Sistemas de equações em diferenças no R: Cournot e Markov

Ivette Luna

03 de junho de 2019

Contents

Modelo dinâmico de Cournot

Sejam n as firmas em um mercado oligopolístico de produto homogêneo, em que as firmas são maximizadoras de lucro, com curvas de demanda e custos lineares tal que a elasticidade preço da demanda e os custos marginais são constantes.

A curva de demanda:

$$p_t = a - b\sum_{i=1}^{n} x_{it}$$

onde x_{it} é a produção da $i-$ésima firma no período t; $a, b > 0$. A curva linear de custos é tal que

$$C_{it} = d + c_i x_{it}$$

em que $d, c_i > 0$.

Uma configuração básica do modelo de Cournot consiste em assumir que as firmas tem como *crença* que a produção das concorrentes no próximo período será igual à observada no período atual. Assim,

$$p^i_{t+1} = a - b(x_{it+1} + \sum_{j\neq i}^{n} x_{jt})$$

Sendo as firmas maximizadoras de lucro

$$\pi^i_{t+1} = p^i_{t+1} x_{it+1} - C_{it+1}$$
$$= a x_{it+1} - b x^2_{it+1} - b x_{it+1} \sum_{j\neq i}^{n} x_{jt} - C_{it+1}$$

O nível de produção que maximiza o lucro em $t+1$ atende às condições de otimalidade.

Ótimo local:

Os candidatos a ótimos são dados pela condição de primeira ordem:

$$\partial \pi^i_{t+1} / \partial x_{it+1} = 0$$

Logo,

$$\Rightarrow a - 2b x_{it+1} - b\sum_{j\neq i}^{n} x_{jt} - c_i = 0$$

Assim, para cada firma:

$$x_{it+1} = -\frac{1}{2}\sum_{j\neq i}^{n} x_{jt} + \frac{a - c_i}{2b}$$

A classirficação de ótimos em máximos ou mínimos locais é feita pela condição de segunda ordem: para que o ponto crítico seja máximo

$$\partial^2 \pi^i_{t+1} / \partial x^2_{it+1} < 0$$

o que é atendido pois

$$\frac{\partial^2 \pi^i_{t+1}}{\partial x^2_{it+1}} = -2b < 0$$

Duopólio

Neste caso $n = 2$ e para $i = 1, 2$:

$$x_{1t+1} = -\frac{1}{2}x_{2t} + \frac{a - c_1}{2b}$$
$$x_{2t+1} = -\frac{1}{2}x_{1t} + \frac{a - c_2}{2b}$$

Trata-se de um sistema não homogêneo de 2×2, com matriz A de coeficientes e de termo independente B

$$A = \begin{bmatrix} 0 & -1/2 \\ -1/2 & 0 \end{bmatrix} \qquad B = \begin{bmatrix} (a - c_1)/2b \\ (a - c_2)/2b \end{bmatrix}$$

Os autovalores de A são

$$\lambda_1 = +1/2 \qquad \lambda_2 = -1/2$$

Portanto, trata-se de um sistema estável, com trajetórias que tendem ao nível de equilíbrio dado por $(I - A)^{-1}B$. E para $n > 2$?

Simulando

Na simulação:

- Definir o número de firmas n e o número de períodos;
- Definir os parâmetros do modelo: a, b, d fixos; $c_i \in (0, 1)$;
- Definir as variáveis necessárias x, A, B;
- Definir as condições iniciais de x e a regra de variação.
- Teste para valores de $n > 3$.

```
library(matlib)
library(ggplot2)
library(tidyr)

n = 2;    # numero de firmas
a = 10  # intercepto da curva de demanda
b = 0.5 # elasticidade da demanda
d = 10    # intercepto da funcao linear custo

c = runif(n, 0, 1)  # custos marginais
tmax = 20           # periodos de simulacao
x = matrix(0, n, tmax) # matriz de tamanho i x t
x[ , 1] = 10*runif(n, 0, 1) # producao inicial das firmas
B = (a-c)/(2*b)     # termos independentes do sistema

A = matrix(-0.5, n, n)
A = A + diag(0.5, n, n)  # matriz de coeficientes do sistema
A
```

```
##      [,1] [,2]
## [1,]  0.0 -0.5
## [2,] -0.5  0.0
```

```r
# autovalores
lambda = eigen(A)$values
lambda
```

```
## [1]  0.5 -0.5
```

