24_1s_MTM1013T11_Q02_AndreGoncalvesdaSilva

April 22, 2024

MTM1013 - MÉT. NUM. ECOMPUTACIONAIS **Prof. Charles R. P. Szinvelski**

0.0.1 QUESTÃO (02): SISTEMAS LINEARES

24/1s - UFSM **Data: 22/04/24 (Das 7h30min às 8h30min)**

Acadêmico: André Gonçalves da Silva **Matrícula: 202210071**
Curso/Turma: 123/11

INSTRUÇÕES SOBRE A ELABORAÇÃO DOS ARQUIVOS A tarefa QUESTÃO (02) tem por finalidade a aplicação direta dos aspectos teóricos e práticos dos métodos numéricos e computacionais abordados na disciplina (Sistemas Lineares (Sistemas Triangulares e tratamento de matrizes) e recursos de programação em linguagem PYTHON 3 empregados nas respectivas implementaçãos). A produção do arquivo-resposta deverá seguir os moldes da produção dos arquivos respostas (formatos IPYNB e PDF e ambos deverão ser entregues via TAREFA aberta no MOODLE) para a QUESTÃO (02) de 22/04/2024.

OBSERVAÇÕES

- Salienta-se que observância da nomeação dos arquivos IPYNB e PDF (ver abaixo) e informações do cabeçalho (prencher devidamente) serão partes constituinte do escore (divisões sucessivas por 2):
  - aa_ss_MTMxxxxTtt_Q0x_NomeSobrenome.ipynb;
  - aa_ss_MTMxxxxTtt_Q0x_NomeSobrenome.pdf;
- Os algoritmos que devem ser empregados para as implementações em linguagem PYTHON 3 e *testes de mesas* são os algoritmos dados pelo material didático disponibilizado nos repositórios MEGA e DRIVE e AS RESOLUÇÕES NÃO PRODUZIDAS POR ESTES ALGORITMOS SERÃO DESCONSIDERADAS. Ressalta-se que os algoritmos do Material Didático são adequações dos algoritmos apresentados em Campos Filho (2018) (ou BURDEN E FAYRES(2016)[1]), e consequentemente estes algoritmos destas bibliografias poderão ser utilizados para a implementação;
- A elaboração das questões buscará caracterizações personalizadas para cada aluno, fato que resultará em um diversificado conjunto de situações-testes para a exploração dos aspectos teóricos e práticos da disciplina em detrimento, eventualmente, da aplicabilidade do tema abordado na questão.

[1] BURDEN, Richard L.; FAIRES, J D.; BURDEN, Annette M. Análise Numérica - Tradução da 10ª edição norte-americana. [Digite o Local da Editora]: Cengage Learning Brasil, 2016. E-book. ISBN 9788522123414. Disponível em: https://integrada.minhabiblioteca.com.br/#/books/9788522123414/. Acesso em: 18 abr. 2024.

OBTENÇÃO DE DADOS PARA ELABORAÇÃO DAS QUESTÕES Ao considerar o meu número de matrícula na UFSM, 3332882, monta-se a tabela

d1	d2	d3	d4	d5	d6	d7	d8	d9	d10
0	0	0	3	3	3	2	8	8	2

e sobre essas informações, colocam-se as seguintes elaborações:

(I) Com as informações da matrícula:

- $A = |d9 - d10| = |8 - 2| = 6$, e se $A = 0$, considere $A = 2$;
- $B = \left\lceil \frac{d1+...+d4}{A} \right\rceil = \left\lceil \frac{0+0+0+3}{6} \right\rceil = \lceil 0,5 \rceil = 1$;
- $C = \left\lfloor \frac{d5+d6+d7+d8}{A} \right\rfloor = \left\lfloor \frac{3+3+2+8}{6} \right\rfloor = \lfloor 2,67 \rfloor = 2$. Se $C = 0$ considere $C = 1$;
- $Dados = [A, B, C] = [6, 1, 2]$;

Considere as informações acima, matrícula e demais dados, e monte o seguinte sistema linear:

$$\begin{cases} (A)x + (d_1 + d_2)y + (d_2 + d_3)z = (B) \\ (d_3 + d_4)x + (B)y + (d_4 + d_5)z = (C) \\ (d_5 + d_6)x + (d_6 + d_7)y + (C)z = (A) \end{cases} \Leftrightarrow \begin{cases} 6x = 1 \\ 3x + y + 2z = 2 \\ 6x + 5y + 2z = 6 \end{cases}.$$

(Determinante via WolframAlpha.)

