MAP0214 - Cálculo Numérico com Aplicações em Física
EP02

Guilherme Pacheco Paredes N°USP: 12693754
Instituto de Física, Universidade de São Paulo - SP

28/09/2024

Os exercícios computacionais deste Ep foram escritos em código Python.

1

a) Aplicando as leis de Kirchhoff, nós obtemos, para as duas malhas:

$$U_3 - U_2 - R_4 I_3 + R_3 I_2 = 0 \tag{1}$$
$$U_2 + R_2 I_1 - U_1 + R_1 I_1 + R_4 I_3 = 0$$
$$\implies U_2 + (R_1 + R_2) I_1 + R_4 I_3 - U_1 = 0 \tag{2}$$

Aplicando a Lei dos Nós, temos a seguinte relação:

$$I_1 - I_2 - I_3 = 0. \tag{3}$$

Portanto, obtemos as três equações lineares são

$$\begin{cases} 0.0 I_1 + 5.3 I_2 - 1.8 I_3 & = 3.1 \\ 11.9 I_1 + 0.0 I_2 + 1.8 I_3 & = 15.0 \\ I_1 - I_2 - I_3 & = 0 \end{cases} \tag{4}$$

que na forma matricial é dado por

$$\begin{bmatrix} 0.0 & 5.3 & -1.8 \\ 11.9 & 0.0 & 1.8 \\ 1.0 & -1.0 & -1.0 \end{bmatrix} \begin{bmatrix} I_1 \\ I_2 \\ I_3 \end{bmatrix} = \begin{bmatrix} 3.1 \\ 15.0 \\ 0.0 \end{bmatrix} \tag{5}$$

---

b) Pelo método da Eliminação de Gauss e utilizando do pivotamento parcial, foi criado o seguinte programa:

```
def eliminacao_gauss(A, b):
    n = len(b)

    # Criar a matriz aumentada
    Ab = [linha + [b[i]] for i, linha in enumerate(A)]
    print("Matriz aumentada inicial:")
    for linha in Ab:
        print(" ".join(f"{elem:8.1f}" for elem in linha))
    print()

    # Etapa 1: Eliminacao para forma escalonada com pivotamento parcial
    for i in range(n):
        # Encontrar o pivo e trocar linhas se necessario
        max_linha = i
        for k in range(i + 1, n):
            if abs(Ab[k][i]) > abs(Ab[max_linha][i]):
                max_linha = k
        if i != max_linha:
            Ab[i], Ab[max_linha] = Ab[max_linha], Ab[i]
            print(f"\nTrocando linha {i + 1} com linha {max_linha + 1}:")
            for linha in Ab:
                print(" ".join(f"{elem:8.1f}" for elem in linha))
            print()

        # Zerar elementos abaixo do pivo
        for j in range(i + 1, n):
            if Ab[j][i] != 0:  # Verifica se o elemento abaixo do pivo nao e zero
                fator = Ab[j][i] / Ab[i][i]
                print(f"\nEliminando elemento na linha {j + 1}, coluna {i + 1}"
                      f"com fator {fator:.7f}:")
                for k in range(i, n + 1):
                    Ab[j][k] -= fator * Ab[i][k]
                for linha in Ab:
                    print(" ".join(f"{elem:8.1f}" for elem in linha))
                print()

    # Etapa 2: Substituicao reversa para resolver as variaveis
    x = [0] * n
    for i in reversed(range(n)):
        x[i] = Ab[i][n] / Ab[i][i]
        print(f"\nCalculando x{i + 1} = {Ab[i][n]:.4f} / {Ab[i][i]:.4f} = {x[i]:.7f}")
        for j in range(i + 1, n):
            x[i] -= Ab[i][j] * x[j] / Ab[i][i]
            print(f"\nAjustando x{i + 1} com x{j + 1} = {x[j]:.7f}")
        print()

    return x

A = [[0.0, 5.3, -1.8],[11.9, 0.0, 1.8],[1, -1, -1]]

b = [3.1, 15.0, 0]

# Resolve o sistema
solucao = eliminacao_gauss(A, b)
print("Solucao:")
print(f"I1 = {solucao[0]:.7f}")
print(f"I2 = {solucao[1]:.7f}")
print(f"I3 = {solucao[2]:.7f}")
```

