Monitoria 1

MODELO DE SEARCH, CRESCIMENTO E ÁRVORE DE LUCAS

Macroeconomia III

EPGE - FGV

9 de outubro de 2017

Modelo de Search no Mercado de Trabalho

McCall 1970

Desemprego?

- Excesso de demanda
- Fricções no mercado de trabalho

Enviroment:

- Em $t = 0$ trabalhador encontra-se desempregado
- Enquanto desempregado recebe oferta de trabalho w, $w \sim^{iid} F(\cdot)$, definida em $[0, \bar{w}]$
- Trabalhador decide se aceita ou não o salário w
  - Se não aceita, recebe transferência b , e procurará nova oferta no próximo período.
  - Se aceita w, tem renda w enquanto se mantém empregado, podendo ser despedido com probabilidade π ao final de cada período

Modelo de Search no Mercado de Trabalho

McCall 1970

- se não aceita w
  - ele procura nova oferta w': w' sorteada segundo $f : [0, \overline{w}] \to \mathbb{R}_{++}$.
  - recebe uma transferência b neste período
- se aceita w
  - ele trabalha e aufere renda w neste período
  - com probabilidade π ele é despedido no período seguinte
  - caso seja despedido, ele começa período seguinte com $w' = 0$
  - caso não seja despedido, ele começa período seguinte com $w' = w$

Modelo de Search no Mercado de Trabalho

Preferências representadas por:

$$U(\{c_t\}_{t=0}^{\infty}) = E\left\{\sum_{t=0}^{\infty} \beta^t u(c_t)\right\}$$

em que

- $\beta \in (0,1)$.
- $u : \mathbb{R}_+ \to \mathbb{R}$.
- $u(0) = 0$.
- $\lim_{c\to 0} u'(c) = \infty$.
- $u'(c) > 0$, $u''(c) < 0 \ \forall c \geq 0$.

Não existe possibilidade de empréstimo ou poupança. Não há *recall*

Modelo de Search no Mercado de Trabalho

Formulação Recursiva

Se o trabalhador aceita w hoje e **segue a política ótima a partir de amanhã**, então aufere

$$u(w) + \beta \left[(1-\pi)v(w) + \pi v(0)\right]$$

Se o trabalhador escolhe procurar nova oferta hoje e **segue a política ótima a partir de amanhã**, então aufere

$$u(b) + \beta E\left[v(w')\right] = u(b) + \beta \int_0^{\overline{w}} v(w') f(w') dw'$$

Modelo de Search no Mercado de Trabalho
Formulação Recursiva

Portanto

$$
\begin{aligned}
v(w) \quad = \quad & \max \left\{ u(w) + \beta \left[(1-\pi) v(w) + \pi v(0) \right], \right. \\
& \left. u(b) + \beta \int_0^{\overline{w}} v(w') f(w') dw' \right\}
\end{aligned}
$$

É possível mostrar que $\exists! v$, contínua e limitada, que satisfaz a equação funcional acima e é solução do problema sequencial.

Modelo de Search no Mercado de Trabalho

Formulação Recursiva

Note que,

$$U = u(b) + \beta \int_0^{\overline{w}} v(w')f(w')dw' = v(0)$$

constante.

Pela continuidade de v, $\exists R$ tal que:

$$u(R) + \beta\left[(1-\pi)v(R) + \pi U\right] = U$$

Assim,

$$v(w) = I(w)\left[u(w) + \beta\left[(1-\pi)v(w) + \pi U\right]\right] + (1 - I(w))\, U$$

onde $I(w) = 1$ se $w \geq R$ e $I(w) = 0$ c.c.

Modelo de Search no Mercado de Trabalho

Nota sobre integração numérica

Integração trapezóide:

Utilizando uma aproximação lagrangeana de $F(x)$ é possível mostrar que

$$\int_a^b F(x)dx \simeq \frac{b-a}{2}(F(a)+F(b)). \tag{1}$$

Quando a e b estiverem próximos o bastante a eq. (1) será uma boa aproximação.

