Universidade Federal de Juiz de Fora

Pós-Graduação em Modelagem Computacional

Métodos Numéricos

Eduardo Santos de Oliveira Marques

Atividade 3
Sistemas Lineares

Juiz de Fora

2023

Questão 1. Ao resolver um sistema de equações lineares utilizando o método de Crout, a matriz de coeficientes é fatorada em $A = LU$, onde L é triangular inferior e U é triangular superior com elementos da diagonal principal iguais a 1. Sendo $det(A_k) \neq 0, k = 1, \ldots, n$, onde A_k é o menor principal de ordem k da matriz A. Mostre que se essas condições forem atendidas, então A pode ser fatorada em LU como sugerido no método de Crout.

Resolução:

Para demonstrar que uma matriz A pode ser fatorada em LU utilizando o método de Crout, com base nas condições dadas, utiliza-se a propriedade de que o determinante de uma matriz triangular é igual ao produto de seus elementos diagonais.

Dado que $det(A_k) \neq 0$ para todo $k = 1, \ldots, n$, pode-se afirmar que a matriz A é não singular, ou seja, seu determinante é diferente de zero. Isso garante que a fatoração LU é possível, já que se a matriz A fosse singular, a fatoração não seria viável. O processo de fatoração LU será demonstrado utilizando o método de Crout por indução.

Base da indução (k = 1): Quando $k = 1$, considera-se o menor principal de ordem 1, que é o primeiro elemento da matriz A, ou seja, $A_1 = [a_{11}]$. Como A_1 é um escalar (único elemento da matriz), seu determinante é igual a $a_{11} \neq 0$ de acordo com as condições dadas. Nesse caso, é possível fatorar a matriz A da seguinte forma:

$$A = LU = [a_{11}] = [1] \cdot [a_{11}]$$

onde $L = [1]$ (matriz triangular inferior com o único elemento 1) e $U = [a_{11}]$ (matriz triangular superior com o elemento da diagonal principal igual a a_{11}).

Hipótese da indução: Suponha-se que a matriz A pode ser fatorada em LU para todos os menores principais de ordem $k - 1$ até $k - 1 = p$.

Passo da indução: Considera-se o menor principal de ordem $k = p + 1$, ou seja, a matriz A_{p+1}.

$$A_{p+1} = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1p} & a_{1,p+1} \\ a_{21} & a_{22} & \cdots & a_{2p} & a_{2,p+1} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ a_{p1} & a_{p2} & \cdots & a_{pp} & a_{p,p+1} \\ a_{p+1,1} & a_{p+1,2} & \cdots & a_{p+1,p} & a_{p+1,p+1} \end{bmatrix}$$

Procura-se encontrar as matrizes L e U para esse menor principal. Seja L_{p+1} a matriz triangular inferior do menor principal A_{p+1} e U_{p+1} a matriz triangular superior desse menor principal. Pela fórmula da expansão do determinante por cofatores, é possível escrever o determinante de A_{p+1} da seguinte forma:

$$det(A_{p+1}) = a_{11} \cdot det(A_p) - a_{12} \cdot det(A_{p,1}) + a_{13} \cdot det(A_{p,2}) - \ldots + (-1)^{p+1} \cdot a_{1,p+1} \cdot det(A_{p,p})$$

Como $det(A_k) \neq 0$ para todo $k = 1, \ldots, n$, pode-se afirmar que $det(A_p) \neq 0$. Além disso, analisa-se o determinante dos menores principais de ordem p, ou seja, $det(A_{p,1}), det(A_{p,2}), \ldots, det(A_{p,p})$. Cada uma dessas matrizes tem ordem p e é uma submatriz de A_{p+1}, e a fatoração LU para elas já foi estabelecida pela hipótese de indução. Portanto, pode-se escrever:

$$A_{p,1} = L_{p,1} \cdot U_{p,1} \quad ; \quad A_{p,2} = L_{p,2} \cdot U_{p,2} \quad ; \quad \cdots \quad ; \quad A_{p,p} = L_{p,p} \cdot U_{p,p}$$

Agora, essas fatorações serão substituídas no determinante de A_{p+1}:

$$det(A_{p+1}) = a_{11} \cdot det(A_p) - a_{12} \cdot (det(L_{p,1}) \cdot det(U_{p,1})) + a_{13} \cdot (det(L_{p,2}) \cdot det(U_{p,2})) - \ldots + (-1)^{p+1} \cdot a_{1,p+1} \cdot (det(L_{p,p}) \cdot det(U_{p,p}))$$

