Gabarito da Lista de Exercícios 1 de Analise Numérica

Prof.: Fabrício Murai

Informações importantes:

- Data de entrega: até 23:59 do dia 23/03/2018.
- Questões podem ser discutidas entre até três alunos. Nomes dos colegas precisam ser listados. Contudo, a escrita das soluções e submissão deve ser feita individualmente.
- Submissão deve ser feita em formato PDF através do Moodle, mesmo que tenham sido resolvidas a mão e escaneadas.
- Todas as soluções devem ser justificadas.
- Se puder, peço por favor que marque o tempo gasto para resolver a lista, para que o tamanho da lista de exercícios seja ajustado em semestres futuros.

1. Considere o seguinte computador hipotético, onde a representação de um número real qualquer, em ponto flutuante, pode ser generalizado da forma $F(2, 3, -7, 7)$.

   (a) Como será representado o número $(-11.9)_{10}$ neste computador?

   $$-(1011)_2 + \epsilon = -(0.1011)_2 \times 2^4 + \epsilon$$

   Como o tamanho máximo da mantissa é 3, teremos:

-	1	0	1	+	4

   (b) O número $(1.1)_{10} \times 2^{-9}$ pode ser representado? Por que?
   Sim. O menor número que pode ser representado nesse sistema é $(0.001)_2 \times 2^{-7} = 2^{-9}$. Note que embora ele não possa ser representado de forma exata, ele não gera um "erro".

2. Considere o polinômio

   $$p(x) = (x-2)^9 = x^9 - 18x^8 + 144x^7 - 672x^6 + 2016x^5 - 4032x^4 + 5376x^3 - 4608x^2 + 2304x - 512.$$

   a. Plote $p(x)$ para $x = 1.920, 1.921, 1.922, \ldots, 2.080$ calculando p através dos seus coeficientes $1, , -18, 144, \ldots$, isto é, usando a expansão. Dica: o método `linspace` do numpy vai ser útil para isso.

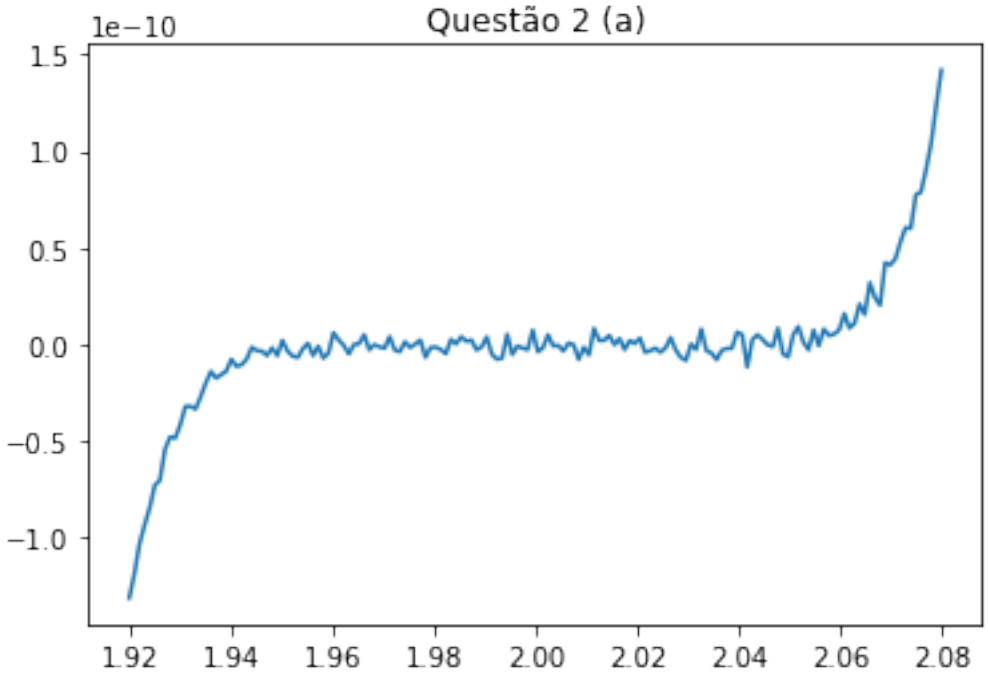
```
import numpy as np
import matplotlib.pyplot as plt

start = 1.920
end = 2.080
step = 0.001
num = (end - start)/step

c = [1, -18, 144, -672, 2016, -4032, 5376, -4608, 2304, -512]

x = np.linspace(start, end, num)
y = np.polyval(c, x)
```

```
plt.plot(x, y)
plt.title('Questao 2 (a)')
```



b. Plote o mesmo gráfico novamente, agora calculando p através da expressão $(x-2)^9$.