Logo basta fazer uma partição fina o bastante do intervalo $[a, b]$. Por exemplo, podemos dividir $[a, b]$ em n intervalos de mesmo tamanho $h = (b-a)/n$.

$$I_n = \frac{h}{2}\left[F(a)+F(b)+2\sum_{i=1}^{n-1}F(x_i)\right]. \tag{2}$$

Modelo de Search no Mercado de Trabalho

Nota sobre integração numérica

Algoritmo:

1. calcule $h = (b - a)/n$
2. monte um grid: $x_i = a + hi$, $i \in \{0, \ldots, n\}$
3. compute $I_n = \frac{h}{2}\left[F(x_0) + F(x_n) + 2\sum_{i=1}^{n-1} F(x_i)\right]$

Modelo de Search no Mercado de Trabalho

Algoritmo

1. Carregamos todos os parâmetros e funções que o *environment* fornece.
2. Definimos um grid para a variável de estado: w.
3. Criamos chutes iniciais para V e G, respectivamente função valor e função política.
4. Definimos limites de tolerância para nosso código: ε pequeno e *itmax* grande.
5. Calculamos o payoff de não aceita a oferta e guardamos em uma variável, N.
6. Para cada valor do grid de w calculamos o payoff de aceitar a oferta e gradamos em um vetor, A.
7. Para cada valor do grid de w calculamos a nova função valor $TV = max\{N, A\}$ e guardamos a função politica em G.
8. Calculamos $d = |TV - V|$, atualizamos V ($V = TV$).
9. Se $d < \varepsilon$ ou as iterações chegaram a *itmax* paramos o código, caso contrário voltamos ao passo 5.

Modelo de Search no Mercado de Trabalho

Exemplo

Considere o seguinte exemplo:

- $(\beta, \pi, \bar{w}, b) = (0.9, 0.3, 10, 0)$.
- $u(x) = \sqrt{x}$.
- $w \sim U([0, \bar{w}])$.

Modelo Clássico de Crescimento

Problema do Planejador

$$\max_{\{c_t, k_{t+1}\}_{t=0}^{\infty}} \left\{ \sum_{t=0}^{\infty} \beta^t u(c_t) \right\}$$

restrito ao conjunto definido por

$$\begin{aligned} c_t \geq 0, k_{t+1} \geq 0 \quad &, \quad \forall t \geq 0 \\ c_t + k_{t+1} \leq f(k_t) + (1-\delta)k_t \quad &, \quad \forall t \geq 0 \\ k_0 \text{ dado} & \end{aligned}$$

- Suponha que $\delta = 1$

Modelo Clássico de Crescimento

Problema do Planejador

Reescrevendo o problema

$$\max_{k_{t+1}\}_{t=0}^{\infty}} \left\{ \sum_{t=0}^{\infty} \beta^t u(f(k_t) - k_{t+1}) \right\}$$

restrito ao conjunto definido por

$$k_{t+1} \in \Gamma(k_t) = [0, f(k_t)] \quad , \quad \forall t \geq 0$$
$$k_0 \text{ dado}$$

Modelo Clássico de Crescimento

Formulação Recursiva

Reescrevendo o problema

$$v(k) = \max_{k' \in \Gamma(k)} \left\{ u(f(k) - k') + \beta v(k') \right\}$$

em que $\Gamma(k) = [0, f(k)]$.

$k' = g(k)$ função política.

Modelo Clássico de Crescimento

Algoritmo

1. Carregamos todos os parâmetros e funções do *environment* .
2. Definimos um grid para a variável de estado: k, capital.
3. Criamos chutes iniciais para V e Gk e Gc, respectivamente função valor e função políticas. Também definimos TV.
4. Definimos limites de tolerância para nosso código: ε pequeno e *itmax* grande. Declaramos também um erro grande inicial $d = 1$, e iteração $it = 0$.
5. Enquanto $d < \varepsilon$ e $it \leq itmax$.
6. Para cada valor de estado k
   - para cada k' do grid, computamos c e $u(c) + \beta V(k')$.
   - Dentre todos os k', calculamos TV (máximo entre todos). E guardamos a função politica em G.
   - Calculamos $d = |TV - V|$, atualizamos V $(V = TV)$.
   - Se $d < \varepsilon$ ou as iterações chegaram a *itmax* paramos o código, caso contrário voltamos ao passo 5.
7. Uma vez que convirja, achamos $V^*(k)$ tq $TV^* = V^*$. Basta recuperar as respectivas funções políticas de acordo com a posição guardada no loop das iterações.