0.0.2 QUESTÕES.

1) (0,5 ponto) Faça o que se pede, a partir das intruções acima:

1.1) (0,25 ponto [Dados]) Informe $Matricula = [d1, d2, d3, d4, d5, d6, d7, d8, d9, d10]$ e $Dados = [A, B, C]$) como listas;

1.2) (0,25 ponto [Dados]) Informe o Sistema Linear, na forma acordada, associado ao tratamento dos dados;

2) (1,0 ponto) A partir de 1.2), faça o que se pede:

2.1) (0,2 ponto) Calcule os menores principais de A ($|A_i|,\ i = 1, 2$);

2.2) (0,4 ponto) Monte a decomposição LU da matriz dos coeficientes ($A = LU$);

2.3) (0,2 ponto) Informe o Sistema Linear $Ly = b$ na forma de lista e na forma de matriz ampliada acordada e resolva-o pela implementação desenvolvida em aula;

2.4) (0,2 ponto) Informe o Sistema Linear $Ux = y$ na forma Informe o Sistema Linear $Ly = b$ na forma de lista e na forma de matriz ampliada acordada e resolva-o pela implementação desenvolvida em aula;

OBSERVAÇÃO: Colocar os documentos IPYNB e PDF da resolução no espaço TAREFA do MOODLE aberta, caso queira entregar o arquivo-resposta com esses recursos computacionais. Caso se resolva via Google COLAB, colocar link compartilhado sem restrições de acesso e também o arquivo PDF associado.

---

[3]:
```
import numpy as np
from math import *
```

1 Código implementado (código geral, não específico do problema)

[23]:
```
def elimination(A, b):
    # junta a matriz A e o vetor b como uma coluna
    M = np.concatenate((A, b), axis=1)

    # para cada coluna k de 0 a n-1 (que no caso n-2), pois vamos transformar em tringular superior
    # trabalhando coluna por coluna, primeiro zerando os elementos abaixo da diagonal principal
    # na primeira coluna, depois na segunda, mas a ultima coluna não precisa ser zerada pois não
    # existem elementos abaixo da diagonal principal
    for k in range(0, M.shape[0]-1):
        # para cada linha i de k+1 a n (que no caso n-1), começa em k+1 pois zeramos somente abaixo da diagonal principal
        for i in range(k+1, M.shape[0]):
            M[i] = M[i] - M[k] * M[i, k] / M[k, k]

    # retorna a matriz A e o vetor b separadamente
    return np.split(M, [M.shape[1]-1], axis=1)

def STS(A, b):
    n = len(b)
    x = np.zeros(n)

    # i vai variar de n-1 a 0 (por isso o n-1 no primeiro argumento, -1 porque python vai ir até 0, e o último
    # -1 diz que deve decrementar -1 de i a cada iteração), o resto será o que está a direita do x em questão,
    # por isso j vai de i+1 a n (que no caso sera n-1)
    for i in range(n-1, -1, -1):
```

```
        rest = sum([A[i, j] * x[j] for j in range(i+1, n)])
        # o .item() é para o numpy parar de reclamar
        x[i] = (b[i] - rest).item() / A[i,i]

    return x

def STI(A, b):
    n = len(b)
    x = np.zeros(n)

    # i vai variar de 0 a n-1, o resto será o que está a esquerda do x em questão
    # logo j vai de 0 a i-1 (é python, então tudo começa em 0 e vai até um valor antes do final)
    for i in range(n):
        rest = sum([A[i, j] * x[j] for j in range(i)])
        x[i] = (b[i] - rest).item() / A[i,i]

    return x

def LUdecomp(A):
    n = A.shape[0]
    L = np.zeros((n, n))
    U = np.zeros((n, n))

    for i in range(n):
        for j in range(i, n):
            U[i, j] = A[i, j] - sum([L[i, k] * U[k, j] for k in range(i)])

        # queria poder usar i aqui, mas com j fica melhor no código, para U o i significa a linha e j a coluna,
        # para L o i significa a coluna e o j linha
        for j in range(i+1, n):
            L[j, i] = (A[j, i] - sum([L[j, k] * U[k, i] for k in range(i)]))/U[i,i]

    L = np.identity(n) + L

    return L, U

def solveLU(A, b):
    L, U = LUdecomp(A)

    y = STI(L, b)
    x = STS(U, y)

    return x
```

```
def formatough(A):
    for i in range(A.shape[0]):
        if len(A.shape) == 2:
            for j in range(A.shape[1]):
                print("[{0:^9}]".format("%.3g" % A[i,j]), end='')
        else:
            print("[{0:^9}]".format("%.3g" % A[i]), end='')

        print()

def printlist(M):
    lista = []
    for i in range(M.shape[0]):
        lista.append(M[i].tolist())

    print(lista)
```

