infty}} \sum_{t=0}^{+\infty} \beta^t u(c_t) \\ &\quad s.t.\ k_{t+1} = f(c_t, k_t). \end{aligned}$$

You may interpret this problem in an economic context. Given any capital stock level k_t in period t, a representative agent maximizes her life-long utility by choosing her period t consumption level c_t (such variables whose value is directly chosen by individuals are called *control variables*; in contrast those not directly chosen by individuals are called *state variables* such as k_t). So essentially the optimization problem is to seek a *policy function* $c_t = h(k_t)$ which maps the state k_t into the control c_t. As soon as c_t is chosen, the *transition function* $k_{t+1} = f(c_t, k_t)$ determines next period state k_{t+1} and the same procedure repeats. Such procedure is *recursive*.

The basic idea of dynamic programming is to collapse a multi-periods problem into a sequence of two-periods problem at any t using the recursive nature of the problem

$$
\begin{aligned}
V(k_t) &= \max_{c_t, k_{t+1}} \sum_{i=0}^{+\infty} \beta^i u(k_{t+i}) \\
&= \max_{c_t, k_{t+1}} \left\{ u(c_t) + \beta \sum_{i=0}^{+\infty} \beta^i u(c_{t+i+1}) \right\} \\
&= \max_{c_t, k_{t+1}} \{ u(c_t) + \beta V(k_{t+1}) \} \\
s.t. \quad & k_{t+1} = f(c_t, k_t). \qquad (1)
\end{aligned}
$$

Equation $V(k_t) = \max_{c_t, k_{t+1}} \{u(c_t) + \beta V(k_{t+1})\}$ is known as *Bellman equation*. The *value function* $V(\cdot)$ is only a function of state variable k_t because the optimal value of c_t is just a function of k_t. Then the original problem can be solved by the methods we learned for two-periods problems plus some tricks.

3 Getting the Euler Equation

The key step now is to find the proper first order conditions. There are several possible approaches, and readers may pick up one of them with which he or she feels comfortable.

3.1 Using Lagrangian

Since the problem looks pretty similar to a maximization problem with equality constraint, one may suggest Lagrangian — Let's try.

Rewrite $V(k_t)$ as

$$
V(k_t) = \max_{c_t, k_{t+1}} \underbrace{\{u(c_t) + \beta V(k_{t+1}) + \lambda_t [f(c_t, k_t) - k_{t+1}]\}}_{\mathscr{L}_t}.
$$

Step 1 Since $V(k_t)$ is maximized value for Lagrangian, the first order conditions with respect to c_t and k_{t+1} must hold,

$$
u'(c_t) + \lambda_t \frac{\partial f(c_t, k_t)}{\partial c_t} = 0, \qquad (2)
$$

$$
\beta V'(k_{t+1}) - \lambda_t = 0. \qquad (3)
$$

Step 2 Since $V(k_t)$ is optimized at k_t, then

$$
\begin{aligned}
V'(k_t) = u'(c_t)\frac{dc_t}{dk_t} + \beta V'(k_{t+1})\frac{dk_{t+1}}{dk_t} + \frac{d\lambda_t}{dk_t}\left[f(c_t,k_t) - k_{t+1}\right] \\
+\lambda_t\left[\frac{\partial f(c_t,k_t)}{\partial k_t} + \frac{\partial f(c_t,k_t)}{\partial c_t}\frac{dc_t}{dk_t} - \frac{dk_{t+1}}{dk_t}\right] \\
= \underbrace{\left[u'(c_t) + \lambda_t\frac{\partial f(c_t,k_t)}{\partial c_t}\right]}_{(A)}\frac{dc_t}{dk_t} + \underbrace{\left[\beta V'(k_{t+1}) - \lambda_t\right]}_{(B)}\frac{dk_{t+1}}{dk_t} \\
+\frac{d\lambda_t}{dk_t}\underbrace{\left[f(c_t,k_t) - k_{t+1}\right]}_{(C)} + \lambda_t\frac{\partial f(c_t,k_t)}{\partial k_t}.
\end{aligned}
$$

$(A) = 0$ by (2), $(B) = 0$ by (3), and $(C) = 0$ by first order condition of Lagrangian. Therefore

$$
V'(k_t) = \lambda_t\frac{\partial f(c_t,k_t)}{\partial k_t}. \tag{4}
$$

Step 3 By (2) and (3) eliminate λ_t

$$
u'(c_t) + \beta V'(k_{t+1})\frac{\partial f(c_t,k_t)}{\partial c_t} = 0.
$$

And since t is arbitrarily taken, this equation must hold if we take one period backward

$$
u'(c_{t-1}) + \beta V'(k_t)\frac{\partial f(c_{t-1},k_{t-1})}{\partial c_{t-1}} = 0. \tag{5}
$$

Next insert (2) into (4) to elimate λ and (5) into (4) to elimate $V'(k_t)$, then Euler equation is obtained.

3.2 Tracing Dynamics of Costate Variable

The other way of thinking is to trace the dynamics of costate variable $V(k_t)$.

Step 1 Since $V(k_t)$ is maximized value of $u(c_t) + \beta V(k_{t+1})$, then the first order condition with respect to c_t gives

$$
u'(c_t) + \beta V'(k_{t+1})\frac{\partial k_{t+1}}{\partial c_t} = u'(c_t) + \beta V'(k_{t+1})\frac{\partial f(c_t,k_t)}{\partial c_t} = 0. \tag{6}
$$

Step 2 Since $V(k_t)$ is optimized at k_t, then

$$V'(k_t) = u'(c_t)\frac{dc_t}{dk_t} + \beta V'(k_{t+1})\left[\frac{\partial f(c_t,k_t)}{\partial k_t} + \frac{\partial f(c_t,k_t)}{\partial c_t}\frac{dc_t}{dk_t}\right]$$
$$= \left[u'(c_t) + \beta V'(k_{t+1})\frac{\partial f(c_t,k_t)}{\partial c_t}\right]\frac{dc_t}{dk_t} + \beta V'(k_{t+1})\frac{\partial f(c_t,k_t)}{\partial k_t}.$$

Apply (6) and get

$$V'(k_t) = \beta V'(k_{t+1})\frac{\partial f(c_t,k_t)}{\partial k_t}. \tag{7}$$

Since t is arbitrarily taken, (6) also holds for one period backward, i.e.

$$V'(k_t) = -\frac{u'(c_{t-1})}{\beta\frac{\partial f(c_{t-1},k_{t-1})}{\partial c_{t-1}}}. \tag{8}$$

Step 3 Apply (6) & (8) into (7) and obtain Euler equation.

3.3 Using Envelope Theorem

Solve the budget constraint for c_t and get $c_t = g(k_t, k_{t+1})$. Apply it to $V(k_t)$ and get a univariate optimization problem

$$V(k_t) = \max_{k_{t+1}} \{u(g(k_t,k_{t+1})) + \beta V(k_{t+1})\}.$$

Step 1 Similar as before, since $V(k_t)$ is the maximized value of $u(g(k_t,k_{t+1})) + \beta V(k_{t+1})$, then the first order condition with respect to k_{t+1} gives

$$u'(g(k_t,k_{t+1}))\frac{\partial g(k_t,k_{t+1})}{\partial k_{t+1}} + \beta V'(k_{t+1}) = 0. \tag{9}$$

Step 2 Since $V(k_t)$ is already optimized at k_t, differentiating $V(k_t)$ with respect to k_t gives

$$\frac{dV(k_t)}{dk_t} = \underbrace{\frac{\partial V(k_t)}{\partial k_t}}_{(A)} + \underbrace{\frac{\partial V(k_t)}{\partial k_{t+1}}\frac{\partial k_{t+1}}{\partial k_t}}_{(B)}. \tag{10}$$

This is pretty intuitive: k_t may generate a direct effect on $V(k_t)$ as part (A) shows; however, k_t may also generate an indirect effect on $V(k_t)$ through k_{t+1} (remember the dynamic budget

constraint). And since $V(k_t)$ is optimized by k_{t+1}, the first order condition implies that $\frac{\partial V(k_t)}{\partial k_{t+1}} = 0$. Therefore equation (10) becomes

$$V'(k_t) = \frac{\partial V(k_t)}{\partial k_t} = u'(g(k_t, k_{t+1}))\frac{dg(k_t, k_{t+1})}{dk_t}. \tag{11}$$

The mathematics behind it is the famous Envelope Theorem. You may find people call it *Benveniste-Scheinkman condition* in some cases.

Step 3 Similar as before, take one period forward for (11) and apply it into (9) then obtain Euler equation.