Modelo Clássico de Crescimento Estocástico

Problema do Planejador

Consideramos agora $f(k_t, z_t) = z_t k_t^\alpha$, em que z_t é um processo estocástico que segue uma cadeia de Markov tal que $P(z_t = \bar{z} | z_{t-1} = \bar{z}) = \xi$ e $P(z_t = \underline{z} | z_{t-1} = \underline{z}) = \zeta$.

$$\max_{\{c_t, k_{t+1}\}_{t=0}^{\infty}} \left\{ \sum_{t=0}^{\infty} \beta^t u(c_t) \right\}$$

restrito ao conjunto definido por

$$\begin{aligned} c_t \geq 0, k_{t+1} \geq 0 \quad &, \quad \forall t \geq 0 \\ c_t + k_{t+1} \leq f(k_t, z_t) + (1-\delta) k_t \quad &, \quad \forall t \geq 0 \\ k_0 \text{ dado} \end{aligned}$$

- Suponha que $\delta = 1$

Modelo Clássico de Crescimento Estocástico

Problema do Planejador

Reescrevendo o problema

$$\max_{k_{t+1}\}_{t=0}^{\infty}} \left\{ \sum_{t=0}^{\infty} \beta^t u(f(k_t, z_t) - k_{t+1}) \right\}$$

restrito ao conjunto definido por

$$k_{t+1} \in \Gamma(k_t, z_t) = [0, f(k_t, z_t)] \quad , \quad \forall t \geq 0$$
$$k_0 > 0 \text{ dado}$$

Modelo Clássico de Crescimento Estocástico
Formulação Recursiva

Reescrevendo o problema

$$v(k,z) = \max_{k' \in \Gamma(k,z)} \left\{ u(f(k) - k') + \beta \sum_{z'} \pi_{zz'} v(k', z') \right\}$$

em que $\Gamma(k,z) = [0, f(k,z)]$.
$k' = g(k)$ função política.

Árvore de Lucas

Enviroment:

- Economia de trocas, número grande de indivíduos, sem heterogeneidade (agente representativo)
- Um único ativo durável
- Possui uma única unidade do ativo (árvore), $s_0 = 1$.
- ativo não sofre depreciação e produz frutos (dividendos) a cada período que evoluem de acordo com um processo estocástico.
- frutos são perecíveis.

Árvore de Lucas

Agentes:

- Preferências sobre plano de consumo $c = \{c_t\}_{t=0}^{\infty}$:

$$U(c) = \mathbb{E}\left[\sum_{t=0}^{\infty} \beta^t u(c_t)\right]$$

com $\beta \in (0,1)$, $u' > 0, u'' < 0$.

- os gastos dos agentes são restritos pela sua riqueza:

$$w_t = (p_t + x_t)s_t$$

que pode ser utilizada para adquirir mais unidades do ativo árvore.

Problema Sequencial:

$$\max_c \mathbb{E} \sum_t \beta^t u(c_t)]$$

$$s.t \qquad c_t + p_t s_{t+1} \leq (p_t + x_t)s_t, \quad \forall t$$

$$c_t, s_{t+1} \geq 0 \qquad s_0, x_0 \quad dados$$

Árvore de Lucas

Formulação Recursiva

Reescrevendo...

$$V(s,x) = \max_{c,s'\geq 0} u(c) + \beta E\left[V(s',x')|x\right]$$

$$s.t \qquad c + p(x)s' \leq [p(x)+x]s$$

onde as variáveis de estado são (s,x).

A solução será dada por uma função política $s' = g(s,x)$.
Condição de market clearing: $g(s,x) = 1$.

Mehra and Prescott - Equity Premium Puzzle
Ambiente

- massa unitária de agentes com desconto β
- uma árvore de Lucas (s)(ativo de risco) e títulos sem risco de um período (B)
- a árvore paga dividendos (y) que crescem a uma taxa x (modificação do enviroment da árvore de Lucas - taxa de crescimento das dotações seguem um processo de Markov).
    - x segue um processo de markov com n estados
    - $\pi(x', x) = P(x_{t+1} = x' | x_t = x)$
    - taxa de crescimento bruta dos dividendos: $x' = \frac{y'}{y}$.

