MENU DU JOUR

Methods

1. Revision on convex programs
   - Finite dimensional
   - E.g. 2-good consumer problem
   - E.g. General Equilibrium and 2-period OLG model
2. Multi-period, finite-horizon programs ($T < \infty$)
3. $T \to \infty$?

Economics

Related Week 3 Tute

1. Deriving economic growth insights from optimal growth model
2. Relation to data
3. Jupyter / Python application

Optimal program

$$V(m,p) = \max_{(x,y)\in G(m,\,p)} U(x,y)$$

indirect utility function

$\equiv$

value function

e.g. Budget constraint

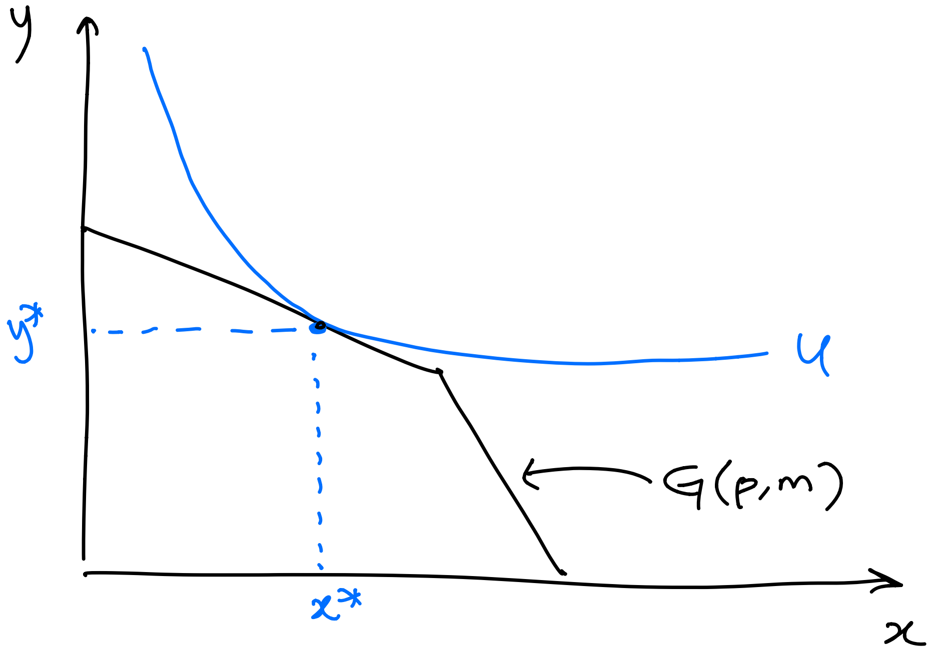
income

market price (relative)

parametric

FLASHBACK

If U, G define convex sets



Two-period OLG example

Suppose:

$$U(c_t, c_{t+1}) = \ln(c_t) + \beta \ln(c_{t+1})$$

$\in (0,1)$

Constraints:

