

Relatório da Lista 2

COC473 - Semestre 2020/PLE

Gabriel G. Milan¹

¹Departamento de Engenharia Eletrônica e de Computação – Universidade Federal do Rio de Janeiro (UFRJ)
21.941-909 – Rio de Janeiro – RJ – Brasil

1. Introdução

Para as implementações em código solicitadas nessa lista (e em todas as outras) foi criado um pacote Python chamado ”alc”(álgebra linear computacional). Esse pacote foi completamente escrito em Python e não está disponível no *Python Package Index* (ou PyPI). Em outras palavras, não pode ser instalado via *pip install alc*. No entanto, de forma a tornar possível replicar os resultados demonstrados nesse relatório, o código está disponibilizado abertamente em <https://github.com/gabriel-milan/linear-algebra>.

Nesse pacote, são implementadas todas as funcionalidades solicitadas (no caso da lista 2: *Power Method* e método de Jacobi para cálculo de autovalores e autovetores).

A fim de conseguir adquirir tais funcionalidades, também foram utilizadas todas as funcionalidades previamente implementadas e descritas no relatório anterior a esse.

Devido a todas essas funcionalidades serem implementadas do zero, o código tornou-se grande, fazendo-o inviável de ser explicitado por completo nesse relatório. De qualquer forma, as partes mais relevantes serão explicitadas, de forma a tornar compreensível a implementação.

Os nomes de variáveis e comentários espalhados ao longo do código foram escritos em inglês, de forma a, ao final dessa matéria, possibilitar a exposição desse trabalho como portfólio.

Ao final desse relatório, serão acrescentadas páginas escaneadas das resoluções manuais da lista e, portanto, os exercícios cuja resolução for estritamente manual (3 e 4) não serão citados em seções desse relatório.

Para os exercícios restantes, cada seção desse relatório fará referência a um deles.

2. Exercício 1

2.1. Implementação do *Power Method*

A implementação do *power method* foi realizada na função *power_method*, disponível em *alc.eigen*. Esse método é um método iterativo para calcular o maior autovalor e seu autovetor associado de uma matriz.

O algoritmo usado na implementação foi o fornecido em sala de aula, partindo de um vetor *x* preenchido com valores aleatórios (sendo $x_1 = 1$), e está codificado conforme abaixo:

```
from alc.constants import constants
from alc.utils import random_array
```

```

def power_method (arr, threshold=constants.epsilon):
    # Start with random array
    x = random_array((arr.shape[0], 1))
    # First element equals 1
    x[0][0] = 1
    # Iterations
    r = 1000
    lambda_i = constants.epsilon
    while (r > threshold):
        y = arr * x
        new_lambda_i = y[0][0]
        x = y / new_lambda_i
        r = abs(new_lambda_i - lambda_i) / abs(new_lambda_i)
        lambda_i = new_lambda_i
    return lambda_i, x

```

O valor de ϵ definido nas constantes do pacote é igual a 10^{-7} e é o valor padrão fornecido para o limiar θ , uma vez que a condição de convergência é que

$$R = \frac{|\lambda^i - \lambda^{i-1}|}{|\lambda^i|} \leq \theta$$

Um código de exemplo de uso dessa função para obter o maior autovalor de A , sendo A a matriz fornecida no exercício 3, e o seu autovetor associado é o seguinte:

```

import alc

A = alc.Array([
    [3, 2, 0],
    [2, 3, -1],
    [0, -1, 3],
])

print ("A = {}".format(A))

eigenvalue, eigenvector = alc.power_method(A)
print ("autovalor = {}".format(eigenvalue))
print ("autovetor = {}".format(eigenvector))

```

O resultado obtido após a execução desse código seria o seguinte:

```

A = alc.Array(
[3, 2, 0],
[2, 3, -1],
[0, -1, 3],
)
autovalor = 5.236067499566277
autovetor = alc.Array(
[1.0],
[1.1180339],
[-0.4999997],
)

```

3. Exercício 2

3.1. Implementação do Método de Jacobi para autovalores/autovetores

A implementação do método de Jacobi foi realizada na função `jacobi_eigen`, disponível em `alc.eigen`. Esse método é um método iterativo para calcular todos os autovalores e autovetores de uma matriz simétrica.