Nota-se que os determinantes de $L_{p,i}$ e $U_{p,i}$, para $i = 1, \ldots, p$, são iguais a 1, pois essas matrizes têm diagonal principal com elementos iguais a 1, conforme mencionado nas condições dadas. Portanto, pode-se simplificar a expressão acima para:

$$det(A_{p+1}) = a_{11} \cdot det(A_p) - a_{12} \cdot 1 + a_{13} \cdot 1 - \ldots + (-1)^{p+1} \cdot a_{1,p+1} \cdot 1$$

$$det(A_{p+1}) = a_{11} \cdot det(A_p) - a_{12} + a_{13} - \ldots + (-1)^{p+1} \cdot a_{1,p+1}$$

Agora, considera-se a matriz L_{p+1}. Essa matriz é triangular inferior, com elementos da diagonal principal iguais a 1 (conforme condições dadas). Portanto, escreve-se:

$$L_{p+1} = \begin{bmatrix} 1 & 0 & \cdots & 0 & 0 \\ l_{21} & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ l_{p+1,1} & l_{p+1,2} & \cdots & 1 & 0 \end{bmatrix}$$

onde os elementos $l_{i,j}$, para $i = 2, \ldots, p+1$, são coeficientes a serem determinados.

Da mesma forma, considera-se a matriz U_{p+1}. Essa matriz é triangular superior, com elementos da diagonal principal iguais a 1 (conforme condições dadas). Portanto, escreve-se:

$$U_{p+1} = \begin{bmatrix} u_{11} & u_{12} & \cdots & u_{1,p+1} \\ 0 & u_{22} & \cdots & u_{2,p+1} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{bmatrix}$$

onde os elementos $u_{i,j}$, para $i = 1, \ldots, p+1$, são coeficientes a serem determinados.

Agora, é possível escrever a fatoração LU da matriz A_{p+1}: $A_{p+1} = L_{p+1} \cdot U_{p+1}$. Substituindo as matrizes L_{p+1} e U_{p+1} e seus elementos correspondentes, têm-se:

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1p} & a_{1,p+1} \\ a_{21} & a_{22} & \cdots & a_{2p} & a_{2,p+1} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ a_{p1} & a_{p2} & \cdots & a_{pp} & a_{p,p+1} \\ a_{p+1,1} & a_{p+1,2} & \cdots & a_{p+1,p} & a_{p+1,p+1} \end{bmatrix} = \begin{bmatrix} 1 & 0 & \cdots & 0 & 0 \\ l_{21} & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ l_{p+1,1} & l_{p+1,2} & \cdots & 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} u_{11} & u_{12} & \cdots & u_{1,p+1} \\ 0 & u_{22} & \cdots & u_{2,p+1} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{bmatrix}$$

Analisa-se a matriz resultante na multiplicação $L_{p+1} \cdot U_{p+1}$. Se calcular o produto dessas matrizes, terá que a matriz resultante é exatamente a matriz A_{p+1}, pois todas as operações envolvem apenas combinações lineares de linhas e colunas de L_{p+1} e U_{p+1}, e essas combinações são os elementos de A_{p+1}. Portanto, mostrou-se que é possível fatorar A_{p+1} em LU_{p+1}.

Questão 2. Implemente ao menos dois dos métodos numéricos discutidos em aula, um direto e um iterativo, para resolver sistemas lineares $Ax = b$. Sejam A e b disponíveis nos arquivos $\mathsf{A-N.dat}$ e $\mathsf{b-N.dat}$, respectivamente, onde $N \in \{4, 6, 8, 10\}$ indica as dimensões dos sistemas lineares.

Resolver os sistemas lineares e fornecer como saída: (i) o condicionamento da matriz de coeficientes, (ii) a solução encontrada e (iii) a diferença relativa em norma infinito entre a solução encontrada e a solução esperada do problema. Compare os resultados encontrados pelos métodos. A solução do problema sem considerar os arredondamentos na geração dos arquivos de dados é $x^T = [11\ 11\ \ldots\ 11]^T$.

