Algebra Linear Computacional - Lista de Exercicios 1

Lucas Resende Pellegrinelli Machado (2018126673)

March 16, 2019

Exercício 1.
Visto que a matriz que dita a transição entre os estados depois de 1 ano é

$$E_{+1} = \begin{bmatrix} 0.90 & 0.07 & 0.02 & 0.01 \\ 0.00 & 0.93 & 0.05 & 0.02 \\ 0.00 & 0.00 & 0.85 & 0.15 \\ 0.00 & 0.00 & 0.00 & 1.00 \end{bmatrix}$$

e temos que o estado atual da população é

$$E_{atual} = \begin{bmatrix} 0.85 & 0.10 & 0.05 & 0.00 \end{bmatrix}$$

Para calcular o estado da população daqui a 10 anos, temos apenas que calcular

$$E_{10anos} = E_{atual} \times (E_{+1})^{10}$$

Que com Python e a biblioteca Numpy pode ser calculada usando:

```python
import numpy as np

# Matriz de transicao
a = np.array([[0.9, 0.07, 0.02, 0.01],
              [0.0, 0.93, 0.05, 0.02],
              [0.0, 0.0,  0.85, 0.15],
              [0.0, 0.0,  0.0,  1.0]])

# Configuracao atual da populacao
b = np.array([0.85, 0.1, 0.05, 0.0])

# Calcula b x a^10
r = b @ np.linalg.matrix_power(a, 10)
print(r)
```

Que nos dá como resposta

$$E_{10anos} = \begin{bmatrix} 0.29637667 & 0.3167509 & 0.1342175 & 0.25265492 \end{bmatrix}$$

Logo temos que a população após 10 anos é composta por aproximadamente 29.6% assintomáticos, 31.7% sintomáticos, 13.4% com AIDS e 25.3% mortos.

Exercício 2.
De acordo com o enunciado, o número tem 5 bits de mantissa e expoentes estão no intervalo $[6, -7] \in \mathbb{N}$.

a. O número tem que ser positivo, logo o primeiro bit tem de ser 0 para isso. Agora, temos que escolher também a menor combinação entre mantissa e expoente possível. Como os expoentes estão no intervalo $[6, -7] \in \mathbb{N}$, para que o número seja o menor possível, precisamos da menor mantissa possível elevada à -6. Para a menor mantissa, temos apenas que colocar os 5 bits como 0. Dessa forma temos:

$$Primeirobit = 0 = (-1)^0$$
$$Mantissa = (1.00000)_2$$
$$Expoente = -6$$

Assim:

$$(-1)^0 \cdot (1.00000)_2 \cdot 2^{-6} = 2^{-6}$$

b. O problema é bem similar com a letra (a). As únicas diferenças é que queremos a maior mantissa possível elevada ao maior expoente possível. Como os expoentes estão no intervalo $[6, -7] \in \mathbb{N}$, o maior possível é 7. Para a mantissa é só colocar os 5 bits como 1. Assim temos:

$$Primeirobit = 0 = (-1)^0$$
$$Mantissa = (1.11111)_2$$
$$Expoente = 7$$

Assim:

$$(-1)^0 \cdot (1.11111)_2 \cdot 2^7 \approx 2^8$$

c. Da mesma forma que transformamos números em notação científica em base 10 (no estilo $x \times 10^y$) para números explicitos movendo as vírgulas para a direita até que exista um número 1 à esquerda da virgula, faremos o mesmo com números em base 2.

$$2^3 \cdot (0.01010)_2 \implies 2^2 \cdot (0.10100)_2 \implies 2^1 \cdot (1.01000)$$

Assim, depois de adicionar o primeiro bit que será 0 visto que o número é positivo, temos que a resposta é:

$$(-1)^0 \cdot (1.01000)_2 \cdot 2^1$$

d. Esse exercício é analogo ao anterior, porém como se trata de expoentes negativos, movemos a virgula para a esquerda até que só exista um dígito à esquerda da virgula. Outra diferença é que o número dessa vez é negativo, logo o primeiro bit será 1.

$$-2^{-2} \cdot (1101.01010)_2 \implies -2^{-3} \cdot (110.10101)_2 \implies -2^{-4} \cdot (11.01010)_2 \implies -2^{-5} \cdot (1.10101)_2$$

Um detalhe importante é que a medida que fazemos isso, perdemos informações de bits que são empurrados além da capacidade de armazenamento para a direita.

Mas direto à resposta, adicionando o bit de sinal:

$$(-1)^1 \cdot 2^{-5} \cdot (1.10101)_2$$

e. Como o número é maior que 1, ele é positivo, logo o primeiro bit é 0. Para garantir que o número será maior que 1, diremos que o expoente é 0 e que a mantissa tem que ser maior que 0. Dessa forma, quando fizermos $expoente \cdot mantissa$ teremos um resultado maior que 1 (visto que com o $expoente = 0$, teremos $2^0 = 1$).

O desafio agora é encontrar a menor mantissa possível maior que $(1.00000)_2$, que levando em conta que os digitos significativos representam números maiores a medida que você olha bits mais à esquerda, se torna fácil perceber que a menor mantissa possível maior que essa é uma com um 1 no bit menos significativo, ou seja, $(1.00001)_2$.

Dessa forma, com o sinal sendo positivo, expoente sendo 0 e a mantissa sendo $(1.00001)_2$, temos que a resposta é:

$$(-1)^0 \cdot (1.00001)_2 \cdot 2^0 \approx 1 + 2^{-5}$$

f. Como o $\epsilon_{machine}$ é a maior precisão que nossa máquina consegue atingir, um jeito fácil de verificar qual é esse valor é ver qual a metade da diferença entre 1 e o menor número possível de se representar na máquina maior que 1.

