Sistema de Equações Lineares

Mário Leite

...

Segundo SANTOS (2000, p97) "*equação linear é toda equação escrita na forma:*
***ax_1 + ax_2 + ax_3 + ... + ax_n,* = b** *em que a_1, a_2, a_3,...,a_n são números reais que recebem o nome de coeficientes das incógnitas x_1, x_2, x_3,...,x_n, e b é um número real chamado de termo indepdendente*".
As equações abaixo são exemplos de equações lineares.

- **3x + 4y = 5**
- **x - 5y = 3**
- **x + y -2z = 12**

As duas primeiras são equações com duas incógnitas (x e y); já a terceira é uma equação com três incógnitas (x, y z). Juntas, essas equações poem formar um sistema de equações lineares.

Agora observe as seguintes equações:

- **x^2 + 10x + 2y - z = 120**
- **seno(x) + 3.cos(y) + 2z = 12**

Neste último exemplo, as três equações não formam um sistema linear, pois além do expoente das incónitas ser maior que 1 também existem funções trigionométricas envolvidas.

Sistemas Lineares

Ainda segundo SANTOS(2000, p217) "*sistema lienar é um conjunto de equações da forma*":

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + a_{13}x_3 + \ldots + a_{1n}x_n = b_1 \\ a_{21}x_1 + a_{22}x_2 + a_{23}x_3 + \ldots + a_{2n}x_n = b_2 \\ \ldots \quad \ldots \quad \ldots \quad \ldots \quad \ldots \\ a_{m1}x_1 + a_{m2}x_2 + a_{m3}x_3 + \ldots + a_{mn}x_n = b_m \end{cases}$$

A solução de um sistema desse tipo é encontrar **a_{ij}** tais façam com que os primeiros termos das equações fiquem idênticos aos segundos termos. Seja, por exemplo, o sistema de três equações e três incógnitas mostrado abaixo:

$$\begin{cases} x + 2y - 3z = 9 \\ 2x + y + z = 0 \\ 3x - y + 4z = -5 \end{cases}$$

Os valores de x, y e z que resolvem esse sistema são:

x = 2
y = -1
z = -3

Existem várias técnicas para resolver um sistema linear de equações; a "Regra de Cramer" é uma delas. Uma maneira prática de resolver um sistema linear é empregar a ferramenta **SciLab** ou congêneres. Mas, podemos criar um programa, codificados em Python que resolvem um sistema linear de **n** equações a **n** incógnitas de maneira relativamente simples. No exemplo dado, resolvemos um sistema de 4 equações a 4 incógnitas com o programa **"SistemaNEquaçoes"**.

```python
'''
SistemaNEquaçoes.py
------------------------------------------------------------------------------
Resolve um sistema linear de N equações com N incógnitas, na forma matricial.
------------------------------------------------------------------------------
'''
def ResolverSistema(LstMatA, LstVetB):
    """
    Resolve um sistema de equações lineares da forma Ax = b, onde A é
    uma matriz quadrada e b um vetor.

    Argumentos:
        LstMatA: (lista): A matriz A do sistema linear (NxN).
        LstVetB: (lista}: O vetor b do sistema linear Nx1.

    Retornos:
        List: O vetor x da solução do sistema linear.
    """
#------------------------------------------------------------------------------
    #Verifica se a matriz A é quadrada.
    if(len(LstMatA) != len(LstMatA[0])):
        raise ValueError("A matriz A deve ser quadrada!")

    #Verifica se o número de equações é igual ao número de incógnitas.
    if(len(LstMatA) != len(LstVetB)):
        raise ValueError("O número de equações deve ser igual ao número
        de incógnitas!")

    #Cria uma matriz ampliada [A | b].
    LstMatApli = []
    for i in range(len(LstMatA)):
        LstMatApli.append(LstMatA[i] + [LstVetB[i]])

    #Reduz a matriz ampliada à forma escalonada reduzida.
    for i in range(len(LstMatA)):
       #Procura o maior elemento em cada coluna, a partir da linha atual.
        maxElem = abs(LstMatApli[i][i])
        maxLin = i
        for j in range(i+1, len(LstMatA)):
            if(abs(LstMatApli[j][i]) > maxElem):
                maxElem = abs(LstMatApli[j][i])
                maxLin = j

        #Troca as linhas atual e a linha com o maior elemento.
        if(maxLin != i):
            LstMatApli[i], LstMatApli[maxLin] = LstMatApli[maxLin],
            LstMatApli[i]

        #Divide a linha atual pelo elemento diagonal.
        divisor = LstMatApli[i][i]
        if(divisor != 0):
            for j in range(i, len(LstMatApli[0])):
                LstMatApli[i][j] /= divisor
        #Subtrai a linha atual de todas as outras linhas.
        for j in range(len(LstMatA)):
            if(j != i):
```

```python
                multiplier = LstMatApli[j][i]
                for k in range(i, len(LstMatApli[0])):
                    LstMatApli[j][k] -= multiplier * LstMatApli[i][k]

    #Verifica se o sistema é Impossível ou Indeterminado.
    ehImpossivel = False
    ehIndeterminado = False
    for i in range(len(LstMatA)):
        if(LstMatApli[i][-1] == 0 and any(LstMatApli[i][j] != 0
            for j in range(len(LstMatA)))):
            ehImpossivel = True
        elif(LstMatApli[i][-1] != 0 and all(LstMatApli[i][j] == 0
            for j in range(len(LstMatA)))):
            ehIndeterminado = True

    #Extrai o vetor x da solução do sistema linear.
    LstVetX = []
    for i in range(len(LstMatA)):
        LstVetX.append(round(LstMatApli[i][-1], 2))

    #Retorna o vetor x da solução do sistema linear formatado
    print("Solução do Sistema:")
    if(ehImpossivel):
        return "Sistema Impossível"
    elif(ehIndeterminado):
        return "Sistema Indeterminado"
    else:
        return LstVetX

#============================================================================
#Programa principal
#Exemplo para um Sistema Linear 4x4.
LstMatA = [[4, 2, 1, -2], [3, -3, -1, -1], [3, 5, 1, 1], [1, -1, -1, 4]]
LstVetB = [3, 2, 0, -2]

solucao = ResolverSistema(LstMatA, LstVetB)
print(solucao)
#Fim do programa "SistemaNEquaçoes" -----------------------------------------
```

