Equilibrio Geral

Ian Teixeira Barreiro

Janeiro 2022

1 Introdução

No modelo da Caixa de Edgeworth, analisamos o equilíbrio geral em uma situação simplificada em que havia uma dotação fixa de cada um dos produtos na economia, e o mercado funcionava unicamente por meio de trocas. Agora, no presente modelo, apresentaremos uma circunstância ainda mais hipotética que permitirá introduzir a produção nos modelos de equilíbrio geral, de modo que agora a quantidade de cada bem na economia não será fixa. A economia tratada será uma economia "Robinson Crusoé", ou seja, uma economia com apenas um indivíduo.

As referências principais para o presente texto são os livros "Microeconomia: Uma Abordagem Moderno, 8a edição"e "Microeconomic Analysis, 3rd Edition"ambos de Hal Varian. O texto se estruturará da presente maneira: i) em primeiro lugar examinaremos a situação hipotética da economia de apenas um indivíduo, tanto quando não há um mercado nessa economia, quanto no cenário em que há, e mostraremos que ambos os cenário levam ao mesmo resultado em equilíbrio; ii) depois descreveremos uma implementação computacional que resolve o modelo.

2 Economia Robinson Crusoé

A economia Robinson Crusoé é uma economia peculiar em que há apenas um indivíduo, que deverá enfrentar o trade-off entre trabalhar para manter sua sobrevivência ou não trabalhar e aproveitar seu tempo de lazer. Podemos analisar a economia apenas pelas decisões de Robinson Crusoé como um indivíduo buscando maximizar sua utilidade dada sua função de produção, ou podemos descrever uma situação extremamente peculiar em que o indivíduo simula uma economia de mercado, em que ele é ao mesmo tempo trabalhador, consumidor e empresário.

2.1 A Economia sem Mercado

Robinson Crusoé está preso em um ilha. Ele poderá decidir usar o seu tempo para coletar cocos, garantindo sua sobrevivência, ou ele poderá usar seu tempo descansando na praia. O náufrago tem uma função utilidade que descreve suas preferências por cocos e lazer. Ao mesmo tempo, possui uma função de produção que descreve, dada a tecnologia disponível, quantos cocos Robinson obtém por unidade de trabalho investida. Essa função determina um conjunto de produção, ou o conjunto de todas as cestas possíveis de lazer e cocos. Qual será a quantidade de cocos e lazer escolhida pelo sujeito? Resolvendo da maneira tradicional, temos que essa será a cesta de consumo mais preferida possível dado o conjunto de produção. Dito de outra maneira, será a cesta dada pelo ponto de tangência entre a função de produção e a curva de indiferença mais alta, o ponto em que a taxa marginal de substituição se iguala ao produto marginal. Assim, como se vê, é possível encontrar a escolha ótima de Robinson Crusoé a partir de uma análise das suas preferências e da sua capacidade de produção enquanto indivíduo.

2.2 A Economia com Mercado

Imaginemos agora um outro cenário. Suponhamos que Robinson Crusoé decida simular um mercado em sua ilha. Ele abre uma empresa produtora de cocos, da qual ele é o único acionista. Além de ser o único acionista de sua empresa, Robinson é também o único trabalhador na economia, e, portanto, o único funcionário possível. Para além disso, por Robinson ser a única pessoa na economia, ele também é o único consumidor.

2.2.1 A Empresa

Como acionista de sua empresa, Robinson deverá observar os preços dos produtos e do trabalho em sua economia para tomar a decisão de quanto produzir e quanto de trabalho contratar que maximize seu lucro. Tomando o preço do único produto na economia como numerário, temos que o problema da firma pode ser expresso da maneira que segue.

$$\max_{X,L_d} X - wL_d$$

Temos que X é a quantidade produzida, w é o salário nominal e L_d é o tempo de trabalho contratado. Assim, assumindo que a função de produção da firma exiba um produto marginal decrescente, como na função $x = L_d^{1/2}$, e tirando a condição de primeira ordem do problema de maximização com respeito a L_d, encontramos o seguinte.

$$
\begin{aligned}
\frac{1}{2}L_d^{\frac{-1}{2}} - w = 0 \\
L_d^{\frac{-1}{2}} = 2w \\
L_d = (2w)^{-2} \\
x = (2w)^{-1} \\
\pi = (2w)^{-1} - w(2w)^{-2} = (4w)^{-1}
\end{aligned}
$$

O lucro π da empresa será entregue como renda ao único acionista da empresa, o próprio Robinson Crusoé, permitindo uma dotação maior no problema do consumidor.

