PYTHON APLICADO AO ENSINO DE ECONOMIA NA GRADUAÇÃO

DELGADO, D. M.[1]*
[1] *Faculdade de Tecnologia "Mário Robertson de Sylos Filho" – Fatec de Mococa*
**darlan.delgado@fatec.sp.gov.br*

Eixo(s) Tecnológico(s): Desenvolvimento Educacional e Social.

Resumo

A pesquisa aborda a aplicação de *softwares* na área de Economia na Educação Profissional. As competências técnicas relacionadas ao uso de *softwares* em situações de trabalho tendem a se tornar cruciais diante do cenário de intensificação da digitalização da economia. Soma-se a isso a globalização, o progresso tecnológico acelerado e as inovações em tecnologia da informação. Esses elementos são desafios aos gestores educacionais para ofertar uma educação profissional orientada à formação integral dos alunos. O objetivo deste artigo é apresentar resultados sobre o uso da linguagem de programação Python no ensino de Economia na graduação. Metodologicamente são apresentados exemplos de códigos para o cálculo de equilíbrio de mercado entre demanda e oferta. Foram elaborados quatro modos de solução matemática para o referido cálculo, demonstrando a flexibilidade de modelagem computacional em Python. Verificou-se que Python tem se destacado no ensino universitário, permite grande flexibilidade na manipulação de variáveis e possibilita aplicações diversificadas em computação científica.

Palavras-chave: *Educação Profissional, Ensino de Economia, Competências técnicas, Python*.

Abstract

The research addresses the application of software in the subjects of Economics in Vocational Education and Training. Technical competencies related to the mastery and use of software in work situations tend to become decisive in the face of technology intensified digitalization of the economy. Globalization, accelerated technological progress and innovations in information technology are elements to be considered in this context. These issues are challenges to educational managers to offer a vocational education and training oriented to the integral formation of the students. The purpose of this paper is to present results on the use of Python programming language in Economics undergraduate teaching. Methodologically, examples of coding are presented to solve the market equilibrium between demand and supply. Four mathematical solution modes were elaborated for that calculation, thus demonstrating the flexibility of computational modeling in Python. It was found that Python has stood out in university education, allows great flexibility in the manipulation of variables and enables diversified applications in scientific computing.

Key-words: *Vocational Education and Training, Economics teaching, Hard skills, Python*.

1. Introdução

A relação existente entre ciência, tecnologia e economia é um tema caro a distintos campos do conhecimento há longa data, incluindo as ciências da Educação, dada a articulação entre educação e trabalho. À Educação Profissional tem sido atribuída relevância estratégica no plano internacional por organismos como a Organização das Nações Unidas para a Educação, a Ciência e a Cultura (Unesco) como política pública na formação de pessoas qualificadas para o trabalho no cenário contemporâneo, pautado por exigências crescentes em termos de competências técnicas (*hard skills)* e socioemocionais (*soft skills*), em particular quando se trata de educação financeira (DELGADO, 2022)[1].

Em 2020 o Fórum Econômico Mundial (*World Economic Forum – WEF*) publicou o relatório *The Future of Jobs 2020* (WEF, 2020)[2], no qual são apresentados resultados de pesquisa sobre as mudanças na economia mundial durante a pandemia da COVID-19. Foi observado que a crise provocada pela pandemia acelerou processos de mudanças nos negócios, nas estruturas de mercado, na logística e na cadeia mundial de suprimentos, além de outros

aspectos sobre aplicações tecnológicas e inovações, o que impactou nas competências esperadas dos profissionais. O Fórum Econômico Mundial indicou, por meio de *rankings*, as competências demandadas mais significativas nesses novos cenários gerados pela pandemia. Entre outras, as habilidades tecnológicas (*technology skills*) e a fluência tecnológica (*technology fluency*) e alfabetização digital (*digital literacy*) marcaram presença.

O domínio de conhecer e ter capacidades de acessar, manipular, analisar e interpretar dados econômicos, contábeis, financeiros e de outras variáveis relevantes para ações concretas em situações de trabalho e tomada de decisão nos âmbitos pessoal e laboral se apresentam como imprescindíveis aos egressos da Educação Profissional. Como atestam Shi e Chen (2020)[3], não é exagero dizer que a capacidade de programar é uma competência necessária atualmente.