2 Confecção dos dados e matriz (não é o gabarito ainda!).

2.1 Devido ao determinante da matriz original ser 0 eu troquei o elemento m_{32} por 0

[9]:
```
d1 = 0
d2 = 2
d3 = 0
d4 = 2
d5 = 2
d6 = 1
d7 = 0
d8 = 0
d9 = 7
d10 = 1

A = abs(d9 - d10)
if A == 0:
    A = 2

B = ceil((d1 + d2 + d3 + d4)/A)

C = floor((d5 + d6 + d7 + d8)/A)
if C == 0:
    C = 1

m11 = A
m12 = d1 + d2
```

```
m13 = d2 + d3
m21 = d3 + d4
m22 = B
m23 = d4 + d5
m31 = d5 + d6
# AVISO: troquei essa entrada da matriz por um zero para evitar um determinante␣
 ↪nulo da matriz
m32 = 0
m33 = C
b1 = B
b2 = C
b3 = A

M = np.array([m11, m12, m13, m21, m22, m23, m31, m32, m33]).reshape(3,3)
b = np.array([b1, b2, b3]).reshape(3,1)

print("Matriz M dos coeficientes")
formatough(M)

print()
print("Vetor b")
formatough(b)
```

```
Matriz M dos coeficientes
[    6    ][    2    ][    2    ]
[    2    ][    1    ][    4    ]
[    3    ][    0    ][    1    ]

Vetor b
[    1    ]
[    1    ]
[    6    ]
```

Extraímos então o sistema de equações lineares $M\vec{x} = \vec{b}$ (troquei por M porque no código já existe um A):

$$\begin{aligned} 6x + 2y + 2z &= 1 \\ 2x + y + 4z &= 1 \\ 3x + z &= 6 \end{aligned} \tag{1}$$

[13]:
```
Mdet = M[0,0]*(M[1,1]*M[2,2] - M[1,2]*M[2,1]) - M[0,1]*(M[1,0]*M[2,2] -␣
 ↪M[1,2]*M[2,0]) + M[0,2]*(M[1,0]*M[2,1] - M[1,1]*M[2,0])
Mminor1det = M[0,0]
Mminor2det = M[0,0]*M[1,1] - M[0,1]*M[1,0]

print("|M|: ", Mdet)
```

```
print("|M_1|: ", Mminor1det)
print("|M_2|: ", Mminor2det)
```

```
|M|:  20
|M_1|:  6
|M_2|:  2
```

[17]:
```
L, U = LUdecomp(M)

print("Matriz L:")
formatough(L)

print()
print("Matriz U:")
formatough(U)

print()
print("L * U: ")
# NOTA: deu um 10^(-17) no m_32, mas é basicamente 0
formatough(L @ U)
```

```
Matriz L:
[    1    ][    0    ][    0    ]
[  0.333  ][    1    ][    0    ]
[   0.5   ][   -3    ][    1    ]

Matriz U:
[    6    ][    2    ][    2    ]
[    0    ][  0.333  ][  3.33   ]
[    0    ][    0    ][   10    ]

L * U:
[    6    ][    2    ][    2    ]
[    2    ][    1    ][    4    ]
[    3    ][ 3.7e-17 ][    1    ]
```