3.4 Example

Consider a discrete time Ramsey problem for a decentralized economy

$$\begin{aligned}
&\max_{\{c_t, b_t\}_{t=0}^{+\infty}} \sum_{t=0}^{+\infty} \beta^t u(c_t) \\
&\quad s.t.\ b_{t+1} - b_t = w_t + r_t b_t - c_t - n b_t.
\end{aligned}$$

Collapse the infinite horizon problem into a sequence of two-periods problem

$$\begin{aligned}
V(b_t) &= \max_{c_t, b_{t+1}} \sum_{i=0}^{+\infty} \beta^i u(c_{t+i}) \\
&= \max_{c_t, b_{t+1}} \left\{ u(c_t) + \beta \sum_{i=0}^{+\infty} \beta^i u(c_{t+i+1}) \right\} \\
&= \max_{c_t, b_{t+1}} \{u(c_t) + \beta V(b_{t+1})\} \\
s.t. \quad & b_{t+1} = w_t + (1 + r_t) b_t - c_t - n b_t.
\end{aligned}$$

Now we solve the problem with all three approaches.

3.4.1 Using Lagrangian

Rewrite Bellman equation in Lagrangian form

$$V(b_t) = \max_{c_t, b_{t+1}} \{u(c_t) + \beta V(b_{t+1}) + \lambda_t [w_t + (1 + r_t) b_t - c_t - n b_t - b_{t+1}]\}.$$

Step 1 The first order conditions of Lagrangian are

$$u'(c_t) - \lambda_t = 0, \tag{12}$$
$$\beta V'(b_{t+1}) - \lambda_t = 0, \tag{13}$$

and eliminate λ_t to get

$$u'(c_t) = \beta V'(b_{t+1}). \tag{14}$$

Step 2 Now differentiate $V(b_t)$ at b_t

$$\begin{aligned} V'(b_t) = {} & u'(c_t)\frac{dc_t}{db_t} + \beta V'(b_{t+1})(1 + r_t - n) \\ & + \frac{d\lambda_t}{dk_t}\underbrace{[w_t + (1 + r_t)b_t - c_t - nb_t - b_{t+1}]}_{=0} \\ & + \lambda_t\left[(1 + r_t - n) - \frac{dc_t}{db_t} - (1 + r_t - n)\right] \\ = {} & \underbrace{[u'(c_t) - \lambda_t]}_{=0}\frac{dc_t}{db_t} + \underbrace{[\beta V'(b_{t+1}) - \lambda_t]}_{=0}(1 + r_t - n) + \lambda_t(1 + r_t - n). \end{aligned}$$

That is,

$$V'(b_t) = \lambda_t(1 + r_t - n). \tag{15}$$

Step 3 Insert (12) and (14) into (15) and get the desired result.

3.4.2 Tracing Dynamics of Costate Variable

Now solve the same problem by tracing the dynamics of the costate variable.

Step 1 Since $V(b_t)$ is the maximized value of $u(c_t) + \beta V(b_{t+1})$, then the first order condition with respect to b_{t+1} gives

$$-u'(c_t) + \beta V'(b_{t+1}) = 0. \tag{16}$$

Step 2 Now differentiate $V(b_t)$ at b_t

$$V'(b_t) = u'(c_t)\frac{\partial c_t}{\partial k_t} + \beta V'(b_{t+1})\left[(1 + r_t - n) - \frac{\partial c_t}{\partial k_t}\right].$$

That is just

$$V'(b_t) = \beta(1 + r_t - n)V'(b_{t+1}). \tag{17}$$

Step 3 Insert (16) twice into (17) and get the desired result.

3.4.3 Using Envelope Theorem

Now solve the same problem with Envelope Theorem.

Step 1 Since $V(b_t)$ is maximized value of $u(c_t) + \beta V(b_{t+1})$, then the first order condition with respect to b_{t+1} gives

$$-u'(c_t) + \beta V'(b_{t+1}) = 0. \tag{18}$$

Step 2 Now the problem is to find $V'(b_{t+1})$. Differentiate $V(b_t)$ at b_t

$$V'(b_t) = \frac{\partial V(b_t)}{\partial b_t} = (1 + r_t - n)u'(c_t). \tag{19}$$

Step 3 Take one period backward for (18) and insert into (19) to obtain the Euler equation.

4 ⋆ Solving for the Policy Function

As seen in previous sections policy function $c_t = h(k_t)$ captures the optimal solution for each period given the corresponding state variable, therefore one may desire to get the solution of the policy function. Dynamic programming method has a special advantage for this purpose, and we will see several approaches in the following.

Now consider the problem of Brock & Mirman (1972). Suppose utility function takes the form $u_t = \ln c_t$ and the production function follows Cobb-Douglas technolody. No depreciation and population growth.

$$\max_{\{c_t, k_t\}_{t=0}^{+\infty}} \sum_{t=0}^{*\infty} \beta^t \ln c_t$$

$$s.t.\ k_{t+1} = k_t^\alpha - c_t.$$

4.1 Solution by Iterative Substitution the Euler Equation

Recall that the recursive structure of dynamic programming method implies that the problem that the optimizer faces in each period is the same as that she faces last period or next period, so the solution to such a problem should be time-invariant. Thus one can start from deriving the solution under some circumstances and iterate it on an infinite time horizon until it is time invariant. However this approach only works when the problem is simple.

4.1.1 Forward Induction

Set up the Bellman equation and solve for Euler equation. This gives

$$\begin{aligned}
\frac{1}{c_t} &= \alpha\beta\frac{k_{t+1}^{\alpha-1}}{c_{t+1}} \\
\frac{k_{t+1}}{c_t} &= \alpha\beta\frac{k_{t+1}^{\alpha}}{c_{t+1}} \\
\frac{k_t^\alpha - c_t}{c_t} &= \alpha\beta\frac{k_{t+1}^{\alpha}}{c_{t+1}} \\
\frac{k_t^\alpha}{c_t} &= \alpha\beta\frac{k_{t+1}^{\alpha}}{c_{t+1}} + 1.
\end{aligned}$$

Apply this condition to itself and get a geometric serial

$$\begin{aligned}
\frac{k_t^\alpha}{c_t} &= 1 + \alpha\beta\left(1 + \alpha\beta\frac{k_{t+2}^{\alpha}}{c_{t+2}}\right) \\
&= 1 + \alpha\beta + \alpha^2\beta^2 + \alpha^3\beta^3 + \dots \\
&= \frac{1}{1-\alpha\beta},
\end{aligned}$$

(why?) and this gives

$$c_t = (1-\alpha\beta)k_t^\alpha.$$

Another way to see this is exploring saving rate dynamics. Express c_t by k_t and k_{t+1}

$$\frac{1}{k_t^\alpha - k_{t+1}} = \beta\frac{1}{k_{t+1}^\alpha - k_{t+2}}\alpha k_{t+1}^{\alpha-1}. \tag{20}$$

Define saving rate at time t as

$$s_t = \frac{k_{t+1}}{k_t^\alpha},$$

Rearranging (20) gives

$$\begin{aligned}
\frac{1}{k_t^\alpha}\frac{1}{1-s_t} &= \beta\frac{1}{k_{t+1}^\alpha}\frac{1}{1-s_{t+1}}\alpha k_{t+1}^{\alpha-1} \\
\frac{k_{t+1}}{k_t^\alpha}\frac{1}{1-s_t} &= \frac{\alpha\beta}{1-s_{t+1}} \\
\frac{s_t}{1-s_t} &= \frac{\alpha\beta}{1-s_{t+1}},
\end{aligned}$$

