

# The Growth Model: Discrete Time Dynamic Programming

Prof. Lutz Hendricks

Econ720

September 17, 2024

# The Issue

We solved the growth model in **sequence language**.

- ▶ the solution is a sequence of objects that satisfies a bunch of difference equations

An alternative: **recursive** formulation.

- ▶ dynamic programming
- ▶ the solution is a set of functions

# Dynamic Programming: An Informal Introduction

The basic idea of DP is to transform a many period optimization problem into a static problem.

To do so, we summarize the entire future by a **value function**.

The value function

- ▶ tells us the maximum utility obtainable from tomorrow onwards for any value of the state variables.
- ▶ an **indirect utility function**

## 2. Simplest example

A household lives for two periods.

$$\max u(c) + \beta v(c'_1, c'_2) \quad (1)$$

subject to

$$s' = e + Rs - c \quad (2)$$

$$c'_1 + pc'_2 = Rs' \quad (3)$$

Notational convention: prime means tomorrow.

# Lagrangian

$$\mathcal{L} = u(e + Rs - s') + \beta v(Rs' - pc'_2, c'_2) \quad (4)$$

FOC:

$$u'(c) = \beta v_1(\cdot') R \quad (5)$$

$$v_1(\cdot') p = v_2(\cdot') \quad (6)$$

Solution:  $c, c'_1, c'_2, s'$  that solve

- ▶ 2 FOCs
- ▶ 2 budget constraints

# Dynamic Programming

The idea:

- ▶ solve one period at a time
- ▶ summarize the entire future with a “value function”  
(an indirect utility function)

If I know the value of saving, I can solve the first period problem.

## 2.1. Tomorrow's Problem

We solve this **backwards**, starting from the last period (“backwards induction”).

The last period problem is a simple static one:

$$\max v(Rs - pc_2, c_2) \quad (7)$$

- ▶ All we need to know about the past is saving  $s$  (assets at the start of this period)

FOC:

- ▶ The same as the static condition from the Lagrangian

$$v_1(c_1, c_2)p = v_2(c_1, c_2) \quad (8)$$

## Decision Rules

The FOC implicitly defines two decision rules of the form  $c_j(s)$   
Indirect utility is then also just a function of  $s$ :

$$W(s) = v(c_1(s), c_2(s)) \quad (9)$$

$$= \max_{c_2} v(Rs - pc_2, c_2) \quad (10)$$

We call  $s$  the **state variable** and  $W$  the **value function**.



## Log utility

Assume log utility tomorrow:

$$v(c_1, c_2) = \alpha \ln c_1 + (1 - \alpha) \ln(c_2) \quad (11)$$

Then the static condition becomes

$$p\alpha/c_1 = (1 - \alpha)/c_2 \quad (12)$$

or

$$pc_2 = \frac{1 - \alpha}{\alpha} c_1 \quad (13)$$

With log utility, expenditure shares are constant ( $\alpha$  for  $c_1$  and  $1 - \alpha$  for  $c_2$ ).

Consumption levels:

$$c_1 = \alpha Rs \quad (14)$$

$$c_2 = (1 - \alpha) Rs/p \quad (15)$$

## Value function

Then we can compute tomorrow's value function:

$$W(s) = \alpha \ln(\alpha R s) + (1 - \alpha) \ln((1 - \alpha) R s / p) \quad (16)$$

with marginal utility of wealth

$$W'(s) = \alpha / s + (1 - \alpha) / s = 1 / s \quad (17)$$

Solution:

- ▶ value function  $W(s)$  and policy functions  $c_j = f_j(s)$
- ▶ policy functions maximize  $W(s)$  point by point
- ▶  $W$  is the max of the RHS

## 2.2. Today's problem

We can now write today's problem as

$$V(s) = \max_{s'} u(s' - e) + \underbrace{\beta [\alpha \ln(\alpha R s') + (1 - \alpha) \ln((1 - \alpha) R s' / p)]}_{W(s')} \quad (18)$$

with FOC

$$u'(c) = \beta W'(s') = \beta / s' \quad (19)$$

Solution:

- ▶ value function  $V(s)$  and policy function  $s' = g(s)$
- ▶  $g$  maximizes Bellman equation given  $W$
- ▶  $V$  is the max of the RHS

## Cross check

Check against the Euler equation:

$$u'(c) = \beta v_1(.) R \quad (20)$$

$$= \beta R \frac{\alpha}{\alpha R s'} = \beta / s' \quad (21)$$

Both solutions given (of course) the same result.

## 2.3. Backward Induction I

Now consider a more general finite horizon problem

$$\max \sum_{t=1}^T u(c_t) \tag{22}$$

subject to  $k_{t+1} = Rk_t - c_t$  and  $k_{T+1} \geq 0$ .