Pugh (2014)[4] afirma que atualmente a modelagem computacional é o "terceiro pilar" da pesquisa científica, ao lado da teoria e da experimentação. Aruoba e Fernández-Villaverde (2014)[5] asseguram que a computação se tornou um instrumento central na ciência econômica e Romero-Aguillar (2018)[6] argumenta que o aprendizado e uso de linguagens de programação têm se apresentado aceleradamente como tarefas dos economistas.

Existe vasta gama de linguagens de programação usadas em economia e em finanças, como Fortran, C, C++, MATLAB, Stata, SAS, Julia, entre outras. No entanto, recentemente a linguagem Python vem ganhando destaque, como assinalam Kuroki (2021)[7] e Shi e Chen (2020)[3].

O objetivo deste artigo é apresentar resultados sobre estudos e experimentações das aplicações de Python no ensino de graduação em Economia com vistas a explicitar suas potencialidades pedagógicas e de solução de problemas matemáticos em Economia.

2. Materiais e métodos

2.1. Materiais

Em termos de recursos de *software*, os códigos foram escritos e rodados no *Jupyter Notebook* versão 6.4.8, presente no ambiente Anaconda 3 e rodando em Python na versão 3.9.

2.2. Metodologia

Os exemplos reportados neste artigo são aplicações que se tornaram possíveis a partir de extensa pesquisa bibliográfica, referenciada ao longo do texto, e da realização do curso "*6.00.1x: Introduction do Computer Science and Programming Using Python*", ofertado na plataforma *edX* pelo Massachusetts Institute of Technology, ao longo do ano de 2022. Destaques precisam ser dados para obras de referência, como *Introduction to Computation and Programming Using Python* (GUTTAG, 2017)[8], o livro-texto do referido curso, *Think Python: how to think like a computer scientist* (DOWNEY, 2015)[9], *Python for Economists* (BELL, 2016)[10], outro material com o mesmo título *Python for Economists* (GUST-BARDON, 2019)[11] e *Matemática com Python: um guia prático* (MARCONDES, 2018)[12]. Adicionalmente, as documentações do Python e das bibliotecas utilizadas são fundamentais para consulta e aplicações metodologicamente corretas.

3. Resultados e Discussão

Nesta seção são apresentados exemplos com o intuito de demonstrar algumas características da linguagem de programação Python quando empregada especificamente para a realização de

procedimentos de cálculos científicos sobre tópicos de Economia. Toma-se, para esta finalidade, o cálculo do preço e da quantidade de equilíbrio entre demandantes e ofertantes por meio de funções matemáticas lineares (funções de primeiro grau), em conformidade como este tópico é apresentado em livros-texto de Microeconomia.

Conforme Pindyck e Rubinfeld (2005)[13], as curvas de demanda e de oferta são empregadas para descrever o mecanismo de mercado. Segundo os autores, "[o] modelo básico da oferta e da demanda é o instrumento-chave da microeconomia" (PINDYCK e RUBINFELD, 2005, p. 18)[13]. De acordo com Mankiw (2013)[14], mercado pode ser entendido, de modo amplo, como um grupo de compradores e vendedores de determinado bem ou serviço. Desse modo, de um lado há os compradores, que determinam a demanda, e de outro há os vendedores, que determinam a oferta do bem ou produto.

O modelo teórico que subjaz ao arcabouço de cálculos simplificados a representar o equilíbrio estático de mercado entre demandantes e ofertantes, gerando como resultado um ponto de equilíbrio identificado pela intersecção entre demanda e oferta e que gera um preço de equilíbrio (P_e) e uma quantidade de equilíbrio (Q_e), é o chamado modelo da concorrência perfeita. Esta estrutura de mercado é calcada sobre certos preceitos como: a) existência de inúmeros ofertantes competindo entre si pela captura da demanda, assim como inúmero compradores, b) nenhum dos agentes econômicos (vendedores ou compradores) têm capacidade de, individualmente, influenciar o preço de mercado, daí o o princípio de que os ofertantes são tomadores de preço de mercado; c) não há barreiras à entrada e à saída do mercado; d) os produtos são homogêneos, implicando que são substitutos entre si; e) há informação perfeita (MANKIW, 2013[14]; PINDYCK e RUBINFELD, 2005[13]).