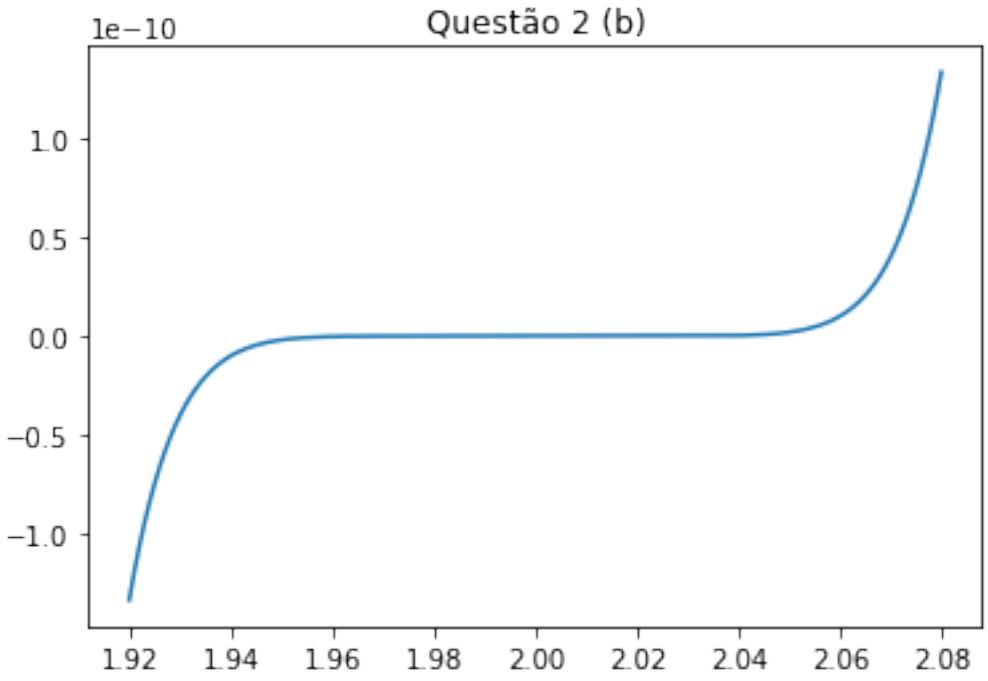
```
import numpy as np
import matplotlib.pyplot as plt

start = 1.920
end = 2.080
step = 0.001
num = (end - start)/step

def f(x): return (x-2)**9

x = np.linspace(start, end, num)
y = f(x)

plt.plot(x, y)
plt.title('Questao 2 (b)')
```



c. Explique a diferença.

Solução. Em 2.a, para avaliar o polinômio é necessário realizar um grande número de operações (adições, multiplicações, potenciações e subtrações), o que acarreta em uma grande propagação de erros. Em 2.b, foi realizado apenas uma subtração e uma multiplicação para cada ponto, o que gerou um erro menor.

3. Quantos números diferentes existem usando precisão-dupla (`float` de 64 bits)? Escreva sua resposta usando potências de 2.

   Solução. Um `float` de 64 bits, também conhecido como `double` tem sua representação computacional especificada pelo padrão IEEE 754 da seguinte maneira:

   - Bit de sinal: 1 bit
   - Expoente: 11 bits
   - Mantissa: 52 bits

   Assim, $2^1 * 2^{11} * 2^{52} = 2^{64}$ valores reais. Porém:

   - Há duas representações possíveis para o número 0, uma com sinal e outra sem sinal.
   - Há dois valores infinitos, `+infinito` e `-infinito`, quando a mantissa é tudo zero e o expoente é todo 1.
   - Também há valores `NaN` quando o expoente é todo 1 e a mantissa tem dígitos não zero.