We again solve it backwards, starting from the last period.

## Last Period

Consider the last date  $t = T$ .

The household cannot save:  $k_{T+1} = 0$

Continuation value:  $V(k, T+1) = 0$

- The problem is static: just eat all income

Terminal value:

$$V(k, T) = u(Rk) \quad (23)$$

## Backward Induction

Now step back to  $t = T - 1$

$$V(k, T - 1) = \max u(Rk - k') + \underbrace{\beta u(Rk')}_{V(k', T)} \quad (24)$$

We can (in principle) solve for  $V(k, T - 1)$

Now step back to  $t = T - 2$ , etc.

Bellman equation for any period:

$$V(k, t) = \max u(Rk - k') + \beta V(k', t + 1) \quad (25)$$

# Backward Induction

This is mainly useful for numerically solving the problem.  
Sometimes, one can solve finite horizon problems analytically  
(see Huggett et al. (2006) for an example).



# Infinite Horizon

Conceptually, we do exactly the same thing.

But now we don't have a last period that would solve for  $W(s')$  on the RHS of the Bellman equation.

Instead, we impose that tomorrow's value function is the same as today's:

$$V(s) = W(s) \tag{26}$$

This works because the problem is **stationary**

- ▶ every period is the same, except for the value of the state variable  $s$

### 3. The Growth Model: Planner's Problem

# Infinite Horizon Problem

Suppose we solve the planner's problem with starting date  $t^*$ :

$$V = \max_{\{c_t, k_t\}} \sum_{t=t^*}^{\infty} \beta^{t-t^*} u(c_t) \quad (27)$$

subject to  $k_{t+1} = f(k_t) - c_t \quad \forall t$ .

Call the optimal solution  $c_t^*, k_t^*$  and the implied lifetime utility  $V$ .

# Value function

Claim: The only fact that we need to know to figure out  $V$  is  $k_{t^*}$

- ▶ other past choices do not show up in preferences or constraints
- ▶  $k_{t^*}$  is the **state variable** of the problem.

Therefore, we can define the **value function** (indirect utility function)

$$V(k_{t^*}) = \sum_{t=t^*}^{\infty} \beta^{t-t^*} u(c_{t^*}) \quad (28)$$

$$= \max \sum_{t=t^*}^{\infty} \beta^{t-t^*} u(f(k_t) - k_{t+1}) \quad (29)$$

# Stationarity

Claim:  $V(k_{t^*})$  does not depend on  $t^*$ .

- ▶ Compare the value functions obtained from the problems starting at  $t^*$  and at  $t^* + 1$ .
- ▶ They are the same functions.
- ▶ That is, solving the problem yields the same value function regardless of the starting date.

Such a problem is called **stationary**.

- ▶ Not all optimization problems have this property.
- ▶ For example, if the world ends at some finite date, then the problem at  $t^* + 1$  looks different from the problem at  $t^*$ .

# Time consistency

- ▶ What if we start the problem at  $t^* + 1$ ?
- ▶ Would the planner want to change his optimal choices of  $k_{t^*+2}, k_{t^*+3}$ , and so on?
- ▶ The answer is obviously “no,” ... although I won’t prove this just yet.
- ▶ A problem with this property is known as **time consistent**:
  - ▶ Give the decision maker a choice to change his mind at a later date and he will choose the same actions again.
- ▶ Not all optimization problems have this property.
  - ▶ For example, changing the specification of discounting easily destroys time consistency (self-control problems arise).

# Recursive structure

Now comes the key insight:

$$V(k_{t^*}) = u(c_{t^*}^*) + \beta \left[ \sum_{t=t^*+1}^{\infty} \beta^{t-(t^*+1)} u(c_t^*) \right] \quad (30)$$

$$= u(c_{t^*}^*) + \beta V(k_{t^*+1}^*) \quad (31)$$

$$= \max_{k_{t^*+1}} u(f(k_{t^*}^*) - k_{t^*+1}^*) + \beta V(k_{t^*+1}^*) \quad (32)$$

We have

- ▶ one term reflecting current period utility
- ▶ a second term summarizing everything that happens in the future, given optimal behavior, as a function of  $k_{t^*+1}^*$ .

## Recursive structure

Since this equation holds for any arbitrary start date, we may drop date subscripts.

Unfortunate convention in macro:

- ▶ no subscript = today
- ▶ prime = tomorrow:  $k' = k_{t+1}$

This yields a **Bellman equation**:

$$V(k) = \max_{k'} u(f(k) - k') + \beta V(k')$$

Claim: **Solving the DP is equivalent to solving the original problem** (the Lagrangian).