Apesar de a demanda e a oferta apresentarem outros determinantes além do preço do próprio bem econômico em si, por conveniência e simplificação didáticas é de praxe analisarem-se os efeitos que apenas as variações do preço do bem provocam nas curvas de demanda e de oferta, bem como no equilíbrio, evocando-se para este fim a condição *coetris paribus* (expressão latina que significa "mantendo tudo mais constante").

A função linear de demanda de mercado em função exclusivamente do preço pode ser representada matematicamente como:

$$Q_d = a - b.P \tag{1}$$

De modo análogo, a função linear de oferta de mercado em função exclusivamente do preço:

$$Q_s = c + d.P \tag{2}$$

Nas quais, Q_d é a quantidade demandada, Q_s é a quantidade ofertada (a letra *s* vem do inglês *supply*), *a*, *b*, *c* e *d* são constantes e *P* é o preço, a variável explicativa.

A condição de existência de equilíbrio reside no fato de que demanda e oferta devem se igualar neste ponto, implicando assim que $Q_d = Q_s$, portanto, igual à quantidade de equilíbrio (Q_e) e que o preço praticado seja aceito por vendedores e compradores, o chamado preço de equilíbrio (P_e). Desse modo, por meio das devidas operações algébricas é possível indicar que a quantidade de equilíbrio (Q_e) será dada por:

$$Q_e = \frac{a.d+b.c}{b+d} \tag{3}$$

De modo semelhante, é possível indicar que o preço de equilíbrio (P_e) será dado por:

$$P_e = \frac{a-c}{b+d} \quad (4)$$

Para apresentar as possibilidades de aplicação da linguagem de programação Python nos cálculos do preço de equilíbrio (P_e) e da quantidade de equilíbrio (Q_e) toma-se o exemplo 2.5, do mercado de trigo, do livro *Microeconomia* (PINDYCK e RUBINFELD, 2005, p. 31)[13]:

> A partir de levantamentos estatísticos, temos conhecimento de que, em 1981, a curva de oferta de trigo poderia ser aproximadamente expressa da seguinte maneira: Oferta: $Q_S = 1.800 + 240P$ onde o preço está expresso em dólares por bushel e as quantidades estão expressas em milhões de bushels por ano. Esses levantamentos indicam também que, em 1981, a curva da demanda de trigo era: Demanda: $Q_D = 3.550 - 266P$.

Dada a função de demanda, tem-se que a constante *a* é igual a 3.550 e a constante *b* é 266. De modo semelhante, tendo-se a função de oferta, verifica-se que a constante *c é* igual a 1.800 e a constante *d* é 240. Com esses valores é possível calcular a quantidade de equilíbrio (Q_e) por meio da substituição dos dados na equação (3) acima:

$$Q_e = \frac{a.d+b.c}{b+d} = \frac{(3.550\ x\ 240) + (266\ x\ 1.800)}{266 + 240} = 2.630\ milhões\ de\ bushels \quad (5)$$

E, pela aplicação da equação (4), pode-se calcular o preço de equilíbrio (P_e):

$$P_e = \frac{a-c}{b+d} = \frac{3.550-1.800}{266+240} = \$3{,}46\ por\ bushel \quad (6)$$

Algumas observações não necessárias. O tradutor da obra de Pindyck e Rubinfeld (2005)[13] indica que *bushel* é um padrão de medida para cereais, sendo um *bushel* equivalente a aproximadamente 27,21 Kg de trigo. Preço e quantidade de equilíbrio foram arredondados.

Em Python, há diversas possibilidades de proceder os cálculos. O primeiro método a ser apresentado é justamente o que se utiliza das equações (3) e (4) e que não necessita da importação de bibliotecas. A Figura 1 mostra o código elaborado no *Jupyter Notebook*.

```
In [1]: # Insere o coeficiente "a" da demanda:
        a = 3550
        # Insere o coeficiente "b" da demanda:
        b = 266
        # Insere o coeficiente "c" da oferta:
        c = 1800
        # Insere o coeficiente "d" da oferta:
        d = 240

        # Cálculo da Quantidade de quilíbrio (Qe):
        qe = (a*d + b*c)/(b + d)
        print(f'A quantidade de equilíbrio é: {qe:,.0f}')

        # Cálculo do Preço de equilíbrio (Pe):
        pe = (a - c) / (d + b)
        print(f'O preço de equilíbrio é: ${pe:,.2f}')

A quantidade de equilíbrio é: 2,630
O preço de equilíbrio é: $3.46
```

Fig. 1. Código para cálculos de *Qe* e *Pe* de modo algébrico.

