Detalhamento : Algoritmo Modelo OLG

Yuri Passuelo

FEARP/USP

1 Hugget_1996_Main.m

Script Principal que orquestra toda a resolução do modelo ao estilo Hugget, começamos primeiramente escolhendo os parâmetros envolvidos na Otimização, os parâmetros provem do *script* `ParameterValues_Fixed.m`

1.1 ParameterValues_Fixed.m

Abaixo temos uma série de parametrizações envolvidas no modelo, como idade máxima e mínima do modelo de gerações sobrepostas, taxa de crescimento populacional, tabua atuarial de sobrevivência.

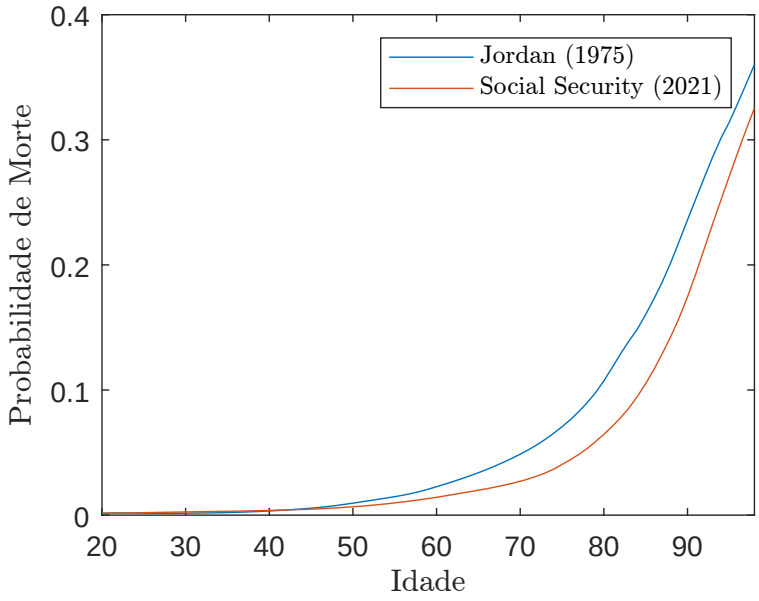
Parâmetro	Valor
Idade Mínima	20
Idade Máxima	98
Aposentadoria	65
η	0.012/0.0/-0.012

A partir dessas primeiras também calculamos a massa de indivíduos em uma determinada idade, calculada a partir de:

$$\mu_t = \frac{\mu_{t-1} s_t}{1-\eta}$$

Parâmetros associados ao *Household*:

Figura 1: Curvas de mortalidade



Parâmetro	Valor
σ	1.5/3
β	1.011
$\underline{c}$	0.05
θ	0.1
N. Simulações	50.000

Parâmetros associados ao função de produção:

Parâmetro	Valor
A	0.895944
α	0.36
δ	0.05

Parâmetros associados ao *Grids*:

Primeiro analisando o processo gerador dos choques, temos que o choque segue um processo AR(1):

$$z_t = \mu + \rho z_{t-1} + \varepsilon$$

$$\varepsilon \sim N(0, \sigma_\varepsilon^2)$$

Assim a parametrização de ρ, σ_ε e a discretização segue:

Parâmetro	Valor
σ_e	0.38
σ_2	0.045
ρ	0.96
N. DPs	4
N. Pontos	7

Por último temos os do grid de capital:

Parâmetro	Valor
K/Y	3
Target Salários	$((1-\alpha)A\frac{(K/Y)}{A})^{\alpha/(1-\alpha)}$
n_k	150
$\underline{k}$	0
$\bar{k}$	300

1.2 EarningProcess_olgm.m

Com base nos itens ρ, σ_ε da parametrização construímos o nosso *grid* de choques z_t que constitui basicamente na aplicação do método de Tauchen (1986) para discretização do processo AR(1) que descrevemos anteriormente, então primeiramente usamos o *script* `tauchen.m` para essa discretização.

Após a discretização aplicamos uma normalização no nosso grid usando o *script* `norm_grid.m` aonde buscamos atribuir probabilidades a cada um dos pontos do nosso grid.

1.3 LaborEndowSimulation_olgm.m

Essencialmente esse Script chama a função `MarkovChainSimulation.m` , ou seja um script que simula cadeias de Markov

1.3.1 MarkovChainSimulation.m

O principal intuito da simulação de uma cadeia de Markov é porque queremos simular as dotações que cada individuo terá a respeito dos choques idiossincráticos na variável de choque z_t em cada período, então com base na discretização construída no processo AR(1) teríamos o a matriz de transição e os pontos discretizados. O algoritmo que simula essa cadeia de markov envolve:

Algorithm 1: Simula Cadeia de Markov

$n \leftarrow 50.000$
$n_s \leftarrow$
$T \leftarrow 79 : (98 - 20)$
P_0 : Probabilidade em $t = 0$
M : Matriz de transição
1. Acumularemos (Soma cumulativa) nas linhas a matriz M:
$CM \leftarrow Cumsum(M)$
2. Faremos a Transposta dessa matriz CM:
$CM_T \leftarrow Transpose(CM)$
3. Criaremos uma matriz de zeros com dimensões n por T:
$SM \leftarrow \text{Zeros}(n, T)$
4. Preencheremos a primeira coluna da matriz
$SM[:, 1] \leftarrow 1 + \text{sum}((*\text{ones}(1, n)) > (\text{ones}(n_s, 1) * \text{cumsum}(P_0'), 2))$
5. *Loop* para preencher o resto:
for $t = 1 : T - 1$ **do**
$SM[:, t + 1] \leftarrow 1 + \text{sum}(, 2)$
end for

A partir disso criamos uma matriz de dimensão $n \times 79$ contendo um número n de simulações para as 79 idades que temos, contendo os possíveis sete choques idiossincráticos.

1.4 LaborSupply_Hugget.m

Dada as simulações realizadas dos choques computaremos a oferta de trabalho efetiva no modelo. O cálculo pega como exemplo simplesmente uma formula:

Primeiro, temos a criação de uma matriz H de dimensão $n \times 79$, com n sendo o número de simulações. A partir disso para cada idade $t \in [20, 65]$ temos

$$H[:,t] = Eff[t] \times \text{Simulation Vector}[:,t]$$

Por fim temos que nossa oferta de trabalho será média de cada coluna da matriz H, depois multiplicaremos cada coluna pela massa de idade correspondente e somaremos

$$HM = \text{Média}(H, \text{Colunas})$$

$$L = \sum_{t \in [20:65]} HM_t \times \text{Massa Idade}_t$$

1.5 HHPrices_Huggett.m

Basicamente dados os parâmetros iniciais, e o que calculamos como a oferta de trabalho, então podemos estimar os preços do modelo que são basicamente:

1. Taxa de juros r
2. Salários w
3. Impostos τ (Nota que aqui os impostos sofrem uma espécie de calibração

$$Y = A * K^{\alpha} L^{1-\alpha}$$

$$r = \frac{\partial Y}{\partial K} = \alpha A (\frac{K}{L})^{\alpha - 1}$$

$$w = \frac{\partial Y}{\partial L} = (1-\alpha) A (\frac{K}{L})^{\alpha}$$

Aqui impostos são calculados como:

$$\tau = \frac{0.195}{1 - \delta \frac{K}{Y}}$$

Benefícios de seguridade social são:

$$b = \theta w R$$

Aonde aqui R representa a massa de aposentados.

Script basicamente é acionado diversas vezes principalmente na estimação do modelo, quando resolvemos o processo de otimização.

1.6 FindEquilibrium.m

Script que basicamente realiza o encontro do equilíbrio do modelo de OLG

Algorithm 2: Resolve problema de OLG

Chutamos K e T
tol $\leftarrow 1e^{-5}$
max_it $\leftarrow 100$
max_dist $\leftarrow 10$
it $\leftarrow 0$
$\lambda \leftarrow 0.98$
while max_dist >= tol $\wedge$ it $\leq$ max_it **do**
    it $\leftarrow$ it $+ 1$
    $r, w, b, \tau \leftarrow$ HHPrices_Hugget(K, L)
     Pega vetores de politica
    $k'_{\text{pol}}, c_{\text{pol}}, V \leftarrow$ HHSolution_VFI_Hugget$(Params)$
     Com base nos resultados simulamos varios cenarios de k':
    $k'_{\text{Simulated}} \leftarrow$ HHSimulation_olgm$(k'_{\text{pol}}, c_{\text{pol}}, , SM)$
    $K_0 \leftarrow \text{mean}(k'_{\text{Simulated}}, 1) * \text{Massa Idade}$
    $K_0 \leftarrow \max(0.01, K_0)$
    $T_0 \leftarrow \frac{k'}{1-\eta}$
    Atualiza K e T
    $K \leftarrow \lambda K + (1 - \lambda)K_0$
    $T \leftarrow \lambda T + (1 - \lambda)T_0$
end while

1.6.1 HHSolution_VFI_Hugget.m

Abordaremos aqui mais sobre como calculamos e resolvemos propriamente o problema de Otimização, a sequência de funções é:

- `HHSolutionByAge_VFI_Huggett.m`
  - `HHIncome_Huggett.m`
  - `HHSolutionByOneState_VFI_Huggett.m`

1.6.2 HHSolutionByAge_VFI_Huggett.m

Aqui iniciamos o procedimento de dados os parâmetros e os preços computados buscamos calcular uma matriz que contém dimensão n_k por n_z, ou seja, quantidade de pontos do grid de capital pela quantidade de pontos do grid de choques, essa função é acionada por meio do *script* `HHIncome_Huggett.m`.