- ▶ We will see conditions when this is true later.



## Recursive structure

The convenient part of this is: we have transformed a multiperiod optimization problem into a two period (almost static) one.

If we knew the value function, solving this problem would be trivial.

The bad news is that we have transformed an algebraic equation into a **functional equation**.

The solution of the problem is a value function  $V$  and an optimal policy function

$$c = \phi(k)$$

Note that  $c$  cannot depend on anything other than  $k$ , in particular not on  $k$ 's at other dates, because these don't appear in the Bellman equation.

# Solution

A solution to the planner's problem is now a pair of functions

- ▶ value function  $V(k)$
- ▶ policy function  $\phi(k)$

These solve the Bellman equation in the following sense.

1. Given  $V(k)$ , setting  $c = \phi(k)$  solves the max part of the Bellman equation.
2. Given that  $c = \phi(k)$ , the value function solves

$$V(k) = u(\phi(k)) + \beta V(f(k) - \phi(k))$$

## Solution: Intuition

Given  $V(k)$ , setting  $c = \phi(k)$  solves the max part of the Bellman equation.

This means:

Point by point, for each  $k$ :

$$\phi(k) = \arg \max_c u(c) + \beta V(f(k) - c) \quad (33)$$

$\phi(k)$  simply collects all the optimal  $c$ 's – one for each  $k$ .

## Solution: Intuition

$$V(k) = u(\phi(k)) + \beta V(f(k) - \phi(k))$$

Note that this uses the optimal policy function for  $c$ .

Think of the Bellman equation as a mapping in a function space:

$$V^{n+1} = T(V^n) = \max u(c) + \beta V^n(f(k) - c)$$

Given an input argument  $V^n$  the mapping produces an output arguments  $V^{n+1}$ .

The solution to the Bellman equation is the  $V$  that satisfies  $V = T(V)$ .

► a fixed point.

## 4. When Can DP Be Used?

The short answer:

If the problem can be written in a **certain format** (next slide)

and if **mild conditions** are met,

then

solving the Dynamic Problem is **equivalent** to solving the sequence problem.

Details are in the slides about Dynamic Programming  
(which we won't go through in class)

# Generic Sequence Problem

$$\begin{aligned} V^*(x(0)) &= \max_{\{x(t+1)\}_{t=0}^{\infty}} \sum_{t=0}^{\infty} \beta^t U(x(t), x(t+1)) \\ &\quad \text{subject to} \\ x(t+1) &\in G(x(t)) \\ x(0) &\text{ given} \end{aligned}$$

$x(t) \in X \subset \mathbb{R}^k$  is the set of allowed states.

The correspondence  $G: X \rightrightarrows X$  defines the constraints.

A solution is a sequence  $\{x(t)\}$

## Mapping into the growth model

$$V^*(k(0)) = \max_{\{k(t+1)\}_{t=0}^{\infty}} \sum_{t=0}^{\infty} \beta^t U(f(k(t)) - k(t+1))$$

subject to

$$k(t+1) \in G(k(t)) = [0, f(k(t))]$$

$$k(t) \in X = \mathbb{R}^+$$

$$k(0) \text{ given}$$

# Recursive Problem

$$V(x) = \max_{y \in G(x)} U(x, y) + \beta V(y), \quad \forall x \in X$$

A solution is a policy function  $\pi : X \longrightarrow X$  and a value function  $V(x)$  such that

1.  $V(x) = U(x, \pi(x)) + \beta V(\pi(x)), \quad \forall x \in X$
2. When  $y = \pi(x)$ , now and forever, the max value is attained.



# The Main Point

This is the upshot:

*If it is possible to write the optimization problem in the **format** of the generic sequence problem  
and if mild **conditions** hold,  
then solving the sequence problem is **equivalent** to  
solving the dynamic program.*

# Reading

- ▶ Acemoglu (2009), ch. 6. Also ch. 5 for background material we will discuss in detail later on.
- ▶ Ljungqvist and Sargent (2004), ch. 3 (Dynamic Programming), ch. 7 (Recursive CE).
- ▶ Stokey et al. (1989), ch. 1 is a nice introduction.

## References I

- Acemoglu, D. (2009): *Introduction to modern economic growth*, MIT Press.
- Huggett, M., G. Ventura, and A. Yaron (2006): "Human Capital and Earnings Distribution Dynamics," *Journal of Monetary Economics*, 53, 265–290.
- Ljungqvist, L. and T. J. Sargent (2004): *Recursive macroeconomic theory*, 2nd ed.
- Stokey, N., R. Lucas, and E. C. Prescott (1989): "Recursive Methods in Economic Dynamics," .