Sempre que for necessário repetir os cálculos para outros exercícios basta informar os valores das constantes. Também seria possível fazer uma simples modificação no código, solicitando ao usuário que insira tais valores por meio da função *input*.

A segunda forma de encontrar a solução para o equilíbrio é por meio de álgebra matricial, considerando-se um sistema de duas equações e uma variável (o preço, *P*). Nesse caso, é necessário importar a biblioteca *NumPy* e desta as funções *matrix* e *linalg*. Torna-se relevante evidenciar ao leitor o fundamento matemático da conversão de um sistema de equações lineares para uma equação matricial, conforme método apresentado por Marcondes (2018)[12].

Um sistema de equações como:

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n = b_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n = b_2 \\ \cdots \\ a_{n1}x_1 + a_{n2}x_2 + \cdots + a_{nn}x_n = b_n \end{cases} \quad (7)$$

Pode ser reescrito como $A.X = B$, sendo:

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \cdots & \cdots & \cdots & \cdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix} \quad X = \begin{bmatrix} x_1 \\ x_2 \\ \cdots \\ x_n \end{bmatrix} \quad B = \begin{bmatrix} b_1 \\ b_2 \\ \cdots \\ b_n \end{bmatrix} \quad (8)$$

Sendo assim, a solução do sistema por álgebra matricial será dada por:

$$X = A^{-1}.B \quad (9)$$

Na qual A^{-1} representa a matriz inversa de *A*.

Para o exemplo do mercado de trigo aqui analisado a partir da obra de Pindyck e Rubinfeld (2005)[13], com o necessário rearranjo das equações para assumirem a forma para a manipulação de álgebra matricial (as quantidades e preços foram consideradas de equilíbrio, que devem ser idênticos), tem-se:

$$\begin{cases} Q_e + 266.P_e = 3.550 \\ Q_e - 240.P_e = 1.800 \end{cases} \quad (10)$$

As matrizes, então, são:

$$A = \begin{bmatrix} 1 & 266 \\ 1 & -240 \end{bmatrix} \quad B = \begin{bmatrix} 3.550 \\ 1.800 \end{bmatrix} \quad X = \begin{bmatrix} Q_e \\ P_e \end{bmatrix} \quad (11)$$

O código de programação para esta operação em Python pode ser visto na Figura 2.

Verifica-se que a saída de resultado (*matrix_X*) é um objeto "matriz" da biblioteca *NumPy*. O acesso aos elementos da matriz se dá pela posição de cada valor na referida matriz, sendo o primeiro elemento (na posição ou *index* 0) a quantidade de equilíbrio (Q_e) e o segundo elemento (na posição ou *index* 1) o preço de equilíbrio (P_e), os quais foram convertidos em números flutuantes (não inteiros) nomeados de *float* em Python.

```
In [3]: # Considere as funções de demanda e de oferta:
        #Qd = 3550 - 266P
        #Qs = 1800 + 240P

        # Rearranje as funções para elboarar as matrizes A e B:
        # Qd + 266P = 3550
        # Qs - 240P = 1800

        import numpy as np
        from numpy import matrix, linalg
        matrix_A = matrix([[1, 266], [1, -240]])
        matrix_B = matrix([[3550], [1800]])

        matrix_A_inv = linalg.inv(matrix_A)
        matrix_X = matrix_A_inv * matrix_B
        print(matrix_X)

        [[2630.03952569]
         [   3.45849802]]

In [4]: type(matrix_X)

Out[4]: numpy.matrix

In [5]: quantE = float(matrix_X[0])
        print(f'A quantidade de equilíbrio é: {quantE:,.0f}')
        pE = float(matrix_X[1])
        print(f'O preço de equilíbrio é: ${pE:,.2f}')

        A quantidade de equilíbrio é: 2,630
        O preço de equilíbrio é: $3.46
```

Fig. 2 - Código para cálculos de *Qe* e *Pe* por meio de álgebra matricial.