and this is

$$s_{t+1} = 1 + \alpha\beta - \frac{\alpha\beta}{s_t}. \tag{21}$$

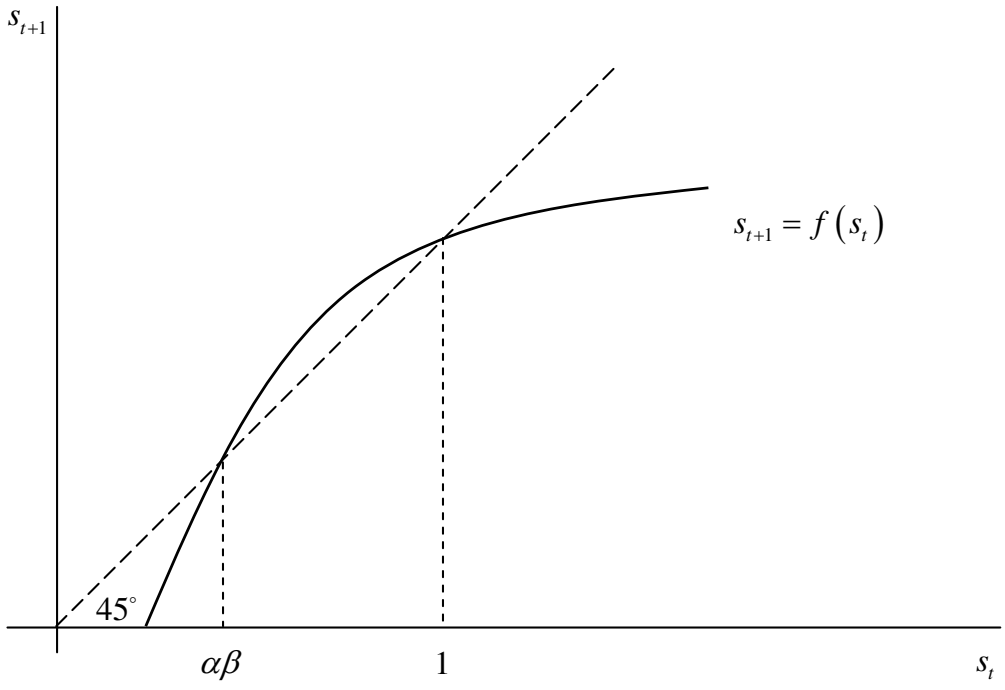

Fig. 1. Solution for s_t

Plot s_{t+1} as a function of s_t as Figure 1, and this gives two solutions, $\alpha\beta < 1$ and 1 respectly. Only the former is plausible. Then

$$c_t = (1 - s_t)k_t^\alpha = (1 - \alpha\beta)k_t^\alpha.$$

4.1.2 Backward Induction

Suppose that the world ends after some finite peroid T. Then surely for the last period

$$s_T = 0.$$

Apply this to (21)

$$s_T = 0 = 1 + \alpha\beta - \frac{\alpha\beta}{s_{T-1}},$$

solve to get

$$s_{T-1} = \frac{\alpha\beta}{1 + \alpha\beta}.$$

Continue this process,

$$s_{T-1} = \frac{\alpha\beta}{1 + \alpha\beta} = 1 + \alpha\beta - \frac{\alpha\beta}{s_{T-2}},$$

and this yields

$$s_{T-2} = \frac{\alpha\beta + \alpha^2\beta^2}{1 + \alpha\beta + \alpha^2\beta^2}.$$

We find that for any t between 0 and T

$$s_t = \frac{\sum_{i=1}^{T-t} \alpha^i\beta^i}{1 + \sum_{i=1}^{T-t} \alpha^i\beta^i} = \frac{\frac{\alpha\beta(1-\alpha^{T-t}\beta^{T-t})}{1-\alpha\beta}}{1 + \frac{\alpha\beta(1-\alpha^{T-t}\beta^{T-t})}{1-\alpha\beta}} = \frac{\alpha\beta(1 - \alpha^{T-t}\beta^{T-t})}{1 - \alpha\beta + \alpha\beta(1 - \alpha^{T-t}\beta^{T-t})}.$$

And in the limit

$$\lim_{T-t\to+\infty} s_t = \alpha\beta$$

implying that

$$c_t = (1 - \alpha\beta)k_t^\alpha.$$

4.2 Solution by Value-Function Iteration

Another solution method is based on iteration of the value function. The value function actually will be different in each period, just as we earlier found the function $g(k_t)$ was different depending on how close we were to the terminal period. But it can be shown (but we do not show this here) that as we iterate through time, the value function converges, just as $g(k_t)$ converged in our earlier example as we iterated back further away from the terminal period. This suggests that if we iterate on an initial guess for the value function, even a guess we know is incorrect, the iterations eventually will converge to the true function.

4.2.1 Guess and Verify

One may guess the form of solution and try to verify whether it's true. We guess that

$$V(k_t) = A + B \ln k_t.$$

Then the problem becomes

$$\begin{aligned} V(k_t) &= \max_{c_t, k_{t+1}} \{\ln c_t + \beta V(k_{t+1})\} \\ &= \max_{c_t, k_{t+1}} \{\ln c_t + \beta(A + B \ln k_{t+1})\} \\ s.t. \quad & k_{t+1} = k_t^\alpha - c_t. \end{aligned}$$

The first order condition with respect to k_{t+1} yields

$$\begin{aligned} -\frac{1}{c_t} + \frac{\beta B}{k_{t+1}} &= 0, \\ k_{t+1} &= \beta B(k_t^\alpha - k_{t+1}), \\ k_{t+1} &= \frac{\beta B}{1+\beta B} k_t^\alpha, \\ c_t &= \frac{1}{1+\beta B} k_t^\alpha. \end{aligned}$$

Then apply the results to the Bellman equation, and the following must hold if our conjecture is right

$$\begin{aligned} V(k_t) &= \ln\left(\frac{\beta B}{1+\beta B} k_t^\alpha\right) + \beta\left[A + B \ln\left(\frac{1}{1+\beta B} k_t^\alpha\right)\right] \\ &= \underbrace{\ln \beta B + \beta A - (1+\beta B)\ln(1+\beta B)}_{A} + \underbrace{\alpha(1+\beta B)}_{B} \ln k_t \\ &= A + B \ln k_t. \end{aligned}$$

Solve to get

$$
\begin{aligned}
B &= \frac{\alpha}{1-\alpha\beta}, \\
A &= \frac{1}{1-\beta}\left[\ln(1-\alpha\beta) + \frac{\alpha\beta}{1-\alpha\beta}\ln\alpha\beta\right], \\
c_t &= \frac{1}{1+\beta B}k_t^\alpha = (1-\alpha\beta)k_t^\alpha.
\end{aligned}
$$

and therefore

$$
\begin{aligned}
k_{t+1} &= \frac{\beta B}{1+\beta B}k_t^\alpha = \alpha\beta k_t^\alpha, \\
c_t &= \frac{1}{1+\beta B}k_t^\alpha = (1-\alpha\beta)k_t^\alpha.
\end{aligned}
$$

4.2.2 Value-Function Iteration

Unfortunately few problems can be solved by simple conjectures. As a last resort one needs onerous effort on value functions. Suppose that the world ends after some finite peroid T. Then surely

$$V(k_{T+1}) = 0,$$

as well as

$$c_T = k_T^\alpha, \text{ and } k_{T+1} = 0.$$

Apply these in Bellman equation,

$$V(k_T) = \ln k_T^\alpha + \beta V(k_{T+1}) = \ln k_T^\alpha.$$

For one period backward,

$$
\begin{aligned}
V(k_{T-1}) &= \max_{c_{T-1},k_T} \{\ln(c_{T-1}) + \beta V(k_T)\} \\
&= \max_{c_{T-1},k_T} \{\ln(c_{T-1}) + \beta \ln k_T^\alpha\} \\
s.t. \quad & k_T = k_{T-1}^\alpha - c_{T-1}.
\end{aligned}
$$

This is simply a two-period intertemporal optimization with an equality constraint. Using Lagrangian

$$\mathscr{L} = \ln(c_{T-1}) + \beta \ln k_T^{\alpha} + \lambda \left[k_{T-1}^{\alpha} - c_{T-1} - k_T\right],$$

first order conditions give

$$\begin{aligned}
\frac{\partial \mathscr{L}}{\partial c_{T-1}} &= \frac{1}{c_{T-1}} - \lambda = 0, \\
\frac{\partial \mathscr{L}}{\partial k_T} &= \alpha\beta \frac{k_T^{\alpha-1}}{k_T^{\alpha}} - \lambda = \alpha\beta \frac{1}{k_T} - \lambda = 0, \\
\frac{\partial \mathscr{L}}{\partial \lambda} &= k_{T-1}^{\alpha} - c_{T-1} - k_T = 0.
\end{aligned}$$

Solve to get

$$\begin{aligned}
c_{T-1} &= \frac{1}{1+\alpha\beta} k_{T-1}^{\alpha}, \\
k_T &= \frac{\alpha\beta}{1+\alpha\beta} k_{T-1}^{\alpha}.
\end{aligned}$$

Then $V(k_{T-1})$ can be expressed as

$$\begin{aligned}
V(k_{T-1}) &= \ln\left(\frac{1}{1+\alpha\beta} k_{T-1}^{\alpha}\right) + \beta \ln\left(\frac{\alpha\beta}{1+\alpha\beta} k_{T-1}^{\alpha}\right)^{\alpha} \\
&= \alpha\beta \ln(\alpha\beta) - (1+\alpha\beta)\ln(1+\alpha\beta) + (1+\alpha\beta)\ln k_{T-1}^{\alpha}.
\end{aligned}$$