Mehra and Prescott - Equity Premium Puzzle

Equação de Bellman

$$V(w,x,y) = \max_{s' \geq 0, B' \geq 0} u(c) + \beta \sum_{x'} V(w',x',y')\pi(x',x)$$

$$\text{sa} \quad c + p(x,y)s' + q(x,y)B' \leq w$$

(r.o)

$$w' = [p(x',y') + y']s' + B'$$

$$y' = x'y$$

onde as duas últimas equações referem-se as leis de movimento.

Mehra and Prescott - Equity Premium Puzzle

Equilíbrio

Definition

Um equilíbrio competitivo recursivo é $\{V, g_s, g_B, p, q\}$ tais que

- dados p e q, V, g_s, g_B resolvem o problema de programação dinâmica dos agentes.
- Market Clearing

$$
\begin{aligned}
s' &= g_s(w, x, y) = 1 \\
B' &= g_B(w, x, y) = 0
\end{aligned}
$$

Mehra and Prescott - Equity Premium Puzzle

Resolvendo o Modelo

Em equilíbrio $c = y$. Então

$$p(x, y) = \beta \sum_{x'} \frac{u'(x'y)}{u'(y)} [p(x', x'y) + x'y] \pi(x', x)$$

Suponha que $u(c) = c^{1-\sigma}/(1-\sigma)$ e $p(x_i, y) = p_i y$ para todo i, temos

$$p_i = \beta \sum_{j=1}^{n} x_j^{1-\sigma} (p_j + 1) \pi(x_j, x_i)$$

Mehra and Prescott - Equity Premium Puzzle

Resolvendo o Modelo

Ou seja, temos um sistema com n equações para encontrar n preços.

- Definindo a matriz $n \times n$ $\mathbf{A}$ em que
  - $a_{ij} = \beta x_j^{1-\sigma} \pi(x_j, x_i)$
- o vetor $\mathbf{n}$ $n \times 1$ em que
  - $b_i = \sum_{j=1}^{n} x_j^{1-\sigma} \pi(x_j, x_i)$
- podemos redefinir o sistema de equações como

$$\mathbf{p} = \mathbf{A}\mathbf{p} + \mathbf{b} \Rightarrow \tag{3}$$

$$(\mathbf{I} - \mathbf{A})\mathbf{p} = \mathbf{b} \Rightarrow \tag{4}$$

$$\mathbf{p} = (\mathbf{I} - \mathbf{A})^{-1}\mathbf{b} \tag{5}$$

se $(\mathbf{I} - \mathbf{A})^{-1}$ existe.

Mehra and Prescott - Equity Premium Puzzle
Resolvendo o Modelo

para os títulos,

$$q(x,y) = \beta \sum_{x'} \frac{u'(x'y)}{u'(y)} \pi(x', x)$$

$$q(x_i, y) = \beta \sum_{j=1}^{n} x_j^{1-\sigma} \pi(x_j, x_i)$$

Mehra and Prescott - Equity Premium Puzzle

Retornos esperados

para a árvore

$$\hat{r}^e(x_j, x_i) = \frac{p(x_j, x_j y) + x_j y - p(x_i, y)}{p(x_i, y)} \quad (6)$$

$$= \frac{p_j x_j + x_j - p_i}{p_i} \quad (7)$$

Então o retorno esperado condicionais

$$r^e(x_i) = \sum_{j=1}^{n} \hat{r}^e(x_j, x_i)\pi(x_j, x_i)$$

e o retorno incondicionais

$$\bar{r}^e = \sum_{i=1}^{n} r^e(x_i)\bar{\pi}(x_i)$$

$\bar{\pi}$ é a distribuição invariante da matriz de Markov.