A terceira alternativa possível de cálculo em Python é o emprego de computação simbólica, ou seja, trabalhar com os objetos matemáticos simbolicamente, no caso em foco, com as funções de demanda e de oferta assumindo o preço (P) como variável simbólica (a variável explicativa). Para assim proceder é necessário importar a biblioteca de computação simbólica para Python chamada *SymPy*.

Como discutido anteriormente, a condição de equilíbrio implica em que $Q_d = Q_s$, portanto, igual à quantidade de equilíbrio (Q_e). Desse modo, pode-se afirmar que $Q_d - Q_s = 0$ no ponto de equilíbrio. É sobre esta constatação matemática que se pode empregar a biblioteca *SymPy* e sua função *solve* para resolver o cálculo e encontrar o valor do preço de equilíbrio. Como o valor resultante (P = 875/253) está contido em um objeto do tipo *lista*, é possível acessá-lo por meio de sua posição ou *index* (igual a 0, no código: *resultado*[0]). Como esse valor dentro da lista é um objeto do tipo *sympy.core.number.Rational* ele pode ser convertido a um número de ponto flutuante (não inteiro) para ser apresentado como P = 3,46. Isso é feito ao se atribuir a uma variável nomeada *pEquilibrio* a função *float* sobre o conteúdo do *index* 0 da lista "resultado".

A quantidade de equilíbrio (Q_e) pode ser calcula ao aplicar a função *subs* (*Substitution*) na função de demanda, indicando a troca da variável simbólica P pelo valor do preço de equilíbrio armazenado na variável *pEquilibrio*. Os detalhes do código podem ser observados na Figura 3.

```
In [5]: # Importando a biblioteca SymPy
        import sympy as sp

        # Estabelecendo as funções e definindo a condição de equilíbrio
        P = sp.symbols('P')
        Qd = 3550 - 266*P
        Qs = 1800 + 240*P
        equilibrio = Qd-Qs
        resultado = sp.solve(equilibrio)

        # Imprimindo o preço de equilíbrio (Pe) contido na lista "resultado"
        pEquilibrio = float(resultado[0])
        print(f'O preço de equilíbrio é: ${pEquilibrio:,.2f}')

        # Imprimindo a quantidade de equilíbrio (Qe) ao substituir Pe em qualquer das funções
        qEquilibrio = Qd.subs(P, pEquilibrio)
        print(f'A quantidade de equilíbrio é: {qEquilibrio:,.0f}')

O preço de equilíbrio é: $3.46
A quantidade de equilíbrio é: 2,630
```

Fig. 3 - Código para cálculos de *Qe* e *Pe* por meio de computação simbólica.

A quarta alternativa factível para os cálculos de equilíbrio em Python reside na elaboração de uma função personalizada e que utiliza a biblioteca *SymPy* em sua construção. Neste caso, utilizam-se as constantes *a* e *b* da função de demanda e *c* e *d* da função de oferta como valores de entrada da função. Em seu interior são empregadas as funções *equation* (*Eq*) e *solve* da biblioteca *SymPy*. Pode-se verificar, na Figura 4, que é um código enxuto, permitindo exibir na tela as funções de demanda e de oferta, bem como os valores do preço e da quantidade no equilíbrio. O código está na Figura 4.

```
In [1]: def equilibrio(a, b, c, d):
            import sympy as sp
            P, Q = sp.symbols('P Q')
            demanda = sp.Eq(a - b*P - Q, 0)
            oferta = sp.Eq(c + d*P - Q, 0)
            resultado = sp.solve((demanda, oferta), (P, Q))
            pe = float(resultado[P])
            qe = float(resultado[Q])
            print(f'A função de demanda é Qd = {a} - {b}.P')
            print(f'A função de oferta é Qs = {c} + {d}.P')
            print(f'O Preço de equilíbrio (Pe) é ${pe:,.2f}')
            print(f'A Quantidade de equilíbrio (Qe) é {qe:,.0f}')

        equilibrio(3550, 266, 1800, 240)

A função de demanda é Qd = 3550 - 266.P
A função de oferta é Qs = 1800 + 240.P
O Preço de equilíbrio (Pe) é $3.46
A Quantidade de equilíbrio (Qe) é 2,630
```

Fig. 4 - Código de uma função que realiza os cálculos de equilíbrio e mostras as funções.

Como se pode verificar pelos quatro exemplos de códigos apresentados – e que não esgotam as possibilidades de abordagem – a linguagem Python possibilita grande flexibilidade e maneabilidade em computação científica.