Again take one period backward,

$$\begin{aligned}
V(k_{T-2}) = \max_{c_{T-2}, k_{T-1}} & \{\ln(c_{T-2}) + \beta V(k_{T-1})\} \\
& s.t.\ k_{T-1} = k_{T-2}^{\alpha} - c_{T-2},
\end{aligned}$$

and the same procedure applies. After several rounds you may find that for time t long before T the value function converges to

$$\begin{aligned}
V(k_t) = \max_{c_t, k_{t+1}} & \left\{ \ln c_t + \beta \left[\frac{1}{1-\beta} \left(\ln(1-\alpha\beta) + \frac{\alpha\beta}{1-\alpha\beta} \ln \alpha\beta \right) \right.\right. \\
& \left.\left. + \frac{\alpha}{1-\alpha\beta} \ln k_{t+1} \right] \right\} \\
s.t. \quad & k_{t+1} = k_t^{\alpha} - c_t.
\end{aligned}$$

As before since $V(k_t)$ is the maximized value the first order condition with respect to k_{t+1} still holds

$$-\frac{1}{c_t}+\frac{\alpha\beta}{1-\alpha\beta}\frac{1}{k_{t+1}}=0,$$

this yields

$$\begin{aligned} c_t &= (1-\alpha\beta)k_t^{\alpha} \\ k_{t+1} &= \alpha\beta k_t^{\alpha}. \end{aligned}$$

Although this solution method is very cumbersome and computationally demanding since it rests on brutal-force iteration of the value function, it has the advantage that it always works if the solution exists. Actually, the convergence of the value function is not incidential. Appendix A.2 tells us that the convergence result is always achieved as long as the value function contains a *contraction mapping* (which works for most of dynamic optimization problems under the neoclassical assumptions). Such result is crucial for both theory and application. In theory, it ensures that a unique equilibrium solution (the *fixed point*) exists so that we can say what happens in the long run; in application, it implies that even we start iteration from an arbitrary value function, the value function will finally converge to the *true* one. Therefore, in practice when the value function is hardly solvable in an analytical way, people usually set up the computer program to perform the iteration and get a numerical solution. An example in Appendix B charaterizes a typical solution procedure using the popular computational softwares, and shows some tips in designing your own numerical programs.

5 Extensions

5.1 Extension 1: Dynamic Programming under Uncertainty

Consider a general discrete-time optimization problem

$$\begin{aligned} \max_{\{c_t,k_{t+1}\}_{t=0}^{+\infty}} \; & E_0\left[\sum_{t=0}^{+\infty}\beta^t u(c_t)\right] \\ s.t. \; & k_{t+1}=z_t f(k_t)-c_t, \end{aligned}$$

in which the production $f(k_t)$ is affected by an i.i.d process $\{z_t\}_{t=0}^{+\infty}$ (technology shock, which realizes at the beginning of each period t) meaning that such shock varies over time, but its deviations in different periods are uncorrelated (think about the weather for the farmers). Now the agent has to maximize the expected utility over time because future consumption is uncertain.

Since technology shocks realize at the beginning of each period, the value of total output is known when consumption takes place and when the end-of-period capital k_{t+1} is accumulated.

The state variables are now k_t and z_t. The control variables are c_t. Similar as before, set up the Bellman equation as

$$\begin{aligned} V(k_t, z_t) &= \max_{c_t, k_{t+1}} \{u(c_t) + \beta E_t [V(k_{t+1}, z_{t+1})]\} \\ s.t. \quad & k_{t+1} = z_t f(k_t) - c_t. \end{aligned}$$

Let's apply the solution strategies introduced before and see whether they work.

Step 1 The first order condition with respect to k_{t+1} gives

$$-u'(c_t) + \beta E_t \left[\frac{dV(k_{t+1}, z_{t+1})}{dk_{t+1}} \right] = 0. \tag{22}$$

Think why it is legal to take derivative within expectation operator.

Step 2 By Envelope Theorem differentiating $V(k_t, z_t)$ with respect to k_t gives

$$\frac{dV(k_t, z_t)}{dk_t} = u'(c_t) z_t f'(k_t). \tag{23}$$

Step 3 Take one step forward for (23) and apply it into (22), then we get

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

Suppose that

$$\begin{aligned} f(k_t) &= k_t^\alpha, \\ u(c_t) &= \ln c_t. \end{aligned}$$

Then Euler equation becomes

$$\frac{1}{c_t} = \beta E_t \left[\frac{1}{c_{t+1}} z_{t+1} \alpha k_{t+1}^{\alpha - 1} \right].$$

In deterministic case our solutions were

$$\begin{aligned} c_t &= (1 - \alpha\beta) k_t^\alpha, \\ k_{t+1} &= \alpha\beta k_t^\alpha. \end{aligned}$$

Now let's guess that under uncertainty the solution is of similar form such that

$$\begin{aligned} c_t &= (1-A) z_t k_t^{\alpha}, \\ k_{t+1} &= A z_t k_t^{\alpha}, \end{aligned}$$

and check whether it's true or false. The Euler equation becomes

$$\begin{aligned} \frac{1}{(1-A) z_t k_t^{\alpha}} &= \beta E_t \left[\frac{1}{(1-A) z_{t+1} k_{t+1}^{\alpha}} z_{t+1} \alpha k_{t+1}^{\alpha-1} \right] \\ &= \alpha\beta \left[\frac{1}{(1-A) k_{t+1}} \right] \\ &= \alpha\beta \left[\frac{1}{(1-A) A z_t k_t^{\alpha}} \right]. \end{aligned}$$

Therefore it's easily seen that

$$A = \alpha\beta,$$

which seems quite similar as before.

However since the consumption and capital stock are random variables, it's necessary to explore their properties by charaterizing corresponding distributions.

Assume that $\ln z_t \sim N\left(\mu, \sigma^2\right)$. Take log of the solution above,

$$\ln k_{t+1} = \ln \alpha\beta + \ln z_t + \alpha \ln k_t.$$

Apply this result recursively,

$$\begin{aligned} \ln k_t &= \ln \alpha\beta + \ln z_{t-1} + \alpha \ln k_{t-1} \\ &= \ln \alpha\beta + \ln z_{t-1} + \alpha(\ln \alpha\beta + \ln z_{t-2} + \alpha \ln k_{t-2}) \\ &\quad \ldots \\ &= \left(1 + \alpha + \alpha^2 + \ldots + \alpha^{t-1}\right) \ln \alpha\beta \\ &\quad + \left(\ln z_{t-1} + \alpha \ln z_{t-2} + \ldots + \alpha^{t-1} \ln z_0\right) \\ &\quad + \alpha^t \ln k_0. \end{aligned}$$

In the limit the mean of $\ln k_t$ converges to

$$\lim_{t \to +\infty} E_0 \left[\ln k_t\right] = \frac{\ln \alpha\beta + \mu}{1-\alpha}.$$

The variance of $\ln k_t$ is defined as

$$\begin{aligned}
\operatorname{var}[\ln k_t] &= E\left\{(\ln k_t - E[\ln k_t])^2\right\} \\
&= E\left\{\left(\left(1+\alpha+\alpha^2+\ldots+\alpha^{t-1}\right)\ln\alpha\beta + (\ln z_{t-1} + \alpha \ln z_{t-2}\right.\right. \\
&\quad \left. + \ldots + \alpha^{t-1}\ln z_0\right) + \alpha^t \ln k_0 - \left[\left(1+\alpha+\alpha^2+\ldots+\alpha^{t-1}\right)\ln\alpha\beta \\
&\quad \left.\left. + \left(1+\alpha+\ldots+\alpha^{t-1}\right)\mu + \alpha^t \ln k_0\right]\right)^2\right\} \\
&= E\left\{\left[(\ln z_{t-1} - \mu) + \alpha(\ln z_{t-2} - \mu) + \alpha^2(\ln z_{t-2} - \mu) + \ldots\right.\right. \\
&\quad \left.\left. + \alpha^{t-1}(\ln z_0 - \mu)\right]^2\right\} \\
&= E\left[\left(\sum_{i=1}^{t} \alpha^{i-1}(\ln z_{t-i} - \mu)\right)^2\right] \\
&= \sum_{i=1}^{t} \alpha^{2i-2} E\left[(\ln z_{t-i} - \mu)^2\right] \\
&\quad + \sum_{\forall i,j \in \{1,\ldots,t\} i \neq j} \alpha^{i-1}\alpha^{j-1} E\left[(\ln z_{t-i} - \mu)(\ln z_{t-j} - \mu)\right] \\
&= \frac{1-\alpha^{2t}}{1-\alpha^2}\operatorname{var}[\ln z_t] \\
&= \frac{1-\alpha^{2t}}{1-\alpha^2}\sigma^2,
\end{aligned}$$

or simply pass the variance operator through the sum and get

$$\begin{aligned}
\operatorname{var}[\ln k_t] &= \operatorname{var}[\ln z_{t-1}] + \alpha^2 \operatorname{var}[\ln z_{t-2}] + \ldots + \alpha^{2t-2}\operatorname{var}[\ln z_0] \\
&= \left(1+\alpha^2+\ldots+\alpha^{2t-2}\right)\sigma^2 \\
&= \frac{1-\alpha^{2t}}{1-\alpha^2}\sigma^2.
\end{aligned}$$

In the limit the variance of $\ln k_t$ converges to

$$\lim_{t\to+\infty} \operatorname{var}(\ln k_t) = \frac{\sigma^2}{1-\alpha^2}.$$

As a conclusion one can say that in the limit $\ln k_t$ converges to a distribution with mean $\frac{\ln\alpha\beta+\mu}{1-\alpha}$ and variance $\frac{\sigma^2}{1-\alpha^2}$.