2.2.2 O Consumidor

O consumidor, dada sua dotação de tempo e recursos, a taxa de salário e observando suas preferências, deverá decidir quanto tempo alocar para o trabalho e quanto consumir. Lembremos que o preço do único produto da economia é 1. Vamos assumir também que o consumidor exibe preferências do tipo Cobb-Douglas. O problema do consumidor será como segue.

$$\max_{X,R} X^{\alpha} R^{1-\alpha}$$

Sujeito a:

$$X + wR = \pi + w\bar{L}$$

Temos que R é o tempo despendido em lazer e $\bar{L}$ é o tempo total disponível. Resolvendo o problema do consumidor para encontrar as demandas por lazer e consumo encontramos que a demanda por bens é $X(w) = \alpha(\pi + w\bar{L}) = \alpha(\frac{1}{4w} + w\bar{L})$ e a demanda por lazer é $R(w) = (1-\alpha)(\bar{L} + \frac{1}{4w^2})$. Assim, a oferta de trabalho será $L_s(w) = \bar{L} - (1-\alpha)(\bar{L} + \frac{1}{4w^2})$. O equilíbrio será no ponto em que a oferta de trabalho se iguala à demanda por trabalho.

$$
\begin{aligned}
L_d = L_s \\
\frac{1}{4w^2} = \bar{L} - (1-\alpha)(\bar{L} + \frac{1}{4w^2}) \\
\bar{L}(1 - 1 - \alpha) = \frac{2-\alpha}{4w^2} \\
w = (\frac{2-\alpha}{4L\alpha})^{\frac{1}{2}}
\end{aligned}
$$

Assim, encontramos analiticamente o salário de equilíbrio.

2.2.3 Exemplo Gráfico

Tomemos um exemplo em que o expoente da Cobb-Douglas é 0.7 e $\bar{L}$ é 1. Utilizando o raciocínio exposto acima encontramos o seguinte equilíbrio.

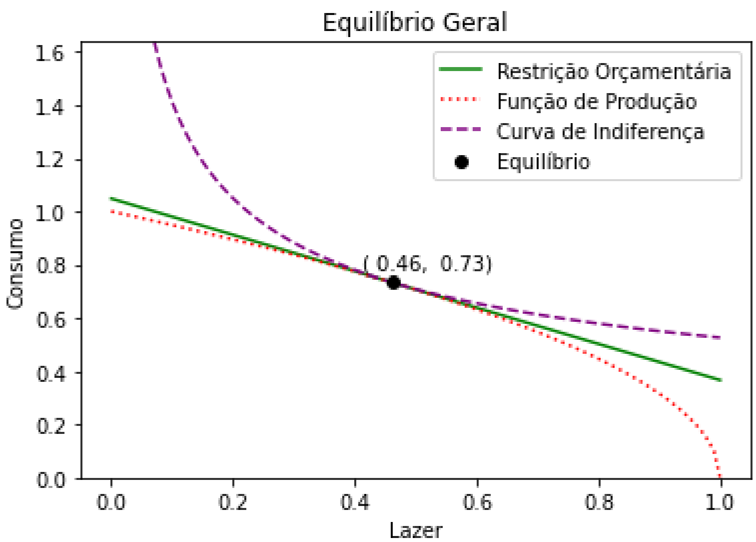

Figura 1: Equilíbrio de uma economia Robinson Crusoé

3 Implementação Computacional

```
import numpy as np
import matplotlib.pyplot as plt

########################################
# Consumidor ----

# Preferencias
cobb_douglas = lambda x, laz, exp: x ** exp * laz ** (1 - exp)

# Restricao Orcamentaria
res_orc = lambda sal, lucro, laz, tempo: lucro + sal * (tempo - laz
    )

# Demanda por produtos
dem_por_prod = lambda exp, sal, tempo: exp * (1 / (4 * sal) + sal *
     tempo)

# Demanda por lazer
dem_laz = lambda exp, sal, tempo: (1 - exp) * (tempo + (1 / (4 *
    sal ** 2)))

# Oferta de trabalho
of_de_trab = lambda exp, sal, tempo: tempo - dem_laz(exp, sal,
    tempo)

########################################
# Firma ----