A seguir, a saída do código:

```
Matriz aumentada inicial:
     0.0      5.3     -1.8      3.1
    11.9      0.0      1.8     15.0
     1.0     -1.0     -1.0      0.0


Trocando linha 1 com linha 2:
    11.9      0.0      1.8     15.0
     0.0      5.3     -1.8      3.1
     1.0     -1.0     -1.0      0.0


Eliminando elemento na linha 3, coluna 1 com fator 0.0840336:
    11.9      0.0      1.8     15.0
     0.0      5.3     -1.8      3.1
     0.0     -1.0     -1.2     -1.3


Eliminando elemento na linha 3, coluna 2 com fator -0.1886792:
    11.9      0.0      1.8     15.0
     0.0      5.3     -1.8      3.1
     0.0      0.0     -1.5     -0.7

Calculando x3 = -0.6755985 / -1.4908831 = 0.4531532

Calculando x2 = 3.1000000 / 5.3000000 = 0.5849057
Ajustando x2 com x3 = 0.4531532

Calculando x1 = 15.0000000 / 11.9000000 = 1.2605042
Ajustando x1 com x2 = 0.7388068
Ajustando x1 com x3 = 0.4531532

Solução:
I1 = 1.1919600
I2 = 0.7388068
I3 = 0.4531532
```

O sistema requereu a troca das linhas 1 e 2 por pivotamento parcial e duas operações. Uma entre as linhas 1 e 2 e outra entre as linhas 1 e 3. As soluções obtidas por esse método foram $I_1 = 1.1919600\text{A}$, $I_2 = 0.7388068\text{A}$ e $I_3 = 0.4531532\text{A}$, limitadas a 7 decimais.

c) Pelo método de Jacobi definindo o critério de parada como $\max|x_i^{k+1} - x_i^k| < \epsilon$ e $\epsilon = 10^{-3}$ o seguinte programa foi implementado:

```
def jacobi(A, b, x0, epsilon=1e-3):

    n = len(A)
    x = x0[:]
    dados_iteracao = []

    while True:
        # Zerando o vetor
        x_novo = [0.0] * n
        # Armazena todas as n atualizações antes de substituir o vetor x original
        for i in range(n):
            # Soma dos elementos anteriores
            s1 = sum(A[i][j] * x[j] for j in range(i))
            # Soma dos elementos seguintes
            s2 = sum(A[i][j] * x[j] for j in range(i + 1, n))
            # Atualização de x_novo[i]
            x_novo[i] = (b[i] - s1 - s2) / A[i][i]

        # Critério de parada baseado na diferença máxima
        max_dif = max(abs(x_novo[i] - x[i]) for i in range(n))

        dados_iteracao.append((len(dados_iteracao) + 1, x_novo[0], x_novo[1],
                               x_novo[2], max_diff))

        if max_dif < epsilon:
            break

        # x se torna o x mais recente
        x = x_novo[:]

    return x_novo, dados_iteracao


# Matriz A e vetor b fornecidos
A = [[11.9, 0.0, 1.8], [0.0, 5.3, -1.8], [1.0, -1.0, -1.0]]
b = [15.0, 3.1, 0.0]

# Vetor inicial
x0 = [1.0, 1.0, 1.0]

# Resolução do sistema
solucao, dados_iteracao = jacobi(A, b, x0)

# Imprimir a tabela de iterações
print(f"\n|{'Iteração':^10}|{'I1':^10}|{'I2':^10}|{'I3':^10}|{'Erro':^10}|")
print("-" * 56)

for dado in dados_iteracao:
    print(f"|{dado[0]:^10}|{dado[1]:^10.7f}|{dado[2]:^10.7f}|{dado[3]:^10.7f}|"
          f"{dado[4]:^10.7f}|")

# Imprimir a solução
print("\nSolução:")
print(f"I1 = {solucao[0]:.7f}")
print(f"I2 = {solucao[1]:.7f}")
print(f"I3 = {solucao[2]:.7f}")
```