O algoritmo usado na implementação foi o fornecido em sala de aula, partindo de uma matriz identidade com mesmo formato que A e está codificado conforme abaixo:

```
from alc.constants import constants
from alc.utils import is_symmetric, eye, generate_p_matrix
from alc.utils import get_greater_value_outside_diagonal

def jacobi_eigen (arr, threshold=constants.epsilon):
    if not is_symmetric(arr):
        raise ValueError ("Input matrices for Jacobi method must be symmetric.")
    A = arr
    X = eye(arr.shape[0])
    greater, i, j = get_greater_value_outside_diagonal(A)
    while (greater > threshold):
        P = generate_p_matrix(A, i, j)
        A = P.t * A * P
        X = X * P
        greater, i, j = get_greater_value_outside_diagonal(A)
    return A, X
```

Devido ao uso recorrente das funções `get_greater_value_outside_diagonal`, que retorna o maior valor absoluto fora da diagonal de uma matriz A qualquer e seus respectivos índices i, j , e `generate_p_matrix`, que gera uma matriz P de rotação de forma que a matriz $A' = P^T AP$ teria os valores em $A'_{i,j}$ e $A'_{j,i}$ anulados, essas foram implementadas separadamente em `alc.utils`.

As implementações de ambas podem ser observadas abaixo:

```
import math
from alc.utils import eye

def get_greater_value_outside_diagonal (arr):
    greater = 0
    greater_i = 0
    greater_j = 0
    for i in range(arr.shape[0]):
        for j in range(arr.shape[1]):
            if (i != j):
                val = abs(arr[i][j])
                if val > greater:
                    greater = val
                    greater_i = i
                    greater_j = j
    return greater, greater_i, greater_j

def generate_p_matrix (arr, i, j):
```

```

if (arr[i][i] != arr[j][j]):
    phi = 1. / 2. * math.atan((2 * arr[i][j]) / (arr[i][i] - arr[j][j]))
else:
    phi = math.pi / 4
p = eye (arr.shape[0])
p[i][i] = math.cos(phi)
p[i][j] = -math.sin(phi)
p[j][i] = math.sin(phi)
p[j][j] = math.cos(phi)
return p

```

Com essas implementações, é possível computar todos os autovalores e autovetores da matriz A , fornecida para o exercício 3, com o seguinte código:

```

import alc

A = alc.Array([
    [3, 2, 0],
    [2, 3, -1],
    [0, -1, 3],
])

print ("A = {}".format(A))

eigenvalues, eigenvectors = alc.jacobi_eigen(A)
print ("autovalores = {}".format(eigenvalues))
print ("autovetores = {}".format(eigenvectors))

```

Dessa forma, o resultado obtido seria o seguinte:

```

A = alc.Array(
[3, 2, 0],
[2, 3, -1],
[0, -1, 3],
)
autovalores = alc.Array(
[5.236068, 0.0, 0.0],
[0.0, 0.763932, 0.0],
[0.0, 0.0, 3.0],
)
autovetores = alc.Array(
[0.6324555, -0.6324555, 0.4472136],
[0.7071068, 0.7071068, 0.0],
[-0.3162278, 0.3162278, 0.8944272],
)

```

4. Soluções manuais

A seguir, será anexado a esse relatório as soluções manuais escaneadas.

ÁLGEBRA LINEAR COMPUTACIONAL - COCH473
GABRIEL GAZOLA MILAN - DRE 116034377

$$\textcircled{3) } \quad A = \begin{bmatrix} 3 & 2 & 0 \\ 2 & 3 & -1 \\ 0 & -1 & 3 \end{bmatrix}, \quad B = \begin{bmatrix} 1 \\ -1 \\ -1 \end{bmatrix}$$

a) ATRAVÉS DO POLINÔMIO CARACTERÍSTICO, CALCULAR OS AUTOVALORES DE A:

$$A - \lambda I = \begin{bmatrix} 3-\lambda & 2 & 0 \\ 2 & 3-\lambda & -1 \\ 0 & -1 & 3-\lambda \end{bmatrix} \quad |A - \lambda I| = D(\lambda)$$

$$\Rightarrow = (3-\lambda)^3 - (3-\lambda)(-1)(-1) - (2.2.(3-\lambda))$$

$$= (3-\lambda)((3-\lambda)^2 - 1 - 4)$$

$$= (3-\lambda)(9-6\lambda+\lambda^2-5)$$

$$(3-\lambda)(\lambda^2-6\lambda+4)=0 \Rightarrow \begin{cases} 3-\lambda=0 \Rightarrow \lambda_1=3 \\ \lambda^2-6\lambda+4=0 \Rightarrow \begin{cases} \lambda_2=3-\sqrt{5} \\ \lambda_3=3+\sqrt{5} \end{cases} \end{cases}$$