Resolução:

Para resolver o sistema linear $Ax = b$, serão utilizados o método direto de Eliminação de Gauss e o método iterativo de Gauss-Seidel.

Antes de começar a implementação, os arquivos $\mathsf{A-N.dat}$ e $\mathsf{b-N.dat}$ precisam ser lidos para obter a matriz de coeficientes A e o vetor b correspondente. Em seguida, os dois métodos são implementados; calculando a solução do sistema linear, o condicionamento da matriz de coeficientes e a diferença relativa em norma infinito entre a solução encontrada e a solução esperada. Por fim, os resultados encontrados são comparados pelos dois métodos com a solução esperada.

Inicia-se com método direto de eliminação de Gauss (fatoração LU). Em seguida, implementaremos o método iterativo de Gauss-Seidel. Como o método iterativo pode ser mais suscetível a erros de precisão, usaremos o método direto como referência para a solução exata do sistema. Segue-se a implementação em Python:

```python
import numpy as np

def gaussian_elimination(A, b):
    N = A.shape[0]
    L = np.eye(N)
    U = A.copy()

    for i in range(N-1):
        if U[i,i] == 0:
            raise ValueError("A eliminacao gaussiana nao pode continuar.")

        for j in range(i+1, N):
            L[j, i] = U[j, i] / U[i, i]
            U[j, i:] = U[j, i:] - L[j, i] * U[i, i:]

    y = np.linalg.solve(L, b)
    x = np.linalg.solve(U, y)
    return x

# Carregar os dados de A e b para N = 4, 6, 8, 10
N_values = [4, 6, 8, 10]
for N in N_values:
    A = np.loadtxt(f"A-{N}.dat")
    b = np.loadtxt(f"b-{N}.dat")

    # Método Direto - Eliminacao de Gauss (Fatoracao LU)
    try:
        solution_direct = gaussian_elimination(A, b)
        condition_number = np.linalg.cond(A)
        expected_solution = np.ones(N) * 11

        # Calcular a diferenca relativa em norma infinito
        relative_difference = np.linalg.norm(solution_direct - expected_solution, np.inf) / np.linalg.norm(expected_solution, np.inf)

        # Imprimir os resultados
        print(f"\nN = {N}")
        print("Método Direto - Eliminacao de Gauss (Fatoracao LU)")
        print("Solucao encontrada:", solution_direct)
        print("Condicionamento da matriz de coef.:", condition_number)
        print("Diferenca relativa em norma infinito:", relative_difference)

    except Exception as e:
        print(f"\nN = {N}")
        print("Erro no Método Direto - Eliminacao de Gauss:")
        print(e)
```

```python
# ------------------------------------------------------------------------
def gauss_seidel(A, b, max_iterations=1000, tol=1e-8):
    N = A.shape[0]
    x = np.zeros(N)

    for _ in range(max_iterations):
        x_new = np.zeros(N)

        for i in range(N):
            x_new[i] = (b[i] - np.dot(A[i, :i], x_new[:i]) - np.dot(A[i, i+1:], x[i+1:])) / A[i, i]

        # Critério de convergencia
        if np.linalg.norm(x_new - x, np.inf) < tol:
            return x_new

        x = x_new
    raise ValueError("Gauss-Seidel nao convergiu com máximo de iteracoes.")

# Carregar os dados de A e b para N = 4, 6, 8, 10
N_values = [4, 6, 8, 10]
for N in N_values:
    A = np.loadtxt(f"A-{N}.dat")
    b = np.loadtxt(f"b-{N}.dat")

    # Método Iterativo - Gauss-Seidel
    try:
        solution_iterative = gauss_seidel(A, b)
        condition_number = np.linalg.cond(A)
        expected_solution = np.ones(N) * 11

        # Calcular a diferenca relativa em norma infinito
        relative_difference = np.linalg.norm(solution_iterative - expected_solution, np.inf) / np.linalg.norm(expected_solution, np.inf)

        # Imprimir os resultados
        print(f"\nN = {N}")
        print("Método Iterativo - Gauss-Seidel")
        print("Solucao encontrada:", solution_iterative)
        print("Condicionamento da matriz de coef.:", condition_number)
        print("Diferenca relativa em norma infinito:", relative_difference)

    except Exception as e:
        print(f"\nN = {N}")
        print("Erro no Método Iterativo - Gauss-Seidel:")
        print(e)
```

Para o código rodar, foram fornecidos os seguintes exemplos de entrada $\mathsf{A-N.dat}$ e $\mathsf{b-N.dat}$ para as dimensões N = 4, 6, 8 e 10.