A seguir, a saída do código:

```
| Iteração |    I1    |    I2    |    I3     |   Erro   |
-----------------------------------------------------------
|    1     |1.1092437 |0.9245283 |-0.0000000 |1.0000000 |
|    2     |1.2605042 |0.5849057 | 0.1847154 |0.3396226 |
|    3     |1.2325641 |0.6476392 | 0.6755985 |0.4908831 |
|    4     |1.1583128 |0.8143542 | 0.5849249 |0.1667150 |
|    5     |1.1720282 |0.7835594 | 0.3439586 |0.2409663 |
|    6     |1.2084768 |0.7017218 | 0.3884688 |0.0818376 |
|    7     |1.2017442 |0.7168385 | 0.5067551 |0.1182863 |
|    8     |1.1838522 |0.7570112 | 0.4849058 |0.0401727 |
|    9     |1.1871571 |0.7495906 | 0.4268410 |0.0580647 |
|    10    |1.1959400 |0.7298705 | 0.4375665 |0.0197201 |
|    11    |1.1943177 |0.7335131 | 0.4660695 |0.0285030 |
|    12    |1.1900063 |0.7431934 | 0.4608045 |0.0096803 |
|    13    |1.1908027 |0.7414053 | 0.4468129 |0.0139916 |
|    14    |1.1929191 |0.7366534 | 0.4493974 |0.0047519 |
|    15    |1.1925281 |0.7375312 | 0.4562656 |0.0068683 |
|    16    |1.1914892 |0.7398638 | 0.4549970 |0.0023326 |
|    17    |1.1916811 |0.7394329 | 0.4516254 |0.0033715 |
|    18    |1.1921911 |0.7382879 | 0.4522482 |0.0011450 |
|    19    |1.1920969 |0.7384994 | 0.4539032 |0.0016550 |
|    20    |1.1918466 |0.7390615 | 0.4535975 |0.0005621 |

Solução:
I1 = 1.1918466
I2 = 0.7390615
I3 = 0.4535975
```

Por este método foram realizadas 20 iterações e foram obtidos os valores $I_1 = 1.1918466\text{A}$, $I_2 = 0.7390615\text{A}$ e $I_3 = 0.4535975\text{A}$ como soluções do sistema, limitadas novamente a 7 decimais.

---

d) A matriz $\mathbb{J}$ é dada pela relação

$$\mathbb{J} = -D^{-1}(\mathbb{L}+\mathbb{U}) = -\begin{bmatrix} 0 & \frac{a_{12}}{a_{11}} & \frac{a_{13}}{a_{11}} & \dots & \frac{a_{1n}}{a_{11}} \\ \frac{a_{21}}{a_{22}} & 0 & \frac{a_{23}}{a_{22}} & \dots & \frac{a_{2n}}{a_{22}} \\ \frac{a_{31}}{a_{33}} & \frac{a_{32}}{a_{33}} & 0 & \dots & \frac{a_{3n}}{a_{33}} \\ \vdots & \vdots & \ddots & 0 & \vdots \\ \frac{a_{n1}}{a_{nn}} & \frac{a_{n2}}{a_{nn}} & \dots & \frac{a_{n,n-1}}{a_{nn}} & 0 \end{bmatrix} \tag{6}$$

tal que

$$A = \mathbb{L} + D + \mathbb{U}, \tag{7}$$

em que $\mathbb{L}$ é a matriz triangular inferior sem diagonal, $\mathbb{U}$ é a matriz triangular superior sem diagonal, D é a matriz diagonal e A é a matriz do sistema em questão. Assim, seja

$$A = \begin{bmatrix} 11.9 & 0.0 & 1.8 \\ 0.0 & 5.3 & -1.8 \\ 1.0 & 1.0 & -1.0 \end{bmatrix} \tag{8}$$

temos que D será:

$$D = \begin{bmatrix} 11.9 & 0.0 & 0.0 \\ 0.0 & 5.3 & 0.0 \\ 0.0 & 0.0 & -1.0 \end{bmatrix} \implies D^{-1} = \begin{bmatrix} \frac{1.0}{11.9} & 0.0 & 0.0 \\ 0.0 & \frac{1.0}{5.3} & 0.0 \\ 0.0 & 0.0 & -\frac{1.0}{1.0} \end{bmatrix} \tag{9}$$

Para $\mathbb{L}$:

$$\mathbb{L} = \begin{bmatrix} 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 \\ 1.0 & -1.0 & 0.0 \end{bmatrix} \tag{10}$$

Para $\mathbb{U}$:

$$\mathbb{U} = \begin{bmatrix} 0.0 & 0.0 & 1.8 \\ 0.0 & 0.0 & -1.8 \\ 0.0 & 0.0 & 0.0 \end{bmatrix} \tag{11}$$