$$s_t + c_t = \overbrace{w_t \cdot 1}^{\text{labour income}}$$

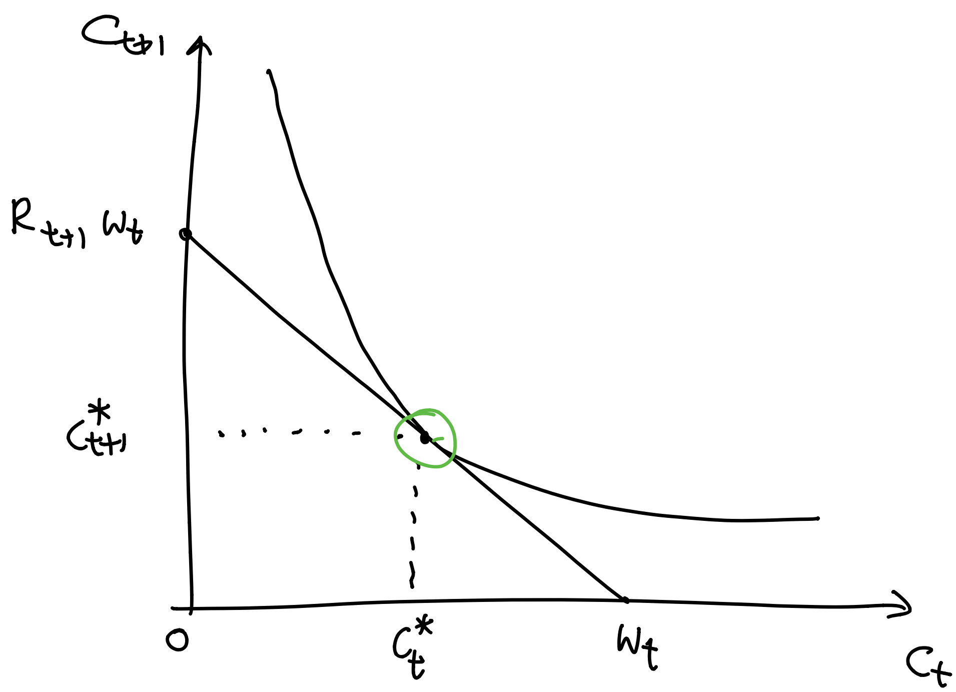
relative price of labour

labour supply

$$c_{t+1} = R_{t+1} s_t$$

$$\Rightarrow \underbrace{\frac{c_{t+1}}{R_{t+1}} + c_t = w_t \cdot 1}_{\text{P.V. lifetime B.C.}}$$

In pictures



Exercise

Show

$$c^*_t = c^*_t(w_t, R_{t+1})$$

income "m"

intertemporal relative price

$$c^*_{t+1} = c^*_{t+1}(w_t, R_{t+1})$$

Solution

$$V(w_t ; R_{t+1}) = \max_{c_t, c_{t+1}} \left\{ \ln(c_t) + \beta \ln(c_{t+1}) : \frac{c_{t+1}}{R_{t+1}} + c_t = w_t \right\}$$

We can write the constrained problem as the Lagrange auxiliary function:

$$L = \ln c_t + \beta \ln c_{t+1} - \lambda \left[c_t + \frac{c_{t+1}}{R_{t+1}} - w_t \right]$$

direction and magnitude of constraint's effect on objective value $U(c_t, c_{t+1})$

Necessary (and sufficient) conditions for max L:

FOC:

Shadow price of w_t (B.C.)

$$\frac{\partial L}{\partial c_t} = 0 \Rightarrow \underbrace{\lambda = \frac{1}{c_t}}$$ (1)

marginal utility of consumption c_t

$$\frac{\partial L}{\partial c_{t+1}} = 0 \Rightarrow \frac{\beta R_{t+1}}{c_{t+1}} = \lambda$$ (2)

$$\frac{\partial L}{\partial \lambda} = 0 \Rightarrow c_t + \frac{c_{t+1}}{R_{t+1}} = w_t$$ (3)

(1) into (2)

$$\boxed{\frac{1}{c_t} = \frac{\beta R_{t+1}}{c_{t+1}}}$$ (2')

$$\frac{1}{\beta}\frac{c_{t+1}}{c_t} = R_{t+1}$$

$MRS(c_t, c_{t+1})$

Note: (2') and (3) form two equations in two unknowns (c_t, c_{t+1}).

(2') into (3):

$$\frac{c_{t+1}}{\beta R_{t+1}} + \frac{c_{t+1}}{R_{t+1}} = w_t$$

$$\Rightarrow \quad \frac{c_{t+1}}{R_{t+1}}\left(\frac{1}{\beta}+1\right) = w_t$$

$$\Rightarrow \quad \boxed{c^*_{t+1} = \frac{\beta}{1+\beta} w_t R_{t+1}} \qquad (3')$$

Then plug (3') back into (2') to get

$$\boxed{c^*_t = \frac{1}{1+\beta} w_t}$$

The are the two demand functions we sought, of the form:

$$c^*_t = c^*_t(\underset{\substack{\uparrow \\ \text{income} \\ \text{"m"}}}{w_t}, \underset{\substack{\uparrow \\ \text{intertemporal} \\ \text{relative price}}}{R_{t+1}})$$

$$c^*_{t+1} = c^*_{t+1}(w_t, R_{t+1})$$

Consider $T > 1$ and $T < +\infty$

decision horizon $\equiv$ $(T+1)$ number of commodities

- $t \in \mathbb{N} := \{0, 1, \ldots\}$
- Consider social planner (or Robinson Crusoe's Corn Economy)
- Assume $\exists\, U$:

$$U(c_0, c_1, \ldots, c_T)$$

- Suppose $\exists$ technology:

$$k_1 = \underbrace{f(k_0)}_{\substack{\text{flow of} \\ \text{production} \\ \text{at } t=0}} - c_0$$