```

```python
# Funcao de producao
f_de_prod = lambda trab: np.sqrt(trab)

# Demanda por trabalho
dem_por_trab = lambda sal: 1 / (4 * sal ** 2)

# Oferta de produtos
of_de_prod = lambda sal: 1 / (2 * sal)

##########################################
# Equilibrio ----

excesso_de_dem = lambda exp, sal, tempo: ((2 - exp) / (4 * sal ** 2)) - exp * tempo

sal_med = lambda sal_s, sal_e: (np.sqrt(2) * sal_s * sal_e) /\
 np.sqrt(sal_s ** 2 + sal_e ** 2)

# Implementacao analitica do equilibrio
def eq_analitico(exp, tempo, tol = 5):

    # Formula para o salario de equilibrio encontrada
    # analiticamente
    sal_eq = np.sqrt((2 - exp) / (4 * tempo * exp))

    # Aplicando o salario de equilibrio nas demais funcoes
    of_trab = of_de_trab(exp, sal_eq, tempo)
    dem_trab = dem_por_trab(sal_eq)
    of_prod = of_de_prod(sal_eq)
    dem_prod = dem_por_prod(exp, sal_eq, tempo)
    laz = dem_laz(exp, sal_eq, tempo)

    # Garantindo que estamos no equilibrio
    assert round(of_trab, tol) == round(dem_trab, tol)
    assert round(of_prod, tol) == round(dem_prod, tol)

    # Lucro
    lucro = f_de_prod(of_trab) - sal_eq * of_trab

    return sal_eq, of_trab, of_prod, laz, lucro

# Funcao numerica para o equilibrio usando bissecao
def equilibrio(exp, tempo, sal_s, sal_e, tol = 5):

    # Garantindo que os valores de entrada
    # fazem sentido
    assert sal_s > 0 and sal_e > 0 and tempo > 0
    assert sal_s < sal_e

    # Encontrando os primeiros pontos da funcao
    # excesso de demanda
    fs = excesso_de_dem(exp, sal_s, tempo)
    fe = excesso_de_dem(exp, sal_e, tempo)

    # Primeira media
    med = (fs + fe) / 2

```

```
    # Enquanto o valor absoluto da media
    # arredondado para um erro for diferente de 0
    while round(abs(med), tol) != 0:

        # Se a media for positiva
        if med > 0:

            # Valor de inicio do salario sera o
            # valor do salario na media
            sal_s = sal_med(sal_s, sal_e)

        # Se a media for negativa
        elif med < 0:

            # Valor final do salario sera o valor
            # do salario na media
            sal_e = sal_med(sal_s, sal_e)

        # Atualiza valores
        fs = excesso_de_dem(exp, sal_s, tempo)
        fe = excesso_de_dem(exp, sal_e, tempo)

        med = (fs + fe) / 2

    # Aplicando as funcoes
    sal_eq = sal_med(sal_s, sal_e)
    of_trab = of_de_trab(exp, sal_eq, tempo)
    dem_trab = dem_por_trab(sal_eq)
    of_prod = of_de_prod(sal_eq)
    dem_prod = dem_por_prod(exp, sal_eq, tempo)
    laz = dem_laz(exp, sal_eq, tempo)

    # Garantindo que estamos no equilibrio
    # dada a tolerancia
    assert round(of_trab, tol) == round(dem_trab, tol)
    assert round(of_prod, tol) == round(dem_prod, tol)

    # Lucro
    lucro = f_de_prod(of_trab) - sal_eq * of_trab

    return sal_eq, of_trab, of_prod, laz, lucro
```

3.1 Gráficos

```



def cobb_douglas_inv(U, x2, exp):

    return (U / (x2 ** (1 - exp))) ** (1 / exp)

def graficos(exp, tempo, laz_s, laz_e, laz_t):

    # Criando vetores e matrizes
    laz_vec = np.linspace(laz_s, laz_e, laz_t)
    prod_vec = laz_vec.copy()
```

```
    X, Y = np.meshgrid(laz_vec, prod_vec)

    # Encontrando o equilibrio
    sal, trab, prod, laz, lucro = eq_analitico(exp, tempo)

    # Encontrando utilidade no equilibrio
    util = cobb_douglas(dem_por_prod(exp, sal, tempo),
                        dem_laz(exp, sal, tempo),
                        exp)

    # Gerando curvas de indiferenca
    U = cobb_douglas(X, Y, exp)

    # Plotando a restricao orcamentaria
    plt.plot(laz_vec, res_orc(sal, lucro, laz_vec, tempo),
             c = 'green',
             label = 'Restri  o Or ament ria')

    # Plotando a funcao de producao
    plt.plot(laz_vec, f_de_prod(tempo - laz_vec),
             c = 'red',
             label = 'Fun  o de Produ  o',
             linestyle = 'dotted')

    # Plotando curvas de indiferenca
    plt.plot(laz_vec, cobb_douglas_inv(util, laz_vec, exp),
             c = 'purple',
             label = 'Curva de Indiferen a',
             linestyle = 'dashed')

    # Plotando o equilibrio
    plt.plot(laz, prod, 'go',
             c = 'black',
             label = 'Equil brio')

    plt.xlabel('Lazer')
    plt.ylabel('Consumo')
    plt.title('Equil brio Geral')
    plt.ylim([0, util + 1])
    plt.text(laz - 0.05, prod + 0.05, '({laz: .2f}, {prod: .2f})'.format(laz = laz, prod = prod))
    plt.legend(loc = 'best')
    plt.show()
```