$\mathsf{A-4.dat}$:

$$\begin{matrix} 1 & 2 & 3 & 4 \\ 2 & 3 & 4 & 5 \\ 3 & 4 & 5 & 6 \\ 4 & 5 & 6 & 7 \end{matrix}$$

$\mathsf{b-4.dat}$:

$$\begin{matrix} 20 \\ 30 \\ 40 \\ 50 \end{matrix}$$

$\mathsf{A-6.dat}$:

$$\begin{matrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 2 & 1 & 0 & 0 & 0 & 0 \\ 0 & 3 & 1 & 0 & 0 & 0 \\ 0 & 0 & 4 & 1 & 0 & 0 \\ 0 & 0 & 0 & 5 & 1 & 0 \\ 0 & 0 & 0 & 0 & 6 & 1 \end{matrix}$$

$\mathsf{b-6.dat}$:

$$\begin{matrix} 10 \\ 20 \\ 30 \\ 40 \\ 50 \\ 60 \end{matrix}$$

$\mathsf{A-8.dat}$:

$$\begin{matrix} 3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 2 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 2 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 2 & 1 \end{matrix}$$

$\mathsf{b-8.dat}$:

$$\begin{matrix} 5 \\ 4 \\ 3 \\ 2 \\ 1 \\ 1 \\ 2 \\ 3 \end{matrix}$$

$\mathsf{A-10.dat}$:

$$\begin{matrix} 5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 4 & 5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 4 & 5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 4 & 5 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 4 & 5 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 4 & 5 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 4 & 5 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 4 & 5 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 4 & 5 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 4 & 5 \end{matrix}$$

$\mathsf{b-10.dat}$:

$$\begin{matrix} 15 \\ 14 \\ 13 \\ 12 \\ 11 \\ 10 \\ 9 \\ 8 \\ 7 \\ 6 \end{matrix}$$

Outputs:

```
N = 4
Erro no Método Direto - Eliminação de Gauss:
O elemento pivô é zero. A eliminação gaussiana não pode continuar.

N = 6
Método Direto - Eliminação de Gauss (Fatoração LU)
Solução encontrada: [10. 0. 30. -80. 450. -2640.]
Condicionamento da matriz de coeficientes: 5214.169594463677
Diferença relativa em norma infinito: 241.0

N = 8
Método Direto - Eliminação de Gauss (Fatoração LU)
Solução encontrada: [1.66666667 0.66666667 1.66666667 -1.33333333 3.66666667 -6.33333333 14.66666667
-26.33333333]
Condicionamento da matriz de coeficientes: 361.95301288206036
Diferença relativa em norma infinito: 3.3939393939393923

N = 10
Método Direto - Eliminação de Gauss (Fatoração LU) Solução encontrada: [3. 0.4 2.28 0.576 1.7392
0.60864 1.313088 0.5495296 0.96037632 0.43169894]
Condicionamento da matriz de coeficientes: 6.297626959392478
Diferença relativa em norma infinito: 0.9636363636363636

N = 4
Erro no Método Iterativo - Gauss-Seidel:
Gauss-Seidel não convergiu após o número máximo de iterações.

N = 6
Método Iterativo - Gauss-Seidel
Solução encontrada: [10. 0. 30. -80. 450. -2640.]
Condicionamento da matriz de coeficientes: 5214.169594463677
Diferença relativa em norma infinito: 241.0

N = 8
Método Iterativo - Gauss-Seidel Solução encontrada: [1.66666667 0.66666667 1.66666667 -1.33333333
3.66666667 -6.33333333 14.66666667 -26.33333333]
Condicionamento da matriz de coeficientes: 361.95301288206036
Diferença relativa em norma infinito: 3.393939393939395

N = 10
Método Iterativo - Gauss-Seidel
Solução encontrada: [3. 0.4 2.28 0.576 1.7392 0.60864 1.313088 0.5495296 0.96037632 0.43169894]
Condicionamento da matriz de coeficientes: 6.297626959392478
Diferença relativa em norma infinito: 0.9636363636363636
```

OBS: O código no relatório ficou um pouco diferente do compilador online, por questões visuais. As direrenças estão principalmente nos *prints*.