• p/ $\lambda=3 \Rightarrow \begin{bmatrix} (3-3) & 2 & 0 \\ 2 & (3-3) & -1 \\ 0 & -1 & (3-3) \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \Rightarrow X = \begin{bmatrix} 1/2 \\ 0 \\ 1 \end{bmatrix} = V_1$

• p/ $\lambda=3-\sqrt{5} \Rightarrow \begin{bmatrix} \sqrt{5} & 2 & 0 \\ 2 & \sqrt{5} & -1 \\ 0 & -1 & \sqrt{5} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \Rightarrow X = \begin{bmatrix} -2 \\ \sqrt{5} \\ 1 \end{bmatrix} = V_2$

• p/ $\lambda=3+\sqrt{5} \Rightarrow \begin{bmatrix} -\sqrt{5} & 2 & 0 \\ 2 & -\sqrt{5} & -1 \\ 0 & -1 & -\sqrt{5} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \Rightarrow X = \begin{bmatrix} -2 \\ -\sqrt{5} \\ 1 \end{bmatrix} = V_3$

b) A É POSITIVA DEFINIDA?

\Rightarrow como todos os autovalores de A são positivos, é possível afirmar que A é positiva definida.

- c) Usar o power method?/ calcular o maior autovalor e o autovetor correspondente:
- Faz-se $X^0 = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$, calcula-se $Y_k = AX^k$, depois $Y'_k = Y_k / \|Y_k\|$ e $Y_k = \lambda^{(k+1)} X^k$,
 - calcular $R = \frac{\|X^k - X^{(k+1)}\|}{\|X^k\|}$, repetindo até que $R \leq \theta$.
 - Fazendo $\theta = 10^{-3}$:
- $Y^0 = \begin{bmatrix} 5 \\ 4 \\ 2 \end{bmatrix}, \lambda_1 = 5, X^1 = \begin{bmatrix} 1 \\ 0,8 \\ 0,4 \end{bmatrix}, R \approx 1; Y^1 = \begin{bmatrix} 4,6 \\ 4 \\ 0,9 \end{bmatrix}, \lambda_2 = 4,6, X^2 = \begin{bmatrix} 1 \\ 0,87 \\ 0,89 \end{bmatrix}, R \approx 0,09$
- $Y^2 = \begin{bmatrix} 4,74 \\ 4,52 \\ -0,61 \end{bmatrix}, \lambda_3 = 4,74, X^3 = \begin{bmatrix} 1 \\ 0,95 \\ -0,13 \end{bmatrix}, R \approx 0,03; Y^3 = \begin{bmatrix} 4,91 \\ 4,99 \\ -1,34 \end{bmatrix}, \lambda_4 = 4,99, X^4 = \begin{bmatrix} 1 \\ 1,02 \\ -0,22 \end{bmatrix}, R \approx 0,03$
- $Y^4 = \begin{bmatrix} 5,03 \\ 5,32 \\ -1,83 \end{bmatrix}, \lambda_5 = 5,03, X^5 = \begin{bmatrix} 1 \\ 1,06 \\ -0,36 \end{bmatrix}, R \approx 0,02; Y^5 = \begin{bmatrix} 5,12 \\ 5,54 \\ -2,15 \end{bmatrix}, \lambda_6 = 5,11, X^6 = \begin{bmatrix} 1 \\ 1,03 \\ -0,42 \end{bmatrix}, R \approx 0,02$
- $Y^6 = \begin{bmatrix} 5,17 \\ 5,67 \\ -2,34 \end{bmatrix}, \lambda_7 = 5,17, X^7 = \begin{bmatrix} 1 \\ 1,10 \\ -0,45 \end{bmatrix}, R \approx 0,01; Y^7 = \begin{bmatrix} 5,19 \\ 5,75 \\ -2,46 \end{bmatrix}, \lambda_8 = 5,19, X^8 = \begin{bmatrix} 1 \\ 1,11 \\ -0,44 \end{bmatrix}, R \approx 0,01$
- $Y^8 = \begin{bmatrix} 5,21 \\ 5,79 \\ -2,53 \end{bmatrix}, \lambda_9 = 5,21, X^9 = \begin{bmatrix} 1 \\ 1,11 \\ -0,49 \end{bmatrix}, R \approx 3 \cdot 10^{-3}; Y^9 = \begin{bmatrix} 5,22 \\ 5,82 \\ -2,57 \end{bmatrix}, \lambda_{10} = 5,23, X^{10} = \begin{bmatrix} 1 \\ 1,11 \\ -0,49 \end{bmatrix}, R \approx 2 \cdot 10^{-3}$
- $Y^{10} = \begin{bmatrix} 5,23 \\ 5,83 \\ -2,59 \end{bmatrix}, \lambda_{11} = 5,23, X_{11} = \begin{bmatrix} 1 \\ 1,12 \\ -0,49 \end{bmatrix}, R \approx 10^{-3}; Y^{11} = \begin{bmatrix} 5,23 \\ 5,84 \\ -2,60 \end{bmatrix}, \lambda_{12} = 5,23, X^{12} = \begin{bmatrix} 1 \\ 1,12 \\ -0,5 \end{bmatrix}, R \approx 6 \cdot 10^{-4}$