   Assim, temos $2 * (2^{11} - 1) * 2^{52} - 1$ números distintos.

4. Considere a matriz

$$A = \begin{bmatrix} 2 & -1 & 4 \\ 3 & 5 & 19 \end{bmatrix}$$

   (a) Mostre que suas colunas são linearmente dependentes.
   Solução. Para isto, basta mostrar que existem coeficientes $(a_1, a_2, a_3) \neq (0, 0, 0)$ que satisfazem

$$a_1 \begin{bmatrix} 2 \\ 3 \end{bmatrix} + a_2 \begin{bmatrix} -1 \\ 5 \end{bmatrix} + a_3 \begin{bmatrix} 4 \\ 19 \end{bmatrix} = 0.$$

Solução 1. De fato, $(a_1, a_2, a_3) = (3, 2, -1)$ satisfaz a equação.

Solução 2. (Essa solução não requer que você adivinhe os coeficientes.) Observa-se que a solução requer que todos os coeficientes sejam não-nulos, caso contrário, um dos vetores seria "múltiplo" de outro. Logo, se as colunas são linearmente dependentes, temos que

$$b_1 \begin{bmatrix} 2 \\ 3 \end{bmatrix} + b_2 \begin{bmatrix} -1 \\ 5 \end{bmatrix} = \begin{bmatrix} 4 \\ 19 \end{bmatrix}.$$

onde $b_1 = -a_1/a_3$ e $b_2 = -a_2/a_3$. Resolvendo esse sistema linear de duas equações e duas variáveis, obtem-se $b_1 = 3$ e $b_2 = 2$.

Solução 3. Usando escalonamento de colunas, podemos zerar a última coluna. Isto mostra que as colunas são linearmente dependentes.

(b) Qual o posto da matriz A?

Solução 1. Removendo-se uma das colunas, permanecemos com duas colunas que não são linearmente dependentes (ou, equivalentemente, múltiplas uma da outra). Logo, o posto é 2.

Solução 2. O posto de linhas é sempre igual ao posto de colunas. Como as linhas são linearmente independentes, conclui-se que o posto é 2.

5. Se $D = BC$, onde

$$B = \begin{bmatrix} 1 & 4 & 7 \\ 2 & 5 & 8 \\ 3 & 6 & 9 \end{bmatrix} \quad \text{e} \quad C = \begin{bmatrix} -1 & 0 & 3 \\ -2 & 0 & 2 \\ -3 & 0 & 1 \end{bmatrix},$$

é correto afirmar que a matriz D possui inversa? Por quê?

Solução 1. Se D possui inversa, então $D^{-1} = C^{-1}B^{-1}$. No entanto, não existe nenhuma matriz $C^{-1}B^{-1}$, pois a matriz B é singular e, portanto, não admite inversa. Isso porque $\det(B) = 0$ quando uma das linhas ou colunas é nula (pode ser provado usando a definição recursiva de determinante).

Solução 2. (Baseada em propriedade sobre os determinantes não vista em sala). Sabe-se que

$$D = BC \Rightarrow \det(D) = \det(B)\det(C)$$

Como $\det(B) = 0$, D é singular e não admite inversa.

6. Considere a matriz

$$A = \begin{bmatrix} 1 & 1 \\ 1 & 3 \end{bmatrix}$$

(a) Escreva o polinômio característico desta matriz.

$$det(A - \lambda I) = \det\left(\begin{bmatrix} 1-\lambda & 1 \\ 1 & 3-\lambda \end{bmatrix}\right)$$

$$\begin{aligned} (1-\lambda)(3-\lambda) - 1 &= 0 \\ \lambda^2 - 4\lambda + 2 &= 0 \end{aligned}$$

(b) Encontre os autovalores de A.

Resolvendo-se a equação do segundo grau acima, temos $\lambda_1 = 2 + \sqrt{2}$ e $\lambda_2 = 2 - \sqrt{2}$.

7. Considere o sistema de equações lineares dado por

$$\begin{bmatrix} 4 & 0 & 0 \\ -2 & 5 & 0 \\ 1 & 7 & 3 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 12 \\ 4 \\ 20 \end{bmatrix}$$

(a) Sem resolver o sistema, determine se este possui uma única solução.
Seja A a matriz de coeficientes. Como $\det(A) = 4 \times 5 \times 3 \neq 0$, conclui-se que o sistema possui uma única solução.