A recursive function:

$$k_{t+1} = f(k_t) - c_t \quad , \quad t \geq 0$$

concave f

Planner's problem:

$$V(k_0) = \max_{\{c_t, k_{t+1}\}_{t=0}^{T}} \left\{ U(c_0, c_1, \dots, c_T) : \begin{array}{l} k_1 = f(k_0) - c_0, \\ k_2 = f(k_1) - c_1, \\ \vdots \\ k_{T+1} = f(k_T) - c_T, \\ k_{T+1} \geq 0 \end{array} \right\}$$

initial KORN

same as writing

$(c_0, c_1, \dots, c_T)$

$(k_1, \dots, k_{T+1})$

A bit more structure... $\beta \in (0,1)$

$$U(\{c_t\}_{t=0}^{T}) = u(c_0) + \beta u(c_1) + \beta^2 u(c_2) + \dots + \beta^T u(c_T) + S_{T+1}$$

strictly convex

$\mathcal{L}$

Exercise

Show a characterization of optimal consumption / investment behavior.

Problem setup:

$$V(k_0) = \max_{\{c_t, k_{t+1}\}_{t=0}^{T}} \left\{ \sum_{t=0}^{T} \beta^t u(c_t) + S_{T+1}(k_{T+1}) : \begin{array}{r} k_{t+1} = f(k_t) - c_t, \\ t = 0, 1, \dots, T \end{array} \right\}$$

Or more verbosely ...

$$V(k_0) = \max_{\substack{(c_0, k_1 \\ c_1, k_2 \\ \dots \\ c_T, k_{T+1})}} \left\{ \begin{array}{l} u(c_0) + \beta u(c_1) + \beta^2 u(c_2) \\ + \dots + \beta^T u(c_T) + S_{T+1} \end{array} \right.$$

$$\begin{aligned}
\text{s.t.}\quad k_1 &= f(k_0) - c_0 && \lambda_0 \\
k_2 &= f(k_1) - c_1 && \lambda_1 \\
k_3 &= f(k_2) - c_2 && \vdots \\
&\vdots \\
k_{T+1} &= f(k_T) - c_T && \lambda_T \\
k_{T+1} &\geq 0 && \mu_T
\end{aligned}$$

FOCs (2T + 1 # conditions)

c_0 : $u'(c_0) = \lambda_0$ (C1)

c_1 : $\beta u'(c_1) = \lambda_1$ (C2)

c_2 : $\beta^2 u'(c_2) = \lambda_2$

$\vdots$

c_T : $\beta^T u'(c_T) = \lambda_T$

k_1 : $-\lambda_0 + \lambda_1 f'(k_1) = 0$ (K1)

k_2 : $-\lambda_1 + \lambda_2 f'(k_2) = 0$

$\vdots$

k_T $-\lambda_{T-1} + \lambda_T f'(k_T) = 0$

k_{T+1} : $\underset{\mu_T}{\underset{\parallel}{\lambda_T}} k_{T+1} = 0,$

Plug (C1), (C2) into (K1):

$$-u'(c_0) + \beta u'(c_1) f'(k_1) = 0$$

$$\Rightarrow \boxed{u'(c_0) = \beta u'(c_1) f'(k_1)}$$

$$\Rightarrow \underbrace{\frac{\beta u'(c_1)}{u'(c_0)}}_{MRS(c_0, c_1)} = \underbrace{\frac{1}{f'(k_1)}}_{MRT(c_0, c_1)}$$

"kind of internal relative price for the planner".

Can show:

$$(4) \quad \boxed{\begin{aligned} u'(c_t) &= \beta u'(c_{t+1}) f'(k_{t+1}) \\ k_{t+1} &= f(k_t) - c_t \end{aligned}}$$

Khademul says...

(5) $\boxed{k_{T+1} = 0}$ terminal condition.

show this from KKT complementary slackness condition on $k_{T+1} \geq 0$ constraint.

Also

(6) $\boxed{k_0 = \bar{k}_0}$ (known) initial condition

This is a second-order difference equation system in (c_t, k_t) with 2 boundary conditions.

This system solves for a sequence:

$$(c_0^*, c_1^*, \dots c_T^*, k_1^*, \dots, k_{T+1}^*)$$

Next week ...