Encontra-se $\lambda = 5,23$ e $v = \begin{bmatrix} 1 \\ 1,12 \\ -0,5 \end{bmatrix}$

d) USANDO O MÉTODO DE JACOBI, OBTER 1000 DE PUNTUAÇÕES E FATORAR A DE A COM $\theta = 10^{-3}$.

* MATRIZ A É SIMÉTRICA!

COMEÇANDO COM $\vec{X} = \vec{I}$, ENCONTRAR O MAIOR VALOR ABSOLUTO FORA DA DIAGONAL DE A, SENDO ELE a_{ij} . COM ISSO, GERAR UMA MATRIZ $P = \begin{bmatrix} 1 & \dots & 0 \\ \vdots & \cos\phi & -\operatorname{sen}\phi \\ 0 & \operatorname{sen}\phi & \cos\phi \end{bmatrix}$

SENDO $p_{i,i} = \cos\phi = p_{ii}$; $p_{j,i} = -\operatorname{sen}\phi$; $p_{i,j} = \operatorname{sen}\phi$ E $\vec{P} = \begin{cases} \frac{1}{2}\operatorname{atan}\left(\frac{2a_{ij}}{a_{ii}-a_{jj}}\right) & a_{ij} \\ \pi/4 & a_{ii} = a_{jj} \end{cases}$

• CONSEGUIR $A_1 = A$, $X_1 = \vec{I}$. FAZER $A_{k+1} = P_k^T A_k P_k$ E $X_{k+1} = X_k P_k$ ATÉ QUE O MAIOR VALOR FORA DA DIAGONAL SEJA MENOR QUE θ .

$$P_1 = \begin{bmatrix} \cos\phi & -\operatorname{sen}\phi & 0 \\ \operatorname{sen}\phi & \cos\phi & 0 \\ 0 & 0 & 1 \end{bmatrix}, A_2 = \begin{bmatrix} 5 & 0 & -0,7 \\ 0 & 1 & -0,7 \\ -0,7 & -0,7 & 3 \end{bmatrix}, X_2 = \begin{bmatrix} 0,7 & -0,7 & 0 \\ 0,7 & 0,7 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$P_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & -\operatorname{sen}\phi \\ 0 & \operatorname{sen}\phi & \cos\phi \end{bmatrix}, A_3 = \begin{bmatrix} 5 & -0,2 & -0,7 \\ -0,2 & 0,8 & 0 \\ -0,7 & 0 & 3,2 \end{bmatrix}, X_3 = \begin{bmatrix} 0,7 & -0,7 & 0,2 \\ 0,7 & 0,7 & -0,2 \\ 0 & 0,3 & 0,3 \end{bmatrix}$$

$$P_3 = \begin{bmatrix} \cos\phi & 0 & -\operatorname{sen}\phi \\ 0 & 1 & 0 \\ \operatorname{sen}\phi & 0 & \cos\phi \end{bmatrix}, A_4 = \begin{bmatrix} 5,2 & -0,2 & 0 \\ -0,2 & 0,8 & -0,1 \\ 0 & -0,1 & 3 \end{bmatrix}, X_4 = \begin{bmatrix} 0,6 & -0,7 & 0,4 \\ 0,7 & 0,7 & 2 \cdot 10^{-2} \\ -0,3 & 0,3 & 0,9 \end{bmatrix}$$

$$P_4 = \begin{bmatrix} \cos\phi & -\operatorname{sen}\phi & 0 \\ \operatorname{sen}\phi & \cos\phi & 0 \\ 0 & 0 & 1 \end{bmatrix}, A_5 = \begin{bmatrix} 5,2 & 0 & 3 \cdot 10^{-3} \\ 0 & 0,8 & -0,1 \\ 3 \cdot 10^{-3} & -0,1 & 3 \end{bmatrix}, X_5 = \begin{bmatrix} 0,6 & -0,6 & 0,4 \\ 0,7 & 0,7 & 2 \cdot 10^{-2} \\ -0,3 & 0,3 & 0,9 \end{bmatrix}$$