Portanto, a matriz $\mathbb{J}$ será:

$$\mathbb{J} = -\begin{bmatrix} 0.0 & 0.0 & \frac{1.8}{11.9} \\ 0.0 & 0.0 & \frac{-1.8}{5.3} \\ -\frac{1.0}{1.0} & \frac{-1.0}{-1.0} & 0.0 \end{bmatrix} = \begin{bmatrix} 0.00 & 0.00 & -0.15 \\ 0.00 & 0.00 & 0.34 \\ 1.00 & -1.00 & 0.00 \end{bmatrix} \tag{12}$$

é possível obter os autovalores da matriz $\mathbb{J}$, e foram obtidos $\lambda_1 = 0.700631i \implies \lambda_1 = 0.70i$, $\lambda_2 = -0.700631i \implies \lambda_2 = -0.70i$ e $\lambda_3 = 0$ e portanto o raio espectral, sendo o módulo do maior autovalor, é $\rho_s = 0.70$. Dada a relação

$$\rho_s^k \approx 10^{-p} \tag{13}$$

em que $p = 3$ é a precisão, então $k \approx 19.4161 \implies k = 19$. De fato, o número de iterações realizado foi 20, portanto a convergência obedece a relação (13).

e) Pelo método de Gauss-Seidel, o seguinte programa foi implementado:

```
def gauss_seidel(A, b, x0, epsilon=1e-3):
    n = len(A)
    x = x0[:]
    dados_iteracao = []

    while True:
        # Vetor anterior recebendo x a cada iteracao
        x_ant = x[:]
        for i in range(n):
            # Soma dos elementos anteriores
            s1 = sum(A[i][j] * x[j] for j in range(i))
            # Soma dos elementos seguintes
            s2 = sum(A[i][j] * x[j] for j in range(i + 1, n))
            # Atualiza x[i] imediatamente
            x[i] = (b[i] - s1 - s2) / A[i][i]

        # Cálculo do critério de parada baseado na diferença máxima
        max_dif = max(abs(x[i] - x_ant[i]) for i in range(n))

        dados_iteracao.append((len(dados_iteracao) + 1, x[0], x[1], x[2], max_dif))

        if max_dif < epsilon:
            break


    return x, dados_iteracao

# Matriz A e vetor b fornecidos
A = [[11.9, 0.0, 1.8], [0.0, 5.3, -1.8], [1.0, -1.0, -1.0]]
b = [15.0, 3.1, 0.0]

# Vetor inicial
x0 = [1.0, 1.0, 1.0]

# Resolução do sistema
solucao, dados_iteracao = gauss_seidel(A, b, x0)

# Imprimir a tabela de iterações
print(f"\n|{'Iteração':^10}|{'I1':^10}|{'I2':^10}|{'I3':^10}|{'Erro':^10}|")
print("-" * 56)
for dado in dados_iteracao:
    print(f"|{dado[0]:^10}|{dado[1]:^10.7f}|{dado[2]:^10.7f}|{dado[3]:^10.7f}|"
          f"{dado[4]:^10.7f}|")

# Imprimir a solução
print("\nSolução:")
print(f"I1 = {solucao[0]:.7f}")
print(f"I2 = {solucao[1]:.7f}")
print(f"I3 = {solucao[2]:.7f}")
```

A seguir, a saída do código:

```
| Iteração |    I1    |    I2    |    I3    |   Erro   |
--------------------------------------------------------
|    1     |1.1092437 |0.9245283 |0.1847154 |0.8152846 |
|    2     |1.2325641 |0.6476392 |0.5849249 |0.4002095 |
|    3     |1.1720282 |0.7835594 |0.3884688 |0.1964561 |
|    4     |1.2017442 |0.7168385 |0.4849058 |0.0964370 |
|    5     |1.1871571 |0.7495906 |0.4375665 |0.0473393 |
```

```
|    6     |1.1943177 |0.7335131 |0.4608045 |0.0232381 |
|    7     |1.1908027 |0.7414053 |0.4493974 |0.0114072 |
|    8     |1.1925281 |0.7375312 |0.4549970 |0.0055996 |
|    9     |1.1916811 |0.7394329 |0.4522482 |0.0027487 |
|    10    |1.1920969 |0.7384994 |0.4535975 |0.0013493 |
|    11    |1.1918928 |0.7389576 |0.4529352 |0.0006624 |

Solução:
I1 = 1.1918928
I2 = 0.7389576
I3 = 0.4529352
```

Por esse método, foi obtido os valores $I_1 = 1.1918928$A, $I_2 = 0.7389576$A e $I_3 = 0.4529352$A como soluções do sistema, limitadas novamente a 7 decimais.