- Want to show that solution system (4)-(6) is a sequence of functions ...

$$\boxed{\begin{aligned} c_t^* &= g_t^c(k_t) \\ k_{t+1}^* &= g_t^k(k_t) \\ &\text{for } t = 0, 1, \ldots \end{aligned}}$$

- Alternative way to solve this "sequence problem" using a "backward induction" method
  → Dynamic Programming

The Dynamic Programming Approach

- $T < +\infty$

$$V_0(k_0) = \max_{\{c_t, k_{t+1}\}_{t=0}^{T}} \left\{ \sum_{t=0}^{T} \beta^t u(c_t) + S_{T+1} : \begin{array}{l} k_{t+1} = f(k_t) - c_t, \quad k_{T+1} \geq 0 \\ t = 0, 1, \dots, T \end{array} \right\}$$

$$= \max_{\{k_{t+1}\}_{t=0}^{T}} \left\{ \sum_{t=0}^{T} \beta^t u\left[f(k_t) - k_{t+1}\right] \quad \text{s.t. } k_{T+1} \geq 0 \right\}$$

$$= \max_{k_1} \left\{ u\left[f(k_0) - k_1\right] + \beta \max_{\{k_{t+1}\}_{t=1}^{T}} \sum_{t=0}^{T-1} \beta^t u\left[f(k_{t+1}) - k_{t+2}\right] : k_{T+1} \geq 0 \right\}$$

tail value

know

$$V_0(k_0) = \max_{k_1} \left\{ u[f(k_0) - k_1] + \beta \underbrace{V_1(k_1)}_{\text{if known}} : k_{T+1} \geq 0 \right\}$$

dunno

dunno

know

then this: 1D problem

New  : Need to know $V_1(\cdot)$

Continuation value function

So...

$$V_t(k_t) = \max_{k_{t+1}} \Big\{ \overbrace{u\left[f(k_t) - k_{t+1}\right]}^{\text{current payoff}} + \overbrace{\beta V_{t+1}(k_{t+1})}^{\text{continuation total payoffs}} : k_{T+1} \geq 0 \Big\}$$

for $t = 0, 1, \ldots T$

Backward Induction !!!

STEP T

Imagine terminal-period problem: $t = T$

$$V_T(\underset{\substack{\uparrow \\ \text{given}}}{k_T}) = \max_{0 \leq k_{T+1} \leq f(k_T)} \left\{ U\left[f(k_T) - k_{T+1}\right] + \beta \underbrace{V_{T+1}(k_{T+1})}_{\substack{\equiv \\ S_{T+1} \\ = \\ 0}} \right\}$$

Optimum:

$$\boxed{k^*_{T+1} = g_T(k_T) = 0}$$

Evaluate value of k^*_{T+1}

$$V_T(k_T) = u[f(k_T) - k^*_{T+1}]$$

$$+\beta S_{T+1} \rightarrow 0$$

$$\boxed{V_T(k_T) = u[f(k_T)]}$$

Step T-1

$$V_{T-1}(k_{T-1}) = \max_{0 \le k_T \le f(k_{T-1})} \left\{ U[f(k_{T-1}) - k_T] + \beta V_T(k_T) \right\}$$

given

known from stage T

$$= \max_{k_T} \left\{ U[f(k_{T-1}) - k_T] + \beta U[f(k_T)] \right\}$$

Get FOC wrt k_T:

$$\boxed{k_T^* = g_{T-1}(k_{T-1})}$$

$$\boxed{\begin{aligned} V_{T-1}(k_{T-1}) &= u\left[f(k_{T-1}) - g_{T-1}(k_{T-1})\right] \\ &\quad + \beta u\left[f(g_{T-1}(k_{T-1}))\right] \end{aligned}}$$

Step T-2

$V_{T-2}(k_{T-2}) = \ldots\ldots$

Get:

$$\boxed{k_{T-2} = g_{T-2}(k_{T-3})}$$

$$\boxed{V_{T-2}(k_{T-2}) = \ldots\ldots}$$

⋮

Stage 0

$$k_1 = g_0(k_0)$$

$$V_0(k_0) = \dots$$

Solution to this Backward Induction problem:

$$\left\{ g_t(k_t) \right\}_{t=0}^{T}$$

which support

$$\boxed{\left\{ V_t(k_t) \right\}_{t=0}^{T}}$$

- Dynamic Programming

→ Dynamic / Time Consistency.

Exercise Assume $T=2$

$$u(c) = \ln(c)$$

$$f(k) = k^{\alpha},$$

$$\alpha \in (0,1)$$

Find optimal solution to the last DP problem.