$$P_5 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & -\operatorname{sen}\phi \\ 0 & \operatorname{sen}\phi & \cos\phi \end{bmatrix}, A_6 = \begin{bmatrix} 5,2 & 9 \cdot 10^{-3} & 3 \cdot 10^{-3} \\ 9 \cdot 10^{-5} & 0,8 & 0 \\ 3 \cdot 10^{-3} & 0 & 3 \end{bmatrix}, X_6 = \begin{bmatrix} 0,6 & -0,6 & 0,4 \\ 0,7 & 0,7 & 9 \cdot 10^{-4} \\ -0,3 & 0,3 & 0,9 \end{bmatrix}$$

$$P_6 = \begin{bmatrix} \cos\phi & 0 & -\operatorname{sen}\phi \\ 0 & 1 & 0 \\ \operatorname{sen}\phi & 0 & \cos\phi \end{bmatrix}, A_7 = \begin{bmatrix} 5,2 & 9 \cdot 10^{-3} & 0 \\ 9 \cdot 10^{-3} & 0,8 & -1 \cdot 10^{-3} \\ 0 & -1 \cdot 10^{-3} & 3 \end{bmatrix}, X_7 = \begin{bmatrix} 0,6 & -0,6 & 0,4 \\ 0,7 & 0,7 & 0 \\ 0,3 & 0,3 & 0,9 \end{bmatrix}$$

• APÓS ESSAS ITERAÇÕES, ENCONTRAMOS:

$$X \approx \begin{bmatrix} 5,2 & 0 & 0 \\ 0 & 0,8 & 0 \\ 0 & 0 & 3 \end{bmatrix} \quad \text{e} \quad V_R = \begin{bmatrix} 0,6 & -0,6 & 0,4 \\ 0,7 & 0,7 & 0 \\ -0,3 & 0,3 & 0,9 \end{bmatrix}$$

e) (*) CHOLEVSKY:

$$A = LL^T \Rightarrow L = \begin{bmatrix} 1,73 & 0 & 0 \\ 1,15 & 1,29 & 0 \\ 0 & -0,77 & 1,53 \end{bmatrix}$$

$$LL^T x = B \Rightarrow L^T x = y, Ly = B$$

$$y = \begin{bmatrix} 0,58 \\ -1,29 \\ 0 \end{bmatrix} \Rightarrow x = \begin{bmatrix} 1 \\ -1 \\ 0 \end{bmatrix}$$

(*) JACOBI: A É DIAGONAL DOMINANTE, CONVERGE

$$\rightarrow \text{COMEÇANDO COM } X_0 = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \text{ E } \theta = 10^{-3}$$

$$X_1 = \begin{bmatrix} -0,33 \\ -0,67 \\ +0,67 \end{bmatrix}, R_1 = 2,16 \quad X_0 = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \quad X_0 = \begin{bmatrix} 0,93 \\ -0,66 \\ 0,03 \end{bmatrix}, R_0 = 0,40$$

$$X_2 = \begin{bmatrix} +0,78 \\ +0,11 \\ +0,11 \end{bmatrix}, R_2 = 1,85 \quad X_2 = \begin{bmatrix} 0,77 \\ -0,94 \\ 0,11 \end{bmatrix}, R_2 = 0,28$$

$$X_3 = \begin{bmatrix} +0,26 \\ -0,81 \\ +0,37 \end{bmatrix}, R_3 = 1,17 \quad X_3 = \begin{bmatrix} 0,96 \\ -0,81 \\ 0,02 \end{bmatrix}, R_3 = 0,20$$

$$X_4 = \begin{bmatrix} +0,88 \\ -0,38 \\ +0,06 \end{bmatrix}, R_4 = 0,85 \quad X_4 = \begin{bmatrix} 0,87 \\ -0,96 \\ 0,06 \end{bmatrix}, R_4 = 0,14$$

$$X_5 = \begin{bmatrix} 0,59 \\ -0,90 \\ 0,21 \end{bmatrix}, R_5 = 0,56 \quad X_5 = \begin{bmatrix} 0,98 \\ -0,89 \\ 0,09 \end{bmatrix}, R_5 = 0,10$$