Till now one may get the illusion that dynamic programming only fits discrete time. Now with slight modification we'll see that it works for continuous time problems as well. Consider a general continuous time optimization problem

$$\max_{c_t,k_t} \int_{t=0}^{+\infty} e^{-\rho t} u(c_t) dt$$
$$s.t.\ \dot{k}_t = \phi(c_t, k_t) = f(k_t) - c_t$$

in which we assume that $\phi(c_t, k_t)$ is quasi-linear in c_t only for simplicity.

Following Bellman's idea, for arbitrary $t \in [0, +\infty)$ define

$$V(k_t) = \max_{c_t,k_t} \int_{t}^{+\infty} e^{-\rho(\tau - t)} u(c_\tau) d\tau.$$

Now suppose that time goes from t to $t + \Delta t$, in which Δt is very small. Let's imagine what happened from t on. First $u(c_t)$ accumulates during Δt. Since Δt is so small that it's reasonable to think that $u(c_t)$ is nearly constant from t to $t + \Delta t$, and the accumulation of utility can be expressed as $u(c_t)\Delta t$. Second, from $t + \Delta t$ onwards the total utility accumulation is just $V(k_{t+\Delta t})$. Therefore $V(k_t)$ is just the sum of utility accumulation during Δt, and discounted value of $V(k_{t+\Delta t})$, i.e.

$$V(k_t) = \max_{c_t,k_t} \left\{ u(c_t)\Delta t + \frac{1}{1 + \rho\Delta t} V(k_{t+\Delta t}) \right\}$$

(Why $V(k_{t+\Delta t})$ is discounted by $\frac{1}{1+\rho\Delta t}$?). Rearrange both sides

$$\begin{aligned}
(1 + \rho\Delta t)V(k_t) &= \max_{c_t,k_t} \{u(c_t)(1 + \rho\Delta t)\Delta t + V(k_{t+\Delta t})\} \\
\rho\Delta t V(k_t) &= \max_{c_t,k_t} \{u(c_t)(1 + \rho\Delta t)\Delta t + V(k_{t+\Delta t}) - V(k_t)\} \\
\rho V(k_t) &= \max_{c_t,k_t} \left\{ u(c_t)(1 + \rho\Delta t) + \frac{V(k_{t+\Delta t}) - V(k_t)}{\Delta t} \right\},
\end{aligned}$$

and take limit

$$\rho V(k_t) = \lim_{\Delta t \to 0} \max_{c_t,k_t} \left\{ u(c_t)(1 + \rho\Delta t) + \frac{V(k_{t+\Delta t}) - V(k_t)}{\Delta t} \right\}.$$

Finally this gives

$$\begin{aligned}\rho V(k_t) &= \max_{c_t,k_t}\left\{u(c_t) + V'(k_t)\dot{k}_t\right\} \\ &= \max_{c_t,k_t}\left\{u(c_t) + V'(k_t)\phi(c_t,k_t)\right\}.\end{aligned}$$

Then you are able to solve it by any of those three approaches. Here we only try one of them.

Step 1 First order condition for the maximization problem gives

$$u'(c_t) + V'(k_t)\frac{\partial\phi(c_t,k_t)}{\partial c_t} = u'(c_t) - V'(k_t) = 0. \tag{24}$$

Step 2 Differentiating $V(k_t)$ gives

$$\rho V'(k_t) = V''(k_t)\phi(c_t,k_t) + V'(k_t)\frac{\partial\phi(c_t,k_t)}{\partial k_t},$$

that is,

$$\left[\rho - \frac{\partial\phi(c_t,k_t)}{\partial k_t}\right]V'(k_t) = V''(k_t)\phi(c_t,k_t) = V''(k_t)\dot{k}_t.$$

Take derivative of $V'(k_t)$ with respect to t

$$\frac{dV'(k_t)}{dt} = \dot{V}'(k_t) = V''(k_t)\dot{k}_t = \left[\rho - \frac{\partial\phi(c_t,k_t)}{\partial k_t}\right]V'(k_t),$$

and get

$$\frac{\dot{V}'(k_t)}{V'(k_t)} = \rho - \frac{\partial\phi(c_t,k_t)}{\partial k_t} = \rho - f'(k_t). \tag{25}$$

Step 3 Take derivative of (24) with respect to t and get

$$\frac{\dot{u}'(c_t)}{u'(c_t)} = \frac{u''(c_t)}{u'(c_t)}\dot{c}_t = \frac{\dot{V}'(k_t)}{V'(k_t)} = \rho - f'(k_t)$$

by (25), and further arrangment gives

$$
\begin{aligned}
-\frac{u''(c_t)c_t}{u'(c_t)}\frac{\dot{c}_t}{c_t} &= f'(k_t) - \rho \\
\frac{\dot{c}_t}{c_t} &= \sigma\left[f'(k_t) - \rho\right].
\end{aligned}
$$

Note that this is exactly the same solution as we got by the optimal control method.

6 Conclusion: Recursive Methods Revisited

Readers may have already the idea of how to solve dynamic programming problems using recursive methods. In general this consists of the following steps:

- Find recursive structure of the problem, i.e. define the value function such that the problem "repeats itself"in every period;
- Find the proper state (which capture the situation when the period starts) and control (which the decision maker wants to choose in the period) variables;
- Find the law of transition between periods, which maps the state and control variables of today into the ones of tomorrow.

Then you would get a full-fledged Bellman equation, and try to get what you are interested in out of it — normally this is simply a procedure of getting used to mathematics.

7 Readings

Ljungqvist and Sargent (2004), Chapter 3.

8 Bibliographic Notes

Dynamic programming methods can be traced back to Bellman (1957), and quite a few excellent textbooks can be referred to. Stokey, Lucas with Prescott (1989) provides a sound mathematical foundation for recursive methods, and Stachurski (2009) is by far the best introduction to stochastic dynamic programming and computational methods with economic applications, but one needs rich knowledge of modern analysis (e.g. Rudin (1976), or Kolmogorov and Fomin (1965)) to go through these two books. Ljungqvist and Sargent (2004) is an encyclopedic approach to the applications and becomes a standard configuration for beginners. Wälde (2010) provides a least demanding access to solution methods.

The classical text of Dixit and Pindyck (1994) gives in-depth extensions of dynamic problems under uncertainty, which is surely a pleasant reading for enthusiastic readers.

9 Exercises

9.1 Dynamic Programming with Externalities

This question concerns a representative agent economy with a continuum of consumers. Preferences are

$$\sum_{t=0}^{+\infty} \beta^t \ln(c_t),$$

in which c_t is consumption in period t. Each individual has the production technology

$$c_t + k_{t+1} = A k_t^{\alpha} K_t^{\gamma},$$

in which k_t is her own capital at the beginning of period t and K_t is the average stock of capital in the economy as a whole, and the parameters satisfy $0 < \alpha < 1$ and $\gamma > 0$. Each period, each agent chooses c_t and k_{t+1}, given k_t and K_t. In equilibrium, of course, it must be the case that $k_t = K_t$ for all t. Let

$$K_{t+1} = f(K_t)$$

be the *equilibrium* law of motion for capital in the economy. Let

$$K_{t+1} = g(K_t)$$

be the *socially optimal* law of motion for capital in the economy. The following questions require you to calculate both of these functions f and g.

a) Socially optimal allocation

(1) [A] Formulate the Bellman equation for the socially optimal allocation.
(2) [B] Solve for the value function.
(3) [B] Verify that the policy function takes the form

$$g(K) = \theta A K^{\alpha+\gamma}$$

and find the value of θ.

b) Equilibrium allocation

(1) [A] Formulate the Bellman equation for an equilibrium.
(2) [B] Solve for the value function.
(3) [B] Verify that the policy function takes the form

$$f(K) = \mu A K^{\alpha+\gamma}$$

and find the value of μ.