Stage $t = T = 2$

$$V_2(k_2) = \max_{k_3} \left\{ \ln\left[k_2^{\alpha} - k_3\right] + \beta \underbrace{V_3(k_3)}_{\equiv S_3} \right\}$$

for maximum on RHS:

$$\boxed{k_3^* = 0 \equiv g_2(k_2)} \quad (1)$$

Sub (1) into objective $\underline{fn}$ on RHS:

$$\boxed{V_2(k_2) = \ln(k_2^{\alpha}) = \alpha \ln k_2} \quad (2)$$

Stage $t=1$

u

$$V_1(k_1) = \max_{k_2} \left\{ \ln(k_1^\alpha) - k_2 + \beta V_2(k_2) \right\}$$

$$= \max_{0 \leq k_2 \leq k_1^\alpha} \left\{ \ln(k_1^\alpha - k_2) + \beta\alpha \ln(k_2) \right\}$$

FOC for max. on RHS:

$$\frac{1}{k_1^\alpha - k_2}(-1) + \frac{\beta\alpha}{k_2} = 0$$

$$\boxed{k_2^* = \frac{\alpha\beta}{1+\alpha\beta} k_1^\alpha \equiv g_1(k_1)} \tag{3}$$

Plug (3) into objective fn:

$$V_1(k_1) = \ln\left[k_1^\alpha - \frac{\alpha\beta}{1+\alpha\beta} k_1^\alpha\right] + \beta\alpha \ln\left(\frac{\alpha\beta}{1+\alpha\beta} k_1^\alpha\right)$$

$$= \ln\left[\frac{1}{1+\alpha\beta} k_1^\alpha\right] + \alpha\beta \ln\left(\frac{\alpha\beta}{1+\alpha\beta} k_1^\alpha\right]$$

$$V_1(k_1) = \ln\left(\frac{1}{1+\alpha\beta}\right) + \alpha \ln k_1$$

$$+ \alpha\beta \ln\left(\frac{\alpha\beta}{1+\alpha\beta}\right)$$

$$+ \alpha\beta(\alpha \ln k_1)$$

$$\boxed{V_1(k_1) = \underbrace{\ln\left(\frac{1}{1+\alpha\beta}\right) + \alpha\beta \ln\left(\frac{\alpha\beta}{1+\alpha\beta}\right)}_{\text{const.}} + \alpha(1+\alpha\beta)\underline{\ln(k_1)}}$$

(4)

Stage $t=0$

$$V_0(k_0) = \max_{k_1} \left\{ \ln(k_0^\alpha - k_1) + \beta V_1(k_1) \right\}$$

$$= \max_{k_1} \left\{ \ln(k_0^\alpha - k_1) + \beta \left[\ln\left(\frac{1}{1+\alpha\beta}\right) + \alpha\beta \ln\left(\frac{\alpha\beta}{1+\alpha\beta}\right) + \alpha(1+\alpha\beta)\ln(k_1) \right] \right\}$$

FOC for RHS max:

$$\frac{1}{k_0^\alpha - k_1} = \frac{\alpha\beta(1+\alpha\beta)}{k_1}$$

$$\Rightarrow \quad k_1 = \alpha\beta(1+\alpha\beta)\left[k_0^\alpha - k_1\right]$$

$$\Rightarrow \quad \boxed{k_1 = \frac{\alpha\beta(1+\alpha\beta)k_0^\alpha}{1+\alpha\beta(1+\alpha\beta)} \equiv g_0(k_0)}$$

Plug optimizer into objective function (inside the $\{\cdot\}$ on RHS of max operator...)

$$\Rightarrow V_0(k_0) = \ldots\ldots \; ?$$

A Backward Induction

Algorithm:

Goal: Find $\left\{ g_t(k_t), \; V_t(k_t) \right\}_{t=0}^{T}$

functions

$[k^1, k^2, \ldots, k^N]$

discrete set

<u>Input:</u>

- State-space, $X \ni k_t$
- <u>Def</u>:
  - u
  - f
  - $\mathcal{B}(V_{t+1}) = V_t$

Pseudocode

1. $k_{T+1} = 0$

   Get: $V_T(\cdot)$

   $g_T(\cdot)$

2. For each $t = T-1, T-2, \ldots 0$:

$$
\begin{tikzcd}
V_t & B(V_{t+1}) \arrow[l] \arrow[dl] \\
g_t &
\end{tikzcd}
$$

Output: $(g_0, g_1, \ldots, g_T)$

$$(V_0, V_1, \dots V_T)$$

Pseudocode for B

discrete, approx.

For k in X:

bracketing/search algorithm

$$V(k) \leftarrow \max_{0 \leq k_{+1} \leq f(k)} \left\{ u\left[f(k) - k_{+1}\right] + \beta V_{+1}(k_{+1}) \right\}$$

interpolated ?

repeat max problem for |X| times.

Implementation:

- Interpolate V_{t+1}:

  Numpy → interp

- "max":

  Scipy → optimize

  → fminbound