→ Após 10 iterações, foi verificar com a implementação computacional. Seriam necessárias 26 iterações para que $R \leq \epsilon$. E o resultado seria:

$$X = X_{25} = \begin{bmatrix} 1 \\ -1 \\ 9,6 \cdot 10^{-5} \end{bmatrix}$$

* Gauss-Seidel: A é simétrica positiva definida, converge

→ mesmos parâmetros do método de Jacobi

$$X_1 = \begin{bmatrix} -0,33 \\ 0,22 \\ 0,41 \end{bmatrix}, R_1 = 2,89 ; X_2 = \begin{bmatrix} 0,18 \\ -0,32 \\ 0,23 \end{bmatrix}, R_2 = 1,73 ; X_3 = \begin{bmatrix} 0,55 \\ -0,62 \\ 0,13 \end{bmatrix}, R_3 = 0,55$$

$$X_4 = \begin{bmatrix} 0,75 \\ -0,79 \\ 0,07 \end{bmatrix}, R_4 = 0,125 ; X_5 = \begin{bmatrix} 0,86 \\ -0,88 \\ 0,04 \end{bmatrix}, R_5 = 0,12 ; X_6 = \begin{bmatrix} 0,92 \\ -0,94 \\ 0,02 \end{bmatrix}, R_6 = 0,06$$

$$X_7 = \begin{bmatrix} 0,96 \\ -0,96 \\ 0,01 \end{bmatrix}, R_7 = 0,03 ; X_8 = \begin{bmatrix} 0,98 \\ -0,98 \\ 7 \cdot 10^{-3} \end{bmatrix}, R_8 = 0,02 ; X_9 = \begin{bmatrix} 0,99 \\ -0,99 \\ 4 \cdot 10^{-3} \end{bmatrix}, R_9 = 0,01$$

$$X_{10} = \begin{bmatrix} 0,99 \\ -0,99 \\ 2 \cdot 10^{-3} \end{bmatrix}, R_{10} = 5 \cdot 10^{-3} ; X_{11} = \begin{bmatrix} 1 \\ -1 \\ 1 \cdot 10^{-3} \end{bmatrix}, R_{11} = 3 \cdot 10^{-3} ; X_{12} = \begin{bmatrix} 1 \\ -1 \\ 6 \cdot 10^{-4} \end{bmatrix}, R_{12} = 2 \cdot 10^{-3}$$

$$X_{13} = \begin{bmatrix} 1 \\ -1 \\ 4 \cdot 10^{-4} \end{bmatrix}, R_{13} = 1 \cdot 10^{-3} \Rightarrow X = X_{13} = \begin{bmatrix} 1 \\ -1 \\ 4 \cdot 10^{-4} \end{bmatrix}$$

* Utilizando os autovalores e autovetores de A:

λ, θ são autovalores e autovetores de A, respectivamente

$$Y = \lambda^{-1} \theta^T B, X = \theta Y$$

$$Y = A^{-1} \theta^* B = \begin{bmatrix} +0,13 & 0 & 0 \\ 0 & +4,31 & 0 \\ 0 & 0 & +\frac{1}{3} \end{bmatrix} \begin{bmatrix} 0,63 & 0,71 & -0,32 \\ -0,63 & 0,71 & 0,22 \\ 0,45 & 0 & 0,89 \end{bmatrix} \begin{bmatrix} 1 \\ -1 \\ 1 \end{bmatrix}$$

$$Y = \begin{bmatrix} -0,07 \\ -1,34 \\ +0,45 \end{bmatrix} \quad X = \theta Y = \begin{bmatrix} 0,63 & -0,63 & +0,45 \\ -0,71 & +0,71 & 0 \\ -0,32 & +0,32 & +0,89 \end{bmatrix} \begin{bmatrix} -0,07 \\ -1,34 \\ +0,45 \end{bmatrix}$$

$$X = \begin{bmatrix} +1 \\ -1 \\ 0 \end{bmatrix}$$

1) Calcular $\det(A)$ usando os autovalores

$$|\det(A)| = \prod_{i=1}^n |\lambda_i| \Rightarrow \text{como } A \text{ é positiva definida,}$$

$$|\lambda_i| = \lambda_i \text{ e, portanto, } |\det(A)| = \det(A)$$

$$\det(A) = \prod_{i=1}^n \lambda_i = 3 \cdot (3-\sqrt{5}) \cdot (3+\sqrt{5}) = 3 \cdot (3^2 - (\sqrt{5})^2) = 3 \cdot (9-5) = 12$$

4) todos os resultados para os métodos computacionais coincidiram