MS211 - Cálculo Numérico

Projeto 1 - Fatoração de Cholesky

Letícia Lopes Mendes da Silva - RA: 184423
Iran Seixas Lopes Neto - RA: 244827

Exercício

Item (a)

A matriz A é simétrica pois $A = A^T$, isto é $a_{ij} = a_{ji}$, $\forall i, j = 1, 2$.
Para que A seja definida positiva, $x^T A x > 0$, $\forall x \neq 0 \in \mathbb{R}^2$. Vamos fazer essa verificação:

$$x^T A x = \begin{pmatrix} x_1 & x_2 \end{pmatrix} \begin{pmatrix} 4 & 6 \\ 6 & 13 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 4x_1 + 6x_2 & 6x_1 + 13x_2 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$$

$$= x_1(4x_1 + 6x_2) + x_2(6x_1 + 13x_2)$$

$$= 4x_1^2 + 6x_1x_2 + 6x_2x_1 + 13x_2^2 = (4x_1^2 + 12x_1x_2 + 9x_2^2) + (4x_2^2)$$

$$= (2x_1 + 3x_2)^2 + (2x_2)^2 > 0$$

Como os termos da expressão são quadráticos e $x_1, x_2 \in \mathbb{R}$, então $x^T A x > 0$, $\forall x \neq 0 \in \mathbb{R}^2$, e, logo, a matriz A é definida positiva.

Item (b)

Fazendo a multiplicação de G por sua transposta, temos:

$$GG^T = \begin{pmatrix} g_{11} & 0 \\ g_{21} & g_{22} \end{pmatrix} \begin{pmatrix} g_{11} & g_{21} \\ 0 & g_{22} \end{pmatrix} = \begin{pmatrix} g_{11}^2 & g_{11}g_{21} \\ g_{11}g_{21} & g_{21}^2 + g_{22}^2 \end{pmatrix}$$

Item (c)

Comparando os elementos de A com GG^T, temos:

$$A = GG^T \Rightarrow \begin{pmatrix} 4 & 6 \\ 6 & 13 \end{pmatrix} = \begin{pmatrix} g_{11}^2 & g_{11}g_{21} \\ g_{11}g_{21} & g_{21}^2 + g_{22}^2 \end{pmatrix}$$

$$\Rightarrow \begin{cases} g_{11}^2 = 4 \\ g_{11}g_{21} = 6 \\ g_{21}^2 + g_{22}^2 = 13 \end{cases}$$

Da fatoração de Cholesky, G é definida como uma matriz triangular inferior, com a mesma dimensão de A e elementos da diagonal estritamente positivos, isto é, $g_{11} > 0$ e $g_{22} > 0$. Assim, podemos resolver o sistema:

$$\begin{cases} g_{11} = 2 \\ g_{11}g_{21} = 6 \\ g_{21}^2 + g_{22}^2 = 13 \end{cases} \Rightarrow \begin{cases} g_{11} = 2 \\ g_{21} = 3 \\ g_{21}^2 + g_{22}^2 = 13 \end{cases} \Rightarrow \begin{cases} g_{11} = 2 \\ g_{21} = 3 \\ g_{22} = 2 \end{cases}$$

$$\therefore G = \begin{pmatrix} 2 & 0 \\ 3 & 2 \end{pmatrix}$$

Item (d)

Substituindo os valores encontrados para g_{11}, g_{21} e g_{22}, podemos efetuar a multiplicação GG^T. Assim:

$$GG^T = \begin{pmatrix} g_{11}^2 & g_{11}g_{21} \\ g_{11}g_{21} & g_{21}^2 + g_{22}^2 \end{pmatrix} = \begin{pmatrix} 2^2 & 2.3 \\ 2.3 & 3^2 + 2^2 \end{pmatrix} = \begin{pmatrix} 4 & 6 \\ 6 & 13 \end{pmatrix} = A$$

Nos itens (e) e (f), utilizamos códigos na linguagem Python. Implementamos as seguintes funções para rápida leitura dos dados e impressão dos resultados:

```
import numpy as np

def read_matrix(n):
    print("A:")
    A = []
    for i in range(n):
        row = input().split(' ')
        for j in range(n):
            row[j] = int(row[j])
        A.append(row)
    return A

def print_matrix(M: list[list[float]], name: str):
    print(name + ":")
    for row in M:
        print(" ".join(map(str, row)))
```

Item (e)

Para construir o algoritmo da Fatoração de Cholesky para uma matriz $A_{n\times n}$, vamos partir de $A = GG^T$, em que o fator $G_{n\times n}$ é uma matriz triangular inferior com $g_{ii} > 0$.

Sabemos que cada elemento de uma matriz A, gerada pela multiplicação PQ, é dado por:

$$a_{ij} = \sum_{k=1}^{n} p_{ik} q_{kj}$$

Tomando $PQ = GG^T$, temos que $p_{ij} = q_{ji}$. Assim:

$$a_{ij} = \sum_{k=1}^{n} g_{ik} g_{jk}$$

Como A é simétrica, basta que iteremos sobre cada elemento a_{ij} por uma metade da matriz para resolver o sistema de equações. Vamos utilizar apenas a sua metade inferior de linha a linha, isto é, $i = 1, 2, .., n$, $j = 1, 2, .., i$. Consideremos também que a matriz G é inicializada com elementos nulos.