9.2 Dynamic Programming with a Kinked Production Function

Consider an economy with a single consumer whose preference are defined by the linear utility function:

$$\sum_{t=0}^{+\infty} \beta^t c_t.$$

There is a single firm which operates the production technology

$$y_t = \begin{cases} A(k_t - \overline{k}) + B\overline{k} & \text{for } k_t \geq \overline{k} \\ Bk_t & \text{for } k_t < \overline{k} \end{cases}.$$

The parameters of the utility and production functions satisfy:

$$A < \frac{1}{\beta} < B.$$

Capital evolves according to

$$k_{t+1} = i_t,$$

i.e. capital depreciates completely every period. The initial level of capital is given by k_0. The capital stock has to be non-negative; consumption and investment, on the other hand, are allowed to assume negative values (only for simplicity).

a)[A] Draw the production function.

b)[A] Provide a Bellman equation for the problem solved by a benevolent social planner.

c)[B] Determine the steady-state level of capital.

d)[C] Find the optimal law of motion for capital. Hint: compare the utility of consuming one unit of output now versus investing and consuming tomorrow. Distinguish the cases $k < \overline{k}$ and $k \geq \overline{k}$.

e)[C] Find the value function. Hint: No value function iteration etc. required – use your information on the optimal law of motion and compute the utility directly.

9.3 Neoclassical Growth Model with Vintage Capital

Consider an economy with a mass one of identical consumers whose preferences are defined by the utility function

$$\sum_{t=0}^{+\infty} \beta^t \ln(c_t),$$

in which c_t is consumption, and the parameter $\beta \in (0,1)$. Each consumer inelastically supplies one unit of labor. The production sector of the economy is subject exogenous productivity change. However, technical change is of the embodies kind such that a specific capital investment has been made to profit from any productivity increase. The frontier productivity A_t grows at the exogenous rate γ

$$A_{t+1} = (1+\gamma)A_t.$$

Capital that is first used at time t uses the frontier technology of time t; future productivity increases do not affect output derived from this capital. Capital can be used for two periods, and then depreciates completely. Thus, at time t two capital vintages k_{t-1} and k_t are in use, in which the index denotes the time of first usage, and the two production functions used at time t are

$$\begin{aligned} Y_t^{t-1} &= A_{t-1} k_{t-1}^{\alpha} \left(L_t^{t-1}\right)^{1-\alpha}, \\ Y_t^{t} &= A_t k_t^{\alpha} \left(L_t^{t}\right)^{1-\alpha}. \end{aligned}$$

Here L_t^{t-1} and L_t^t are the amounts of labor used with capital of vintage $t-1$ and t, respectively. Labor can be allocated freely between the two vintages. The feasibility condition for the goods market is

$$c_t + k_{t+1} = Y_t^{t-1} + Y_t^t.$$

a)[B] Formulate the optimization problem solved by a benevolent social planner in this economy.

b)[C] Derive the first order conditions, and for given level of capital, derive the optimal allocation of labor across the different vintages. What is the balanced growth rate of the economy?

References

Bellman, R. (1957): *Dynamic Programming*. Princeton: Princeton University Press.

Brock, W. A. and L. Mirman (1972): "Optimal Economic Growth and Uncertainty: The Discounted Case."*Journal of Economic Theory*, 4, June, 479-513.

Dixit, A. K. (1990): *Optimization in Economic Theory (2nd Ed.)*. New York: Oxford University Press.

Dixit, A. K. and R. S. Pindyck (1994): *Investment under Uncertainty*. Princeton: Princeton University Press.

Kolmogorov, A. N. and S. V. Fomin (1965): *Elements of the Theory of Functions and Functional Analysis*. New Jersey: Graylock Press.

Ljungqvist, L. and T. J. Sargent (2004): *Recursive Macroeconomic Theory (2nd Ed.)*. Cambridge: MIT Press.

Rudin, W. (1976): *Principles of Mathematical Analysis (3rd Ed.)*. New York: McGraw-Hill.

Stachurski, J. (2009): *Economic Dynamics: Theory and Computation*. Cambridge: MIT Press (*forthcoming*).

Stokey, N. L., R. E. Lucas with E. C. Prescott (1989): *Recursive Methods in Economic Dynamics*. Cambridge: Harvard University Press.

Wälde, K. (2010): *Applied Intertemporal Optimization*, mimeo, University of Mainz.

Appendix

A Useful Results of Mathematics

A.1 Envelope Theorem

Theorem A.1 *Suppose that value function $m(a)$ is defined as following:*

$$m(a) = \max_x f(x(a), a).$$

Then the total derivative of $m(a)$ with respect to a equals the partial derivative of $f(x(a), a)$ with respect to a, if $f(x(a), a)$ is evaluated at $x = x(a)$ that maximizes $f(x(a), a)$, i.e.

$$\frac{dm(a)}{da} = \left.\frac{\partial f(x(a), a)}{\partial a}\right|_{x=x(a)}.$$

Proof Since $m(a)$ is maximized value of $f(x(a), a)$ at $x = x(a)$, then

$$\frac{\partial f(x(a), a)}{\partial x} = 0$$

by the first order condition. Therefore the total derivative of $m(a)$ with respect to a is

$$\begin{aligned}\frac{dm(a)}{da} &= \frac{\partial f(x(a), a)}{\partial x}\frac{dx(a)}{da} + \frac{\partial f(x(a), a)}{\partial a} \\ &= \frac{\partial f(x(a), a)}{\partial a}\end{aligned}$$

since the first term is equal to 0. □

A.2 The Theoretical Foundation of Dynamic Programming: Some Serious Mathematical Concerns

Although using the method dynamic programming seems not too hard, we never thought about two fundamental questions concerning the prototype problem like (1):

(1) Does the solution exist at all?
(2) If yes, is it unique?

Answering these two questions needs some knowledge of *functional analysis*. Here are some brief results as following.

Definition A *metric space* (S, ρ) is a non-empty set S and a *metric*, or *distance* $\rho : S \times S \to \mathbb{R}$, which is defined as a mapping, $\forall x, y, v$, with

(1) $\rho(x, y) = 0 \Leftrightarrow x = y$,
(2) $\rho(x, y) = \rho(x, y)$, and
(3) $\rho(x, y) \leq \rho(x, v) + \rho(v, y)$.

For example, a plane $\left(\mathbb{R}^2, d_2\right)$ is a metric space, in which the metric $d_2 : \mathbb{R}^2 \times \mathbb{R}^2 \to \mathbb{R}$ is defined as

$$d_2(\mathbf{x}, \mathbf{y}) = \|\mathbf{x} - \mathbf{y}\|_2 = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2}, \forall \mathbf{x}, \mathbf{y} \in \mathbb{R}^2,$$

i.e. $d_2(\cdot)$, or $\|\cdot\|_2$, is just the Euclidean distance.

Definition A *norm* is a mapping $\mathbb{R}^n \ni x \mapsto \|x\| \in \mathbb{R}$ on $\mathbb{R}^n$, with

(1) $\forall x \in \mathbb{R}^n$, $\|x\| = 0 \Leftrightarrow x = 0$,
(2) $\forall x \in \mathbb{R}^n$, $\forall \alpha \in \mathbb{R}$, $\|\alpha x\| = |\alpha|\|x\|$ and
(3) $\forall x, y \in \mathbb{R}^n$,$\|x + y\| \leq \|x\| + \|y\|$.

In the definition of metric space, the set S is just arbitrary. It can be a subset of n dimensional space, i.e. $S \subseteq \mathbb{R}^n$, but it can also be a *function space* $B(X)$ — a set containing all (normally, bounded) functions mapping a set X to $\mathbb{R}$, $B : X \to \mathbb{R}$. Then we define a *supremum norm* on such function space

$$d_\infty = \|f - g\|_\infty = \sup_{x \in X} |f(x) - g(x)|, \forall f, g \in B(X),$$

and this metric space of bounded functions on X with supremum norm is denoted by $(B(X), d_\infty)$.

Having defined all the necessary jargons, we continue with a special mapping.

Definition Suppose a metric space (S, ρ) and a function $T : S \to S$ mapping S to itself. T is a *contraction mapping* with *modulus* β, if $\exists \beta \in (0, 1)$, $\rho(Tx, Ty) \leq \beta\rho(x, y)$, $\forall x, y \in S$.