E importante certificar-se de ter os arquivos ‘A-N.dat‘ e ‘b-N.dat‘ disponíveis no diretório de execução do código. Além disso, é necessário ter a biblioteca NumPy instalada para executar esse código.

O código acima apresenta uma implementação básica dos métodos direto (eliminação de Gauss) e iterativo (Gauss-Seidel) para resolver sistemas lineares. Note que é possível ajustar os parâmetros, como o número máximo de iterações e a tolerância, para obter resultados mais precisos ou melhorar a eficiência. Além disso, é importante observar que, dependendo do condicionamento da matriz A, o método iterativo pode exigir um número significativamente maior de iterações para convergir ou até mesmo não convergir. Os resultados obtidos podem variar dependendo das escolhas feitas durante a implementação.

Questão 3. Sejam A, B e C matrizes quadradas e não singulares de ordem n, e w, x e y vetores de dimensão n. Mostre como calcular w na equação abaixo sem que nenhum cálculo de inversão de matrizes seja necessário. Além disso, indique também como reduzir o custo computacional para o cálculo de w utilizando métodos de decomposição de matrizes.

$$w = A^{-1}(A^{-1} + B)x + A^{-1}(C + A^{-1})y$$

Resolução:

Para calcular o vetor w na equação dada sem realizar inversões diretas de matrizes, é possível utilizar a técnica de decomposição LU (decomposição de matrizes em matrizes triangulares inferior e superior) ou a técnica de decomposição de Cholesky (caso as matrizes sejam simétricas definidas positivas).

Para este caso, será utilizado a decomposição LU. Decompôe-se a matriz A em uma matriz triangular inferior L e uma matriz triangular superior U, de modo que $A = LU$. Então, a matriz A^{-1} pode ser calculada como $A^{-1} = U^{-1}L^{-1}$.

Primeiro, é preciso resolver a equação $w = A^{-1}(A^{-1} + B)x + A^{-1}(C + A^{-1})y$ em duas partes: a primeira parte envolvendo $A^{-1}(A^{-1} + B)x$ e a segunda parte envolvendo $A^{-1}(C + A^{-1})y$.

Parte 1: $A^{-1}(A^{-1}+B)x$. Substituindo A^{-1} por $U^{-1}L^{-1}$:

$$A^{-1}(A^{-1}+B)x = (U^{-1}L^{-1})(U^{-1}L^{-1}+B)x$$

Realizando algumas simplificações:

$$A^{-1}(A^{-1}+B)x = (U^{-1}L^{-1})(U^{-1}L^{-1}x+Bx)$$

$$A^{-1}(A^{-1}+B)x = U^{-1}(L^{-1}(U^{-1}L^{-1}x+Bx))$$

Definindo $z = L^{-1}x$, podendo ser calculado resolvendo o sistema triangular inferior $Lz = x$. Em seguida, define-se $p = U^{-1}z$, que pode ser calculado resolvendo o sistema triangular superior $Up = z$. Portanto, têm-se:

$$A^{-1}(A^{-1}+B)x = U^{-1}p$$

Parte 2: $A^{-1}(C+A^{-1})y$. Da mesma forma, substituindo A^{-1} por $U^{-1}L^{-1}$:

$$A^{-1}(C+A^{-1})y = (U^{-1}L^{-1})(C+U^{-1}L^{-1})y$$

Realizando as simplificações:

$$A^{-1}(C+A^{-1})y = (U^{-1}L^{-1})(C+L^{-1}(U^{-1}L^{-1})y)$$
$$A^{-1}(C+A^{-1})y = U^{-1}(L^{-1}(C+L^{-1}(U^{-1}L^{-1})y))$$

Definindo $v = L^{-1}y$, podendo ser calculado resolvendo o sistema triangular inferior $Lv = y$. Em seguida, define-se $q = U^{-1}v$, que pode ser calculado resolvendo o sistema triangular superior $Uq = v$. Portanto, têm-se:

$$A^{-1}(C+A^{-1})y = U^{-1}q$$

Agora, juntando as duas partes, é possível calcular w:

$$w = U^{-1}p + U^{-1}q$$

Esse método evita o cálculo direto de inversões de matrizes e, em vez disso, envolve resolver sistemas lineares triangulares inferiores e superiores, o que pode ser computacionalmente mais eficiente.