Como queremos descobrir g_{ij}, vamos isolar este elemento na equação acima:

$$a_{ij} = \sum_{k=1}^{j-1} g_{ik} g_{jk} + g_{ij} g_{jj} + \sum_{k=j+1}^{n} g_{ik} g_{jk}$$

Como G é uma matriz triangular inferior, $g_{jk} = 0$ para $k > j$. Ou seja, o somatório à direita é igual a zero, podendo ser descartado. Assim:

$$a_{ij} = \sum_{k=1}^{j-1} g_{ik} g_{jk} + g_{ij} g_{jj}$$

$$g_{ij} g_{jj} = a_{ij} - \sum_{k=1}^{j-1} g_{ik} g_{jk} = s_{ij}$$

Com isso, podemos obter g_{ij}, a depender do valor de j. Se $j < i$, então g_{jj} é o pivô da linha j, que já foi encontrado; logo, basta dividir s_{ij} por g_{jj}. Agora, se $j = i$, então g_{jj} é g_{ij} e $s_{ij} = g_{ij}^2$; logo, basta calcular a raiz quadrada de s_{ij}. Isto é:

$$g_{ij} = \begin{cases} \dfrac{s_{ij}}{g_{jj}}, & j < i \\ \sqrt{s_{ij}}, & j = i \end{cases}$$

Partindo deste raciocínio, construímos o algoritmo abaixo, que obtém o fator de Cholesky G a partir de uma matriz A simétrica e definida positiva.

```python
def cholesky(n: int, A: list[list[int]]) -> list[list[float]]:
    '''
    Computes the Cholesky decomposition of a matrix A.
    Args:
        n (int): Dimension of matrix A.
        A (list[list[int]]): Matrix A.
    Returns:
        G (list[list[float]]): Cholesky factor G.
    '''
    G = [[0 for _ in range(n)] for _ in range(n)]
    for i in range(n):
        for j in range(i+1):
            s = A[i][j]
            # Computes the sum g_ik.g_jk, k < j, then subtracts from a_ij
            for k in range(j):
                s -= G[i][k] * G[j][k]
            if j < i:
                G[i][j] = s / G[j][j]     # Divides the sum by g_jj
            else:    # j == i
                G[i][j] = np.sqrt(s)      # Computes the square root of the
                                                    sum
    return G
```

```python
def main():
    n = int(input("n: "))
    A = read_matrix(n)
    G = cholesky(n, A)
    # Output
    print_matrix(G, "G")
```

Item (f)

Para uma matriz A ser definida positiva, o fator de Cholesky G gerado deve ter a diagonal principal composta apenas de números reais positivos. Assim, podemos criar a condição $g_{ij} > 0$, para $j = i$. Isto é:

$$\sqrt{s_{ii}} > 0 \Rightarrow s_{ii} > 0$$

Caso essa condição não seja cumprida em algum momento, o algoritmo irá falhar, interrompendo a iteração, e retornará que a matriz A não é definida positiva.

Para isso, vamos utilizar uma variável booleana "positive", que inicia verdadeira e se torna falsa caso $s_{ii} \leq 0$ em algum momento da iteração. Ela é responsável por encerrar o algoritmo caso necessário e indicar o resultado a ser impresso.

Assim, utilizando a mesma estrutura do código do item (e), adicionando apenas a condição mencionada, construímos o código abaixo.

```python
def cholesky(n: int, A: list[list[int]]) -> tuple[list[list[float]],bool]:
    '''
    Checks if matrix A is positive definite.
    If so, computes Cholesky factor G.
    Args:
        n (int): Dimension of matrix A.
        A (list[list[int]]): Matrix A.
    Returns:
        tuple: Contains:
            - G (list[list[float]]): Cholesky factor G.
            - positive (bool): Whether matrix A is positive definite.
    '''
    G = [[0 for _ in range(n)] for _ in range(n)]
    positive = True
    for i in range(n):
        for j in range(i+1):
            s = A[i][j]
            # Computes the sum g_ik.g_jk, k < j, then subtracts from a_ij
            for k in range(j):
                s -= G[i][k] * G[j][k]
            if j < i:
                G[i][j] = s / G[j][j]        # Divides by g_jj
            else:            # j == i
                if s > 0:    # Since g_ii is a positive real number
                    G[i][j] = np.sqrt(s)     # Computes the square root

                else:   # G does not exist, so A is not positive definite
                    positive = False
                    break
        if not positive:
            break
    return G, positive
```

```python
def main():
    n = int(input("n: "))
    A = read_matrix(n)
    G, positive = cholesky(n, A)
    # Output
    if positive: print_matrix(G, "G")
    else: print("Error: A is not positive definite.")
```

Usando a matriz fornecida no enunciado:

$$A = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 2 & 0 & 5 & 6 \\ 3 & 5 & -1 & 7 \\ 4 & 6 & 7 & -2 \end{pmatrix}$$

obtemos que A não é uma matriz definida positiva, o que corresponde ao esperado, pois os seguintes elementos da diagonal não são positivos: $a_{33} = -1 < 0$ e $a_{44} = -2 < 0$.