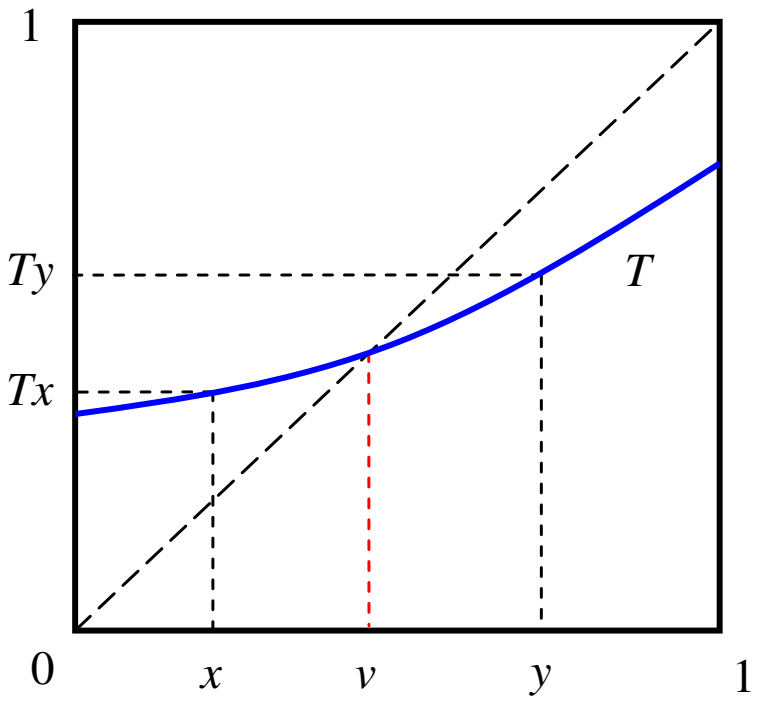

Fig. A.1. Contraction Mapping

An example in Figure A.1 shows a contraction mapping $T : (0, 1) \to (0, 1)$. The distance between images Tx and Ty is less than $|y - x|$. One may notice that under a contraction mapping like this, a *fixed point* $v \in S = (0, 1)$ exists such that $Tv = v$. Indeed, the following theorem tells us that this is a general phenomenon.

Theorem A.2 *(Contraction Mapping Theorem) If (S, ρ) is a complete metric space and $T : S \to S$ is a contraciton mapping with modulus β, then*

(1) T has a unique fixed point $v \in S$, and
(2) $\forall v_0 \in S$, $\rho(T^n v_0, v) \leq \beta^n \rho(v_0, v)$, $n \in N$.

T^n means that the mapping is applied for n times. But what does the Theorem imply for our questions on dynamic programming? Well, look at the prototype problem

$$V(k_t) = \max_{c_t, k_{t+1}} \{u(c_t) + \beta V(k_{t+1})\}. \tag{A.1}$$

The right hand side is just a mapping of function $V(\cdot)$, mapping the function space to itself. And the equilibrium solution making $V = TV$ is simply a fixed point of the mapping! Now the Contraction Mapping Theorem tells us that a unique fixed point exists if the mapping is a contraction mapping, therefore, if we want to say that there is a unique solution for the prototype problem, we have to make sure that the mapping in (A.1) is a contraction mapping.

However, showing a mapping to be a contraction one directly by definition is usually tricky. Fortunately, the following theorem makes the task more tractable.

Theorem A.3 *(Blackwell's sufficient conditions for a contraction) Suppose $X \subseteq \mathbb{R}^n$ and $B(X)$ is the function space for all bounded functions $f : X \to \mathbb{R}$ with supremum norm $\|\cdot\|_\infty$. If a mapping $T : B(X) \to B(X)$ satisfies*

(1) (Monotonicity condition) $\forall f, g \in B(X)$ and $\forall x \in X$ with $f(x) \leq g(x)$ implies $(Tf)(x) \leq (Tg)(x)$, $\forall x \in X$;
(2) (Discounting condition) $\exists \beta \in (0, 1)$ such that

$$[T(f + a)](x) = f(x) + a \leq (Tf)(x) + \beta a, \forall f \in B(X), a \geq 0, x \in X,$$

then T is a contraction mapping with modulus β.

Now we can show that our prototype problem of dynamic programming satisfies Blackwell's sufficient conditions for a contraction, therefore there exists a unique fixed point for the mapping. Suppose that we are going to solve the following dynamic optimization problem with exact utility and production functions,

$$\begin{aligned} V(k) = \max_{k'} \left\{ \frac{c^{1-\theta}}{1-\theta} + \beta V(k') \right\} &= \max_{k'} \left\{ \frac{[Ak^\alpha + (1-\delta)k - k']^{1-\theta}}{1-\theta} + \beta V(k') \right\} \\ s.t. \quad c + k' &= Ak^\alpha + (1-\delta)k, \end{aligned}$$

in which we write k and k' instead of k_t and k_{t+1} for simplicity, and the right hand side defines the mapping T. Since k takes its maximum value when $c = 0$, k is thus bounded above by $\overline{k}$ such that

$$
\begin{aligned}
0 + \overline{k} &= A\overline{k}^{\alpha} + (1-\delta)\overline{k}, \\
\overline{k} &= \left(\frac{A}{\delta}\right)^{\frac{1}{1-\alpha}}.
\end{aligned}
$$

Therefore define the state space $X \subseteq \left[0, \overline{k}\right]$ as a complete subspace of $\mathbb{R}$, and $B(X)$ the function space of all bounded continuous functions on X with supremum norm. Then we need to show that the mapping $T : B(X) \to B(X)$ in the complete (why?) metric space $(B(X), d_{\infty})$ satisfies Blackwell's sufficient conditions for a contraction.

Check the monotonicity condition. Let $f(x) \leq g(x)$, $\forall x \in X$, then

$$
\begin{aligned}
Tf(k) &= \max_{k'} \left\{ \frac{[Ak^{\alpha} + (1-\delta)k - k']^{1-\theta}}{1-\theta} + \beta f(k') \right\} \\
&\leq \max_{k'} \left\{ \frac{[Ak^{\alpha} + (1-\delta)k - k']^{1-\theta}}{1-\theta} + \beta \left[f(k') + g(k') - f(k') \right] \right\} \\
&= \max_{k'} \left\{ \frac{[Ak^{\alpha} + (1-\delta)k - k']^{1-\theta}}{1-\theta} + \beta g(k') \right\} \\
&= Tg(k).
\end{aligned}
$$

Check the discounting condition.

$$
\begin{aligned}
\left[T(f+a)\right](k) &= \max_{k'} \left\{ \frac{[Ak^{\alpha} + (1-\delta)k - k']^{1-\theta}}{1-\theta} + \beta f(k') + \beta a \right\} \\
&= Tf(k) + \beta a.
\end{aligned}
$$

Both conditions hold. Therefore the dynamic optimization problem has a unique equilibrium solution.

B Numerical Solution Using MATLAB

We start this section from a simple example. Consider the following social planner's problem:

$$
\begin{aligned}
&\max_{\{c_t, k_t\}_{t=0}^{+\infty}} \sum_{t=0}^{*\infty} \beta^t \ln c_t \\
&\quad s.t.\ k_{t+1} = Ak_t^{\alpha} - c_t - \delta k_t.
\end{aligned}
$$

One can easily write down its Bellman equation as following:

$$V(k) = \max_{k'} \{\ln c + \beta V(k')\}$$
$$s.t. \quad c + k' = Ak^{\alpha} + (1-\delta)k,$$

in which we write k and k' instead of k_t and k_{t+1} for simplicity. You can solve for the Euler equation, which gives

$$\frac{1}{c} = \beta \frac{1}{c'} \left[A\alpha k'^{\alpha-1} + (1-\delta) \right].$$

In steady state $c = c'$, and we insert the values for the other parameters, $\beta = 0.6$, $A = 20$, $\alpha = 0.3$, and $\delta = 0.5$, then the steady state level of capital stock k^* is around 10.38.

Also note that the highest value that k can achieve in the steady state is under $c = 0$, i.e. $0 + \overline{k} = A\overline{k}^{\alpha} + (1-\delta)\overline{k}$. $\overline{k}$ is approximately 12, therefore we take 12 as the upper bound of k in the program.