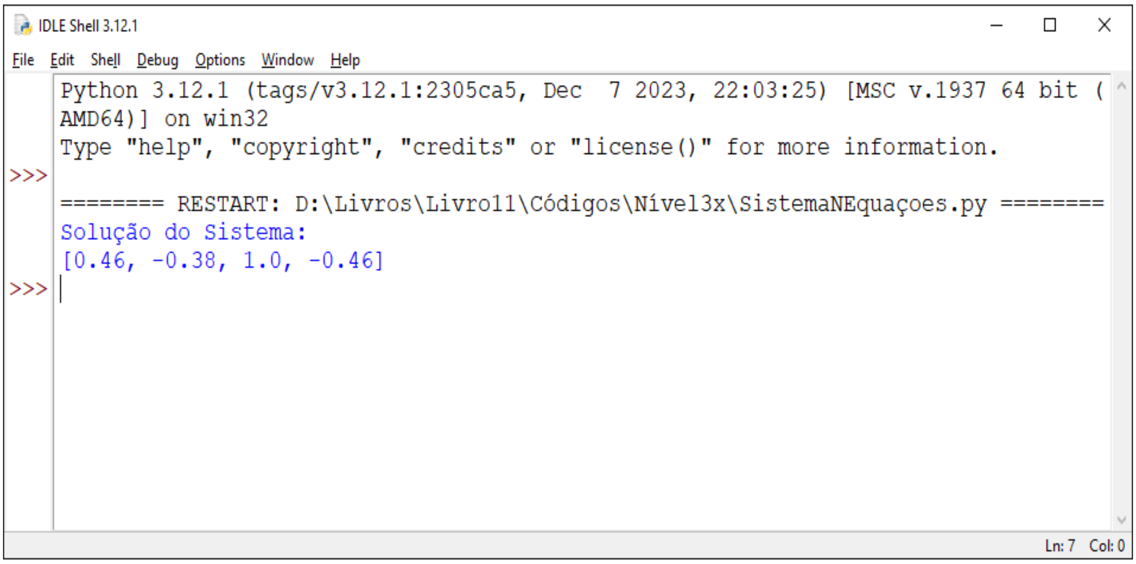

Figura 1 - Saída do programa "SistemaNEquaçoes"