Aqui queremos resolver o problema de otimização que consiste em:

$$V(k,z,t) = \max_{c,k'}\{u(c) + \beta s_{t+1} E_{z'|z}[V(k',z',t+1)]\}$$

Sujeito à:

$$c + k' \leq k(1 + r(1-\tau) - \delta$$

$$c \geq 0$$

$$k' \geq \begin{cases} \underline{k} & t < T \\ 0 & \text{Caso Contrario} \end{cases}$$

Para a resolução desse problema utilizamos uma solução via *Backward induction*, isso pois essencialmente temos uma quantidade finita de estados para serem analisados, no caso que é uma combinação de grid de k, grid de z e um conjunto de idades t_0 até T

Começamos primeiramente com o caso aonte $t = T$, ou seja, da última idade, como os individuos tem vida finita, e estão no ultimo periodo de suas vidas, e esses sabem que este é o ultimo periodo, então vão consumir toda sua renda portanto nesse cenário:

$$c = y = k(1 + (1 - \tau)r - \delta) + T + b$$

$$k' = 0$$

$$V(k, z, T) = u(c) = u(y) = u(k(1 + (1 - \tau)r - \delta) + T + b)$$

Portanto a função politica para essa idade é consumir toda renda e poupança disponível, independente do ponto de k ou que choque z foi recebido.

A partir da idade $T - 1$ voltamos ter um problema de decisão de consumo e poupança, aonde teremos que otimizar:

$$V(k, z, T - 1) = \max_{c,k'}\{u(c) + \beta s_T E_{z'|z}[V(k', z', T)]\}$$

Ou de uma forma mais aberta:

$$V(k, z, T - 1) = \max_{c,k'}\{u(c) + \beta s_T E_{z'|z}[u(k'(1 + (1 - \tau)r - \delta) + T + b)]\}$$

E assim por diante até que cheguemos na idade $t = t_0$.

Seguindo uma estrutura algorítmica:

Algorithm 3: Algoritmo versão *backward induction*

K : Vetor/Grid de Capital

Z : Vetor/Grid de Choques idiossincráticos

P : Matriz de transição dos choques z

V^T_{fun} : Função valor na idade T

c^T_{pol} : Função política consumo na idade T

k^T_{pol} : Função política capital na idade T

y : Matriz de renda

$\underline{c}$: Consumo minimo

for $t \in T-1 : t_0$ **do**

$c^t_{\text{pol}} \leftarrow \text{Zeros}(n_k, n_z)$

$k^t_{\text{pol}} \leftarrow \text{Zeros}(n_k, n_z)$

$V^t_{\text{fun}} \leftarrow \text{Zeros}(n_k, n_z)$

for $z \in Z$ **do**

$E[V^{t+1}_{\text{fun}}] \leftarrow P[:, z]V^{t+1}_{\text{fun}}$

for $k \in K$ **do**

$k'_{\max} \leftarrow \min(\bar{k}, y(k, z, t) - \underline{c})$

if $k'_{\max} \leq \underline{k}$ **then**

$k' \leftarrow \underline{k}$

else

$k' \leftarrow \arg\max_{k'}\{u(y(k,z,t) - k') + \beta s_{t+1} E_{z|z'}[V(k', z', t+1)], u(\underline{k}) + \beta s_{t+1} E_{z|z'}[V(k', z', t+1)]\}$

end if

end for

end for

end for

1.6.3 HHIncome_Huggett.m

Essencialmente é a criação de uma matriz n_k por n_z, com o procedimento relativamente bem simples, para o calculo dessa matriz utilizamos:

- Grid de capital k
- Preços r e w

- Impostos θ e τ
- Depreciação δ
- Benefícios de seguridade social b
- Transferências acidentais T
- Choque idiossincrático de produtividade z

Essencialmente podemos dividir a renda do individuo em duas partes, a primeira é relativa a renda do capital, ou seja, rendimentos que o individuo tem da sua poupança:

$$y_1 = (1 + (1 - \tau)r - \delta)k$$

Também temos a renda do individuos não proveniente do capital, que se resume a renda do trabalho mais as transferências acidentais mais a renda proveniente do beneficio.

$$y_2 = e(z, t)w + T + b_t$$

Aonde $e(z, t)$ corresponde a eficiência do individuo na idade t multiplicada pelo parâmetro de choque idiossincrático z.

$$e(z, t) = e(t)z$$

Sendo portanto a renda final:

$$y = y_1 + y_2$$

Dado n_k pontos do grid de capital e n_z pontos do grid de choques idiossincráticos temos então uma matrix $n_k \times n_z$.

1.6.4 HHSolutionByOneState_VFI_Huggett.m

Detalhando um pouco mais a parte da otimização busca a o *script* da função `HHSolutionByOneState_VFI` que consiste em uma série de etapas para garantir a computação da função politica, como já expressado primeiro precisamos garantir nosso primeiro passo que é a otimização para a idade $t = T$, que é realizada no *script* anterior, após isso realizamos

2 Comparação : Otimizações

Nessa seção compararemo diferentes formas para otimizar nosso problema