Mehra and Prescott - Equity Premium Puzzle
Retornos esperados

para os títulos

$$r^f(x_i) = \frac{1 - q(x_i, y)}{q(x_i, y)} = \frac{1 - q_i,}{q_i}$$

e o retorno incondicionais

$$\bar{r}^f = \sum_{i=1}^{n} r^f(x_i)\bar{\pi}(x_i)$$

Então a média do **prêmio de risco** é

$$\bar{r}^e - \bar{r}^f$$

Mehra and Prescott - Equity Premium Puzzle
Calibração

Mehra and Prescott (1985)

- $n = 2$
- $x_1 = 1 + \mu - \delta$, $x_2 = 1 + \mu + \delta$
- Matriz de transição

$$\begin{pmatrix} \phi & 1-\phi \\ 1-\phi & \phi \end{pmatrix}$$

- μ: média do crescimento do consumo per capita
- δ: o desvio padrão do crescimento do consumo per capita
- ϕ: autocorrelação de primeira ordem do crescimento do consumo per capita $(2\phi - 1)$
- $\beta \in (0, 1)$ e $\sigma \in [0, 10]$

Sistemas Lineares

problema

Queremos resolver sistemas do tipo

$$Ax = b$$

em que

- A é $n \times n$
- x é $n \times 1$
- b é $n \times 1$

Sistemas Lineares

Método de Jacobi

Sistema

$$\begin{aligned}
a_{1,1}x_1 + a_{1,2}x_2 + \ldots + a_{1,n}x_n &= b_1 \\
a_{2,1}x_1 + a_{2,2}x_2 + \ldots + a_{2,n}x_n &= b_2 \\
\vdots \qquad\qquad & \qquad \vdots \\
a_{n,1}x_1 + a_{n,2}x_2 + \ldots + a_{n,n}x_n &= b_n
\end{aligned}$$

Dado x, podemos calcular

$$\begin{aligned}
x_1 = & \quad \tfrac{1}{a_{1,1}}(b_1 - a_{1,2}x_2 - a_{1,3}x_3 - \ldots - a_{1,n}x_n) \\
x_2 = & \quad \tfrac{1}{a_{2,2}}(b_2 - a_{2,1}x_1 - a_{2,3}x_3 - \ldots - a_{2,n}x_n) \\
\vdots \quad & \qquad\qquad\qquad \vdots \\
x_n = & \tfrac{1}{a_{n,n}}(b_n - a_{1,n}x_1 - a_{2,n}x_3 - \ldots - a_{1,n-1}x_{n-1})
\end{aligned}$$

Sistemas Lineares

Método de Jacobi

Assim, temos um algoritmo para computar a solução do sistema

- chute x_0
- compute x de acordo com o sistema anterior
- se $|x_0 - x| < \epsilon$ pare, temos a solução
- caso contrário, faça $x_0 = x$ e volte ao segundo passo

Sistemas Lineares

Eliminação de Gauss

$$A = \begin{pmatrix} a_{1,1} & \dots & a_{1,n} \\ \vdots & \ddots & \vdots \\ a_{n,1} & \dots & a_{n,n} \end{pmatrix}, b = \begin{pmatrix} a_{1,n+1} \\ \vdots \\ a_{n,n+1} \end{pmatrix}$$

defina $E_j = [a_{j,1}, \dots, a_{j,n}, a_{j,n+1}]$.

Tomando o cuidado para que $a_{1,1} \neq 0$ fazemos

$$(E_j - (a_{j,1}/a_{1,1})E_1) \rightarrow (E_j)$$

Teremos novas A e b, em que

$$A = \begin{pmatrix} a_{1,1} & a_{1,2} & \dots & a_{1,n} \\ 0 & a_{2,2} & \dots & a_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & a_{n,2} & \dots & a_{n,n} \end{pmatrix}$$

Sistemas Lineares
Eliminação de Gauss

Podemos repetir um procedimento similar para as outras linhas. Para $i = 2, \dots n - 1$

$$(E_j - (a_{j,i}/a_{i,i})E_i) \to (E_j) \text{ para } j = i + 1, \dots, n$$

desde que $a_{i,i} \neq 0$, de modo que teremos novas A e b, em que A será triangular superior.

Podemos computar $x_n = a_{n,n+1}/a_{n,n}$ e recursivamente

$$x_i = \frac{a_{1,n+1} - \sum_{j=i+1}^{n} a_{i,j} x_j}{a_{i,i}}$$

para $j = n - 1, n - 2, \dots, 1$.