(b) Resolva utilizando o método das substituições sucessivas.

$$\begin{aligned} x_1 &= 12/4 = 3 \\ x_2 &= \frac{4 - [(-2) \times 3]}{5} = 2 \\ x_3 &= \frac{20 - [1 \times 3 + 7 \times 2]}{3} = 1 \end{aligned}$$

8. Sejam o sistema linear $Ax = b$, de ordem n, e a matriz C de ordem n e não singular. Assinale V antes da sentença se ela for verdadeira e F se for falsa e justifique:

( ) A matriz CA não é singular.
Falso. Se A for singular, $\det(CA) = \det(C)\det(A) = 0$, ou seja, CA pode ser singular.

( ) Se C for uma matriz de permutação, então $\det(CA) = \det(A)$.
Falso. Se C for uma matriz de permutação, então $\det(CA) = \det(A)$.
O correto é $\det(CA) = (-1)^t \det(A)$, onde t é o número de trocas de linhas necessártias para se transformar C na matriz identidade.

( ) O sistema $Ax = b$ não é necessariamente equivalente ao sistema $CAx = Cb$.
Falso. O sistema $Ax = b$ não é necessariamente equivalente ao sistema $CAx = Cb$.
Se C não é singular, sempre pode ser multiplicado dos dois lados da equação.

9. Seja $A = \begin{bmatrix} 3 & 2 & 4 \\ 1 & 1 & 2 \\ 4 & 3 & -2 \end{bmatrix}$.

(a) Calcule a decomposição (manualmente) $PA = LU$ **Não precisa resolver** $Ly = Pb$**, nem** $Ux = y$**.**

(b) Calcule o determinante de A a partir de P, L e U.

L	Multiplicadores	A	Operações	p
1	$m_{11} = 0.75$	$3 \quad 2 \quad 4$		1
2	$m_{21} = 0.25$	$1 \quad 1 \quad 2$		2
3		$\underline{4} \quad 3 \quad -2$		$\underline{3}$
4		$0 \quad \underline{-0.25} \quad 5.5$	$L_1 - 0.75L_3$	$\underline{1}$
5	$m_{22} = -1$	$0 \quad 0.25 \quad 2.5$	$L_2 - 0.25L_3$	2
6		$0 \quad 0 \quad 8$	$L_5 + L_4$	$\underline{2}$

$$L = \begin{bmatrix} 1 & 0 & 0 \\ 0.75 & 1 & 0 \\ 0.25 & -1 & 1 \end{bmatrix} \text{ e } U = \begin{bmatrix} 4 & 3 & -2 \\ 0 & -0.25 & 5.5 \\ 0 & 0 & 8 \end{bmatrix}.$$

10. Modifique o método LU (sem pivotação) abaixo para que funcione <u>in-place</u>. Isto é, a implementação não deve alocar memória nova para L ou U, mas sim sobrescrever A.

```
def LU(A):
    # U = np.copy(A)
    m, n = A.shape
    # L = np.eye(n)
```

```
    for k in range(n-1):
        for j in range(k+1,n):
            A[j,k] = A[j,k]/A[k,k]          # L[j,k] = U[j,k]/U[k,k]
            A[j,k:n] -= A[j,k] * A[k,k:n] # U[j,k:n] -= L[j,k] * U[k,k:n]
    return L, U
```

11. Modifique o método LU visto em sala para incluir pivotação. Dica: o método `swap` será útil.

```
def swap(a,b):
    temp = np.copy(a)
    a[:] = b
    b[:] = temp
```

Solução.

```
def LUPivot(A) :
    U = np.copy(A)
    m, n = A.shape
    L = np.eye (n)
    for k in range (n-1):
        maxLine = np.argmax( abs(U[:,k]) )
        swap( U[k,:], U[maxLine,:] )
        for j in range (k+1, n) :
            L[j, k] = U[j, k] / U[k, k]
            U[j, k:n] -= L[j, k] * U[k, k:n]
            swap( L[k,:], L[maxLine,:] )
    return L, U
```