Os autovalores para $n = 2$ são os esperados, e por tanto, temos um sistema estável, com oscilações amortecidas e convergindo a um **steady state** dado por

$$x_{it+1} = x_{it} = x_i^*$$

```r
I = diag(1, n, n)

Xeq = inv(I-A)%*%B
Xeq
```

```
##          [,1]
## [1,] 6.007994
## [2,] 6.893859
```

Simulando as trajetórias:

```r
for (t in 2:tmax){

        x[ ,t] = A %*% x[ , t-1] + B
}

x = t(x) # transposta


series = as.data.frame(x)
series$t <- seq(1, tmax, 1)

head(series)
```

```
##         V1         V2 t
## 1 1.157356 0.04306029 1
## 2 9.433393 9.31917774 2
## 3 4.795334 5.18115904 3
## 4 6.864344 7.50018840 4
## 5 5.704829 6.46568373 5
## 6 6.222081 7.04544107 6
```

```r
series_tidy = gather(series, -t, key="serie", value="valor")


plot_series = ggplot(series_tidy, aes(x=t, y = valor, color = serie)) +

  geom_line() +

  theme_light()


# para salvar a image
# ?ggsave
```

```
ggsave("series_duopolio.png",
       plot_series,
       width = 5,
       height=3,
       units="in")

getwd()
```

```
## [1] "D:/iluna 2019/aulas/posgrad/HO-012/slides 2019/aula9-eqs-em-diferencasp6"
```

Cadeias de Markov

Resolvendo o exercício do slide, temos a seguinte matriz de transição:

```
# Markov
# -------------------------

coefs <- c(3, 2, 0,0,0, 1, 5, 5, 0, 0, 0, 1, 8, 7, 5, 0, 0, 4, 16, 32, 0, 0, 0, 11, 100)

n_states = 5
S <- matrix(coefs, n_states)
S
```

```
##      [,1] [,2] [,3] [,4] [,5]
## [1,]    3    1    0    0    0
## [2,]    2    5    1    0    0
## [3,]    0    5    8    4    0
## [4,]    0    0    7   16   11
## [5,]    0    0    5   32  100
```

```
total_2005 <- rowSums(S)
total_2005
```

```
## [1]   4   8  17  34 137
```

```
# construindo a matriz de transicao
A = S/total_2005
A
```

```
##      [,1]      [,2]       [,3]      [,4]      [,5]
## [1,] 0.75 0.2500000 0.00000000 0.0000000 0.0000000
## [2,] 0.25 0.6250000 0.12500000 0.0000000 0.0000000
## [3,] 0.00 0.2941176 0.47058824 0.2352941 0.0000000
## [4,] 0.00 0.0000000 0.20588235 0.4705882 0.3235294
## [5,] 0.00 0.0000000 0.03649635 0.2335766 0.7299270
```

```
rowSums(A) # soma 1
```

```
## [1] 1 1 1 1 1
```

Analisando as raízes do sistema dinâmico

$$X_{t+1} = A \cdot X_t$$

```
# para analisar a dinamica basta olhar para os autovalores
eigen(A)$value # raizes da matriz de transicao
```

```
## [1] 1.0000000 0.9229558 0.6010358 0.3750432 0.1470686
```

temos que todas as raízes são menores o iguais a um em módulo, e que todas asraízes são positivas. Logo, não haverá oscilações, mas a raíz unitária não permite atingir o steady state no longo prazo.