[28]:
```
Lamp = np.concatenate((L, b), axis=1)
print("L b = y")
print("Como lista de listas: ")
printlist(Lamp)
print("Como matriz ampliada padrão: ")
formatough(Lamp)
print()

y = STI(L, b).reshape(3,1)
Uamp = np.concatenate((U, y), axis=1)
print("U y = x")
```

```
printlist(Uamp)
formatough(Uamp)
print()
```

```
L b = y
Como lista de listas:
[[1.0, 0.0, 0.0, 1.0], [0.3333333333333333, 1.0, 0.0, 1.0], [0.5,
-2.9999999999999996, 1.0, 6.0]]
Como matriz ampliada padrão:
[    1    ][    0    ][    0    ][    1    ]
[  0.333  ][    1    ][    0    ][    1    ]
[   0.5   ][   -3    ][    1    ][    6    ]

U y = x
[[6.0, 2.0, 2.0, 1.0], [0.0, 0.33333333333333337, 3.3333333333333335,
0.6666666666666667], [0.0, 0.0, 9.999999999999998, 7.5]]
[    6    ][    2    ][    2    ][    1    ]
[    0    ][  0.333  ][  3.33   ][  0.667  ]
[    0    ][    0    ][   10    ][   7.5   ]
```

[32]:
```
x = solveLU(M, b)
print("Finalmente a solução do sistema: ")
print(x)
```

```
Finalmente a solução do sistema:
[ 1.75 -5.5   0.75]
```

3 Gabarito

3.1 1

3.1.1 1.1

[34]:
```
print("Matrícula: %s"%([d1, d2, d3, d4, d5, d6, d7, d8, d9, d10]))
print("Dados: %s"%([A, B, C]))
```

```
Matrícula: [0, 2, 0, 2, 2, 1, 0, 0, 7, 1]
Dados: [6, 1, 1]
```

3.1.2 1.2

troquei m_{32} por 0 para não dar determinante nulo

$$\begin{aligned} 6x + 2y + 2z &= 1 \\ 2x + y + 4z &= 1 \\ 3x + z &= 6 \end{aligned} \tag{2}$$

[36]:
```
Mamp = np.concatenate((M, b), axis=1)
print("Sistema linear: ")
formatough(Mamp)
```

```
Sistema linear:
[    6    ][    2    ][    2    ][    1    ]
[    2    ][    1    ][    4    ][    1    ]
[    3    ][    0    ][    1    ][    6    ]
```

3.2 2

3.2.1 2.1

[35]:
```
print("Menor principal |M_1|: %s"%Mminor1det)
print("Menor principal |M_2|: %s"%Mminor2det)
```

```
Menor principal |M_1|: 6
Menor principal |M_2|: 2
```

3.2.2 2.2

[38]:
```
print("M = ")
formatough(L)
print("*")
formatough(U)
```

```
M =
[    1    ][    0    ][    0    ]
[  0.333  ][    1    ][    0    ]
[   0.5   ][   -3    ][    1    ]
*
[    6    ][    2    ][    2    ]
[    0    ][  0.333  ][  3.33   ]
[    0    ][    0    ][   10    ]
```

3.2.3 2.3

[39]:
```
Lamp = np.concatenate((L, b), axis=1)
print("Sistema linear L b = y")
print()
print("Como lista de listas: ")
printlist(Lamp)
print()
print("Como matriz ampliada padrão: ")
formatough(Lamp)
print()
```

```
Sistema linear L b = y
```

```
Como lista de listas:
[[1.0, 0.0, 0.0, 1.0], [0.3333333333333333, 1.0, 0.0, 1.0], [0.5,
-2.9999999999999996, 1.0, 6.0]]

Como matriz ampliada padrão:
[    1    ][    0    ][    0    ][    1    ]
[  0.333  ][    1    ][    0    ][    1    ]
[   0.5   ][   -3    ][    1    ][    6    ]
```

3.2.4 2.4

[42]:
```
y = STI(L, b).reshape(3,1)
Uamp = np.concatenate((U, y), axis=1)
print("Sistema linear U y = x")
print()
print("Como lista de listas: ")
printlist(Uamp)
print()
print("Como matriz ampliada padrão: ")
formatough(Uamp)
print()
```

```
Sistema linear U y = x

Como lista de listas:
[[6.0, 2.0, 2.0, 1.0], [0.0, 0.33333333333333337, 3.3333333333333335,
0.6666666666666667], [0.0, 0.0, 9.999999999999998, 7.5]]

Como matriz ampliada padrão:
[    6    ][    2    ][    2    ][    1    ]
[    0    ][  0.333  ][  3.33   ][  0.667  ]
[    0    ][    0    ][   10    ][   7.5   ]
```

[46]:
```
print("Solução do sistema M x = b: ")
print()
formatough(solveLU(M, b))
```

```
Solução do sistema M x = b:

[  1.75   ]
[  -5.5   ]
[  0.75   ]
```