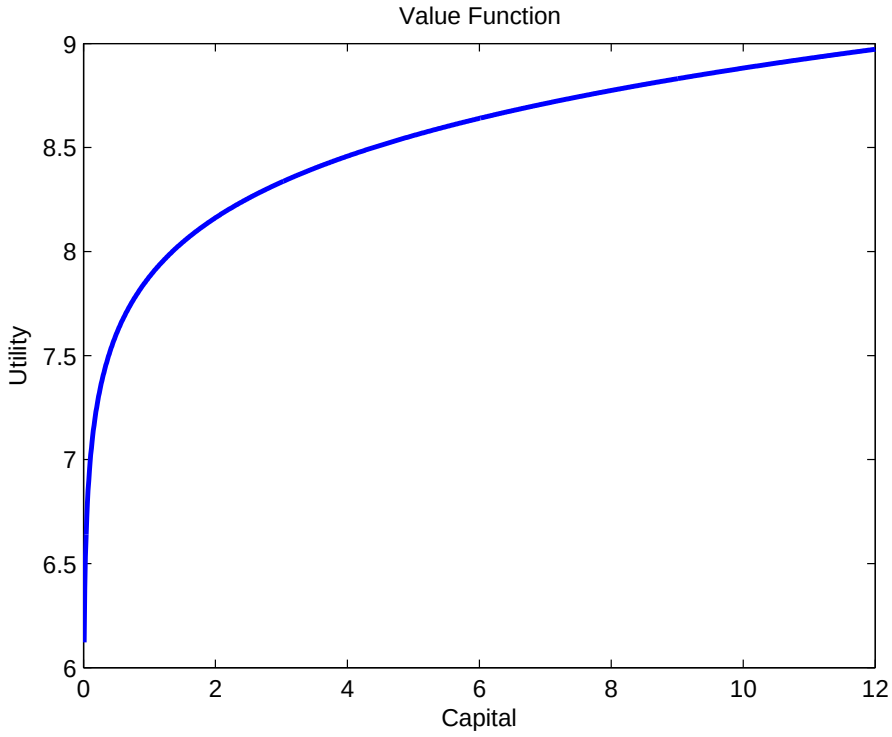

Fig. B.1. Plotting the Value Function

To make the computer work for you we borrow the idea of Section 4.2.2. Suppose that we start from the end of the world such that $V(k') = 0$ for any value k', then by backward induction we can compute the value function $V(k)$ for one period backward. If we repeat the same procedure for sufficiently many rounds $V(k)$ would eventually converge to the true value function, i.e. the distance between $V(k)$s in two successive repeats converges to zero. To make the programming easier, in practice people simply feed the discrete, rather than continuous values of k to computer (`incr=0.01` in this example) and let the machine do the

tedious computation of $V(k)$ for each value of k. FIGURE B.1 shows the final result of $V(k)$, and FIGURE B.2 plot each k' for its corresponding k. Steady state is obtained where $k = k'$, which is around 10.38 as we calculated. Finally, to make it interesting, FIGURE B.3 captures the complete process of convergence.

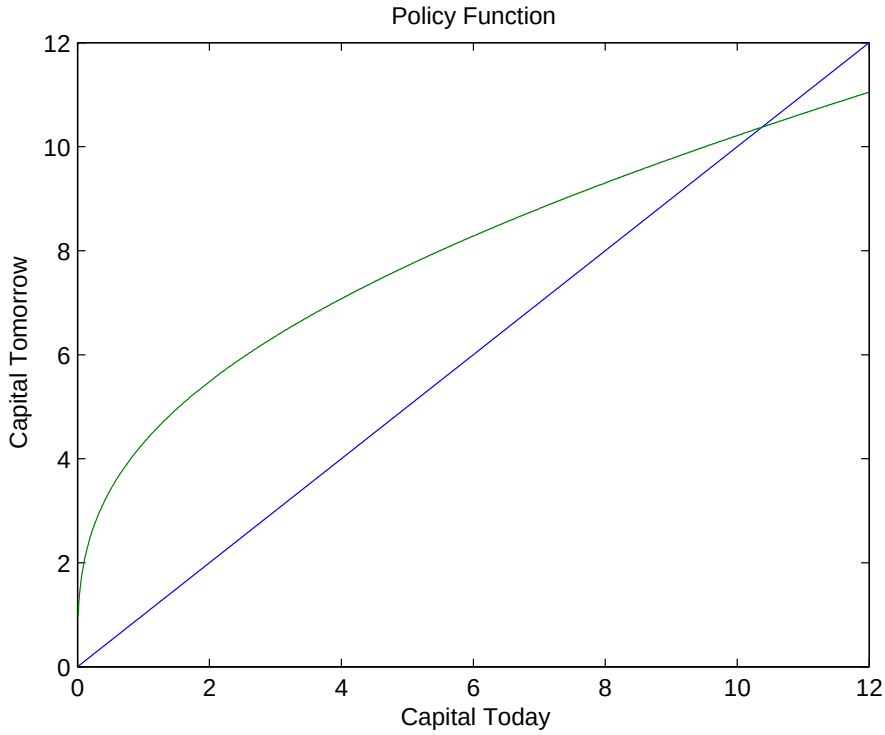

Fig. B.2. FINDING THE STEADY STATE

The source code(tested in MATLAB R2007a as well as GNU Octave 2.9.13, on the platforms of Microsoft Windows Vista and XP):

```
% iterate.m
% Simple example of value function iteration
clear all;
% Parameter values
beta=0.6;
A=20;
alpha=0.3;
delta=0.5;
% Technical parameters
incr=0.01; % Increment in the capital grid
maxk=12; % Maximum value for capital
maxcrit=10 ^(-5); % Converge criterion
% Setting up the capital grid
kgrid=[incr:incr:maxk];
```

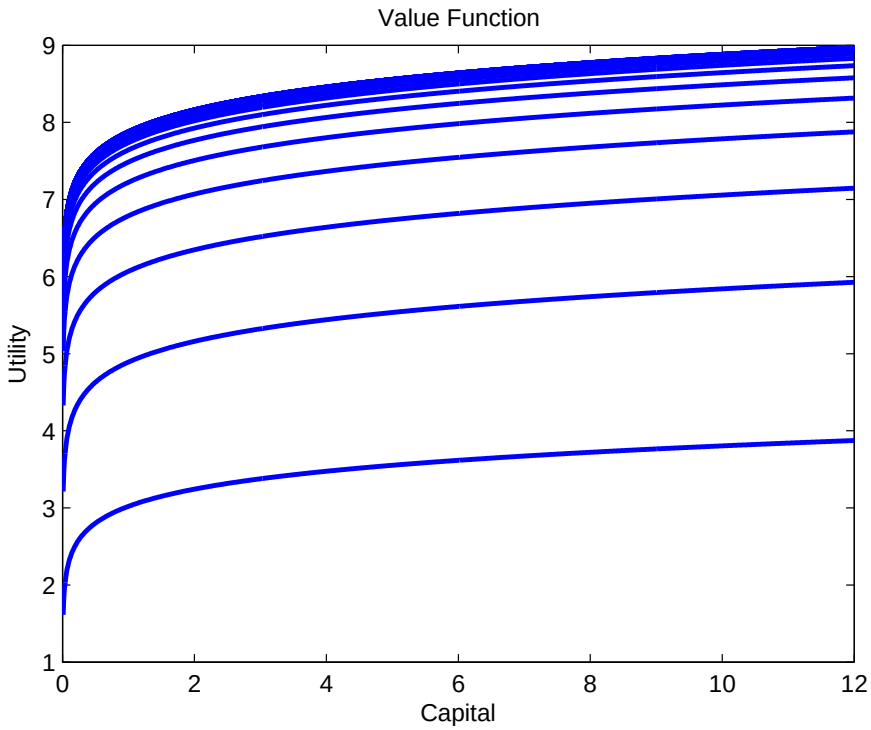

Fig. B.3. The Convergence Process

```
n=length(kgrid);
% Possible consumption values depending on k and k'; this is an n-by-n matrix
% Each column is different k, each row is different k'
cons=ones(n,1)*((1-delta)*kgrid+A*kgrid.^alpha)-kgrid'*ones(1,n);
% Throwing out negative values and taking logs to get period utilities
util=log(max(cons,0.00001*ones(n,n)));
% The initial guess for the value function
v=zeros(1,n);
% crit is the test function
crit=1;
while crit>maxcrit;
% The actual maximization step; g stores the optimizing index
[vnew g]=max(util+beta*v'*ones(1,n));
crit=max(abs(v-vnew));
v=vnew;
end;
% Plotting Figures
figure;
pl=plot(kgrid,v);
hold on;
set(pl,'LineWidth',2);
xlabel('Capital');
ylabel('Utility');
```

```
title('Value Function');
hold off;
figure;
plot(kgrid,[kgrid; incr*g]);
hold on;
set(pl,'LineWidth',2);
xlabel('Capital Today');
ylabel('Capital Tomorrow');
title('Policy Function');
hold off;
```