Lembra-se que, para aplicar essa abordagem, é preciso ter acesso à decomposição LU da matriz A. Se as matrizes forem simétricas definidas positivas, você também pode optar pela decomposição de Cholesky, que pode ser ainda mais eficiente nesses casos específicos.

Questão 4. Foi visto/usado durante o desenvolvimento do Método dos Gradientes Conjugados que $(r^{(k)})^T r^{(k-1)} = 0$, onde $r^{(k)}$ é o vetor resíduo no passo k e T representa a operação que gera a transposta. Mostre que isso é verdade.

Resolução:

Para entender o porquê $(r^{(k)})^T r^{(k-1)} = 0$ ser verdadeiro no Método dos Gradientes Conjugados, é preciso definir algumas propriedades desse método iterativo para resolver sistemas lineares.

No Método dos Gradientes Conjugados, busca-se encontrar a solução x para o sistema linear $Ax = b$, onde A é uma matriz simétrica e positiva definida. O algoritmo itera para melhorar a aproximação da solução x a cada passo k.

O vetor resíduo no passo k, denotado por $r^{(k)}$, é definido como o vetor que representa o resíduo atual após a iteração k, dado por:

$$r^{(k)} = b - Ax^{(k)}$$

onde $x^{(k)}$ é a aproximação atual da solução no passo k.

Agora, analisa-se a expressão $(r^{(k)})^T r^{(k-1)}$. Observa-se que o índice k no vetor resíduo $r^{(k)}$ indica o passo atual, enquanto o índice $k-1$ indica o passo anterior. Portanto, $(r^{(k)})^T r^{(k-1)}$ representa o produto interno entre o vetor resíduo atual e o vetor resíduo do passo anterior. Substituindo as definições de $r^{(k)}$ e $r^{(k-1)}$ na expressão, têm-se:

$$(r^{(k)})^T r^{(k-1)} = (b - Ax^{(k)})^T (b - Ax^{(k-1)})$$

Expandindo essa expressão:

$$(r^{(k)})^T r^{(k-1)} = (b^T - (Ax^{(k)})^T)(b - Ax^{(k-1)})$$

Nota-se que $(Ax^{(k)})^T = x^{(k)T} A^T$ (a transposta do produto é igual ao produto das transpostas na ordem inversa), então:

$$(r^{(k)})^T r^{(k-1)} = (b^T - x^{(k)T} A^T)(b - Ax^{(k-1)})$$

Usando a propriedade distributiva da multiplicação de matrizes:

$$(r^{(k)})^T r^{(k-1)} = b^T b - b^T A x^{(k-1)} - x^{(k)T} A^T b + x^{(k)T} A^T A x^{(k-1)}$$

Lembrando que A é simétrica ($A^T = A$), e como resultado $b^T A x^{(k-1)} = x^{(k)T} A x^{(k-1)}$:

$$(r^{(k)})^T r^{(k-1)} = b^T b - 2x^{(k)T} A x^{(k-1)} + x^{(k)T} A x^{(k-1)}$$

Finalmente, simplificando a expressão para:

$$(r^{(k)})^T r^{(k-1)} = b^T b - x^{(k)T} A x^{(k-1)}$$

Um ponto importante a ser ressaltar é que o Método dos Gradientes Conjugados, a direção de busca em cada iteração é escolhida de forma a ser conjugada com as direções de busca anteriores. Isso implica que $x^{(k)T} A x^{(k-1)} = 0$ para todo k, o que torna a expressão $(r^{(k)})^T r^{(k-1)} = 0$ válida.

Portanto, concluí-se que durante o desenvolvimento do Método dos Gradientes Conjugados, é verdade que $(r^{(k)})^T r^{(k-1)} = 0$, e isso está relacionado à propriedade de conjugação das direções de busca escolhidas pelo algoritmo. Essa propriedade é fundamental para o sucesso e eficiência do Método dos Gradientes Conjugados na resolução de sistemas lineares.

APÊNDICE A – Códigos da Atividade

Abaixo são apresentados os códigos realizados, todos desenvolvidos e testados na plataforma `https://www.onlinegdb.com/`. A **Questão 2** pode ser acessada através do link: `https://onlinegdb.com/wCX0Afk4h9`.