4. Conclusões

A necessidade de fluência tecnológica vem se tornando uma competência crucial para inúmeras ocupações e profissionais de diversas áreas. No campo da gestão empresarial, os conhecimentos em Economia e disciplinas correlatas passam a ser aplicados por meio do emprego de recursos tecnológicos mais sofisticados, como Python e suas bibliotecas.

Foi objetivo neste artigo demonstrar, por meio de exemplos de códigos, como o simples cálculo de equilíbrio entre duas funções lineares de demanda e oferta permite ao usuário aplicar conhecimentos matemáticos em distintas formas de abordagem e modelagem computacional

por meio da linguagem de programação, expandindo possibilidades para além dos limites de planilhas de cálculo.

Python é a linguagem de programação que mais tem se destacado tanto no ensino universitário quanto na sua adoção por empresas. Sua sintaxe simples, objetiva e elegante tem atraído um número crescente de usuários. Isso fica particularmente claro na literatura internacional sobre o ensino de Economia e na área de finanças. No contexto da teoria microeconômica é possível realizar toda a abordagem matemática inerente com uso de Python.

Referências

[1] DELGADO, D. M. Educação financeira no contexto de internacionalização da educação profissional e tecnológica e globalização econômica. In: MARTINS, T. B; BATISTA, S. S. dos S. **Concepções e práticas nas políticas educacionais**: o ensino superior e a formação técnica e tecnológica. Piracicaba-SP: UNIMEP, 2022.

[2] WORLD ECONOMIC FORUM (WEF). **The future of jobs report 2020**. Switzerland, october 2020. Disponível em: https://www.weforum.org/reports/the-future-of-jobs-report-2020/. Acesso em: fev. 2022.

[3] SHI, X.; CHEN, Y. New teaching method of Python programming for liberal arts students. **International Journal of Innovation and Research in Educational Sciences**, vol. 7, issue 3, p. 261-271, 2020. Disponível em: https://www.ijires.org/index.php/issues?view=publication&task=show&id=584. Acesso em: ago. 2022.

[4] PUGH, D. R. **Python for research and teaching economics**. Proceedings of the 13th Python in Science Conference. Disponível em: https://conference.scipy.org/proceedings/scipy2014/pdfs/pugh.pdf. Acesso em: fev. 2022.

[5] ARUOBA, S. B.; FERNÁNDEZ-VILLAVERDE, J. **A comparison of programming languages in economics**. NBER Working Paper Series, June 2014. Disponível em: https://www.nber.org/papers/w20263. Acesso em: mar. 2022.

[6] ROMERO-AGUILLAR, R. **Python for economists**. Disponível em: http://randall-romero.com/wp-content/uploads/Notes/Using-Python-CompEcon-English.pdf. Zietz, J., 2007. Dynamic programming: an introduction by example. J. Econ. Educ. 38 (2), 165–186. Acesso em: abr. 2022.

[7] KUROKI, M. Using Python and Google Colab to teach undergraduate microeconomic theory. **International Review of Economics Education**, 38, 2021. Disponível em: https://doi.org/10.1016/j.iree.2021.100225. Acesso em: fev. 2022.

[8] GUTTAG, J. V. **Introduction to computation and programming using Python**: with application to understanding data. 2nd ed. Cambridge, MA: The MIT Press, 2017. E-book.

[9] DOWNEY, A. **Think Python**: how to think like a computer scientist. 2nd ed, v. 2.4.0. Needham, MA: Green Tea Press, 2015. E-book.

[10] BELL, A. **Python for economists**. Harvard, 2016. Disponível em: https://scholar.harvard.edu/files/ambell/files/python_for_economists.pdf. Acesso em: mar. 2022.

[11] GUST-BARDON, N. I. **Python for economists**. Disponível em: https://natalia.irena.gust-bardon.org/projects/gust-bardon-python-for-economists.pdf. Acesso em: mar. 2022.

[12] MARCONDES, G. A. B. **Matemática com Python**: um guia prático. São Paulo: Novatec, 2018.

[13] PINDYCK, R. S.; RUBINFELD, D. L. **Microeconomia**. 6 ed. São Paulo: Pearson Prentice Hall, 2005.

[14] MANKIW, N. G. **Princípios de microeconomia**. São Paulo: Cengage Learning, 2013.