Considerando as probabilidades de transição constantes, podemos estimar a nova estrutura produtiva para os anos de 2010 e 2015:

```
total_2010 = colSums(S)
total_2010
```

```
## [1]   5  11  21  52 111
```

```
total_2015 <- round(A%*%total_2010)
total_2015
```

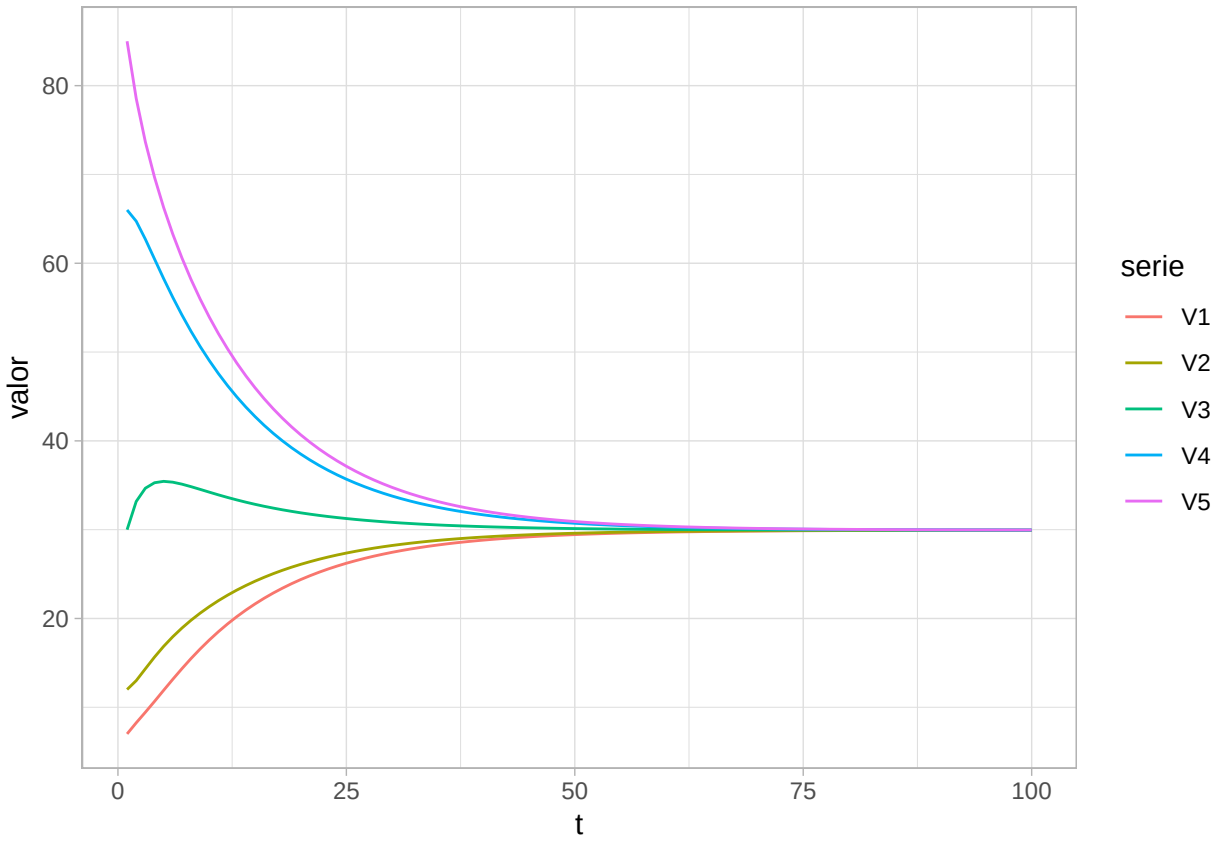
```
##      [,1]
## [1,]    6
## [2,]   11
## [3,]   25
## [4,]   65
## [5,]   94
```

```
total_2020 <- round(A%*%total_2015)
total_2020
```

```
##      [,1]
## [1,]    7
## [2,]   12
## [3,]   30
## [4,]   66
## [5,]   85
```

Simulando para um t longo:

```
tmax = 100

X = matrix(0, n_states, tmax)
X[, 1] = total_2020

for (t in 2:tmax){


  #X[, t] = round(A %*% X[, t-1] )
  X[, t] = A %*% X[, t-1]

}

series = as.data.frame( t(X) )
series$t = seq(1, tmax, 1)

series_tidy = gather(series, -t, key="serie", value="valor")

ggplot(series_tidy, aes(x=t, y = valor, color = serie)) +

  geom_line() +

  theme_light()
```



Sites interessantes

- Markov chains 1
- Markov chains 2