Na questão (e) descobrimos que tal número é $(-1)^0 \cdot (1.00001)_2 \cdot 2^0 \approx 1 + 2^{-5}$, logo a diferença para 1 é 2^{-5} e a metade disso é 2^{-6}, sendo essa a resposta.

Exercício 3.
Primeiro temos que

$$A_{m\times n} = U_{m\times m}\Sigma_{m\times m}V^T_{n\times n}$$

Logo, usando a propriedade que $(AB)^T = B^T A^T$

$$A^T_{n\times m} = V_{n\times n}\Sigma^T_{m\times m}U^T_{m\times m}$$

1. Temos que:

$$AA^T = U\Sigma V^T V\Sigma^T U^T$$

Usando que como V é ortogonal, $V^T V = I$

$$AA^T = U\Sigma\Sigma^T U^T$$

Usando que Σ é uma matriz diagonal retangular, temos que $\Sigma\Sigma^T = \Sigma^2$

$$AA^T = U\Sigma^2U^T$$

2. Da mesma forma que:

$$A^TA = V\Sigma^TU^TU\Sigma V^T$$

Usando que como U é ortogonal, $U^TU = I$

$$A^TA = V\Sigma^T\Sigma V^T$$

Usando que Σ é uma matriz diagonal retangular, temos que $\Sigma^T\Sigma = \Sigma^2$

$$AA^T = V\Sigma^2V^T$$

Exercício 4.
Primeiro temos que mostrar que U e V são ortonormais, ou seja, todas as colunas são unitárias e o produto escalar entre todas é 0.

- Testando se U é ortonormal:

$$C_1 = \begin{bmatrix} -0.632 & 0.316 & -0.316 & 0.632 \end{bmatrix}$$
$$C_2 = \begin{bmatrix} 0.000 & -0.707 & -0.707 & 0.000 \end{bmatrix}$$

$$C_1 \cdot C_2 = (0.316)(-0.707) + (-0.316)(-0.707) = 0$$
$$|C_1| = \sqrt{(-0.632)^2 + (0.316)^2 + (-0.316)^2 + (0.632)^2} = 1$$
$$|C_2| = \sqrt{(0.000)^2 + (-0.707)^2 + (-0.707)^2 + (0.000)^2} = 1$$

Como $C_1 \cdot C_2 = 0$, $|C_1| = 1$ e $|C_2| = 1$, então U é ortonormal.

- Testando se V é ortonormal:

$$C_1 = \begin{bmatrix} -0.707 & 0.707 \end{bmatrix}$$
$$C_2 = \begin{bmatrix} -0.707 & -0.707 \end{bmatrix}$$

$$C_1 \cdot C_2 = (-0.707)(-0.707) + (0.707)(-0.707) = 0$$
$$|C_1| = \sqrt{(-0.707)^2 + (0.707)^2} = 1$$
$$|C_2| = \sqrt{(-0.707)^2 + (-0.707)^2} = 1$$

Como $C_1 \cdot C_2 = 0$, $|C_1| = 1$ e $|C_2| = 1$, então V é ortonormal.

O que resta mostrar é que de fato

$$C \approx U\Sigma V^T$$

O que é facil mostrar com o seguinte script em Python

```
import numpy as np

U = np.array([[-0.632, 0.000],
       [0.316, -0.707],
       [-0.316, -0.707],
       [0.632, 0.0]])

Sig = np.array([[2.236, 0.0],
       [0.0, 1.0]])

Vt = np.array([[-0.707, 0.707],
       [-0.707, -0.707]])

C = U @ Sig @ Vt
print(C)
```

Que gera o resultado (valores arredendados para 4 casas decimais):

$$\begin{bmatrix} 0.9991 & -0.9991 \\ 0.0003 & 0.9994 \\ 0.9994 & 0.0003 \\ -0.9994 & 0.9994 \end{bmatrix} \approx \begin{bmatrix} 1 & -1 \\ 0 & 1 \\ 1 & 0 \\ -1 & 1 \end{bmatrix}$$

O que é correto visto que a representação em SVD de uma matriz será uma aproximação da mesma, logo com os valores com 3 casas decimais indicados pelo exercício temos um pequeno erro em relação a matriz desejada (na casa de 10^{-4}).

Exercício 5.
Temos que achar um vetor $v = (x, y)$ tal que com

$$C^* = \begin{bmatrix} 1 & -1 \\ 0 & 1 \\ 1 & 0 \\ -1 & 1 \\ 1 & 1 \end{bmatrix} \text{ e } U^* = \begin{bmatrix} -0.632 & 0.000 \\ 0.316 & -0.707 \\ -0.316 & -0.707 \\ 0.632 & 0.000 \\ x & y \end{bmatrix}$$

Temos

$$C^* = U^*\Sigma V^T$$

Resolvendo $U^* \times \Sigma$:

$$U^* \times \Sigma = \begin{bmatrix} (-0.632 \cdot 2.236 + 0.0 \cdot 0.0) & (-0.632 \cdot 0.0 + 0.0 \cdot 1.0) \\ (0.316 \cdot 2.236 - 0.707 \cdot 0.0) & (0.316 \cdot 0.0 - 0.707 \cdot 1.0) \\ (-0.316 \cdot 2.236 - 0.707 \cdot 0.0) & (-0.316 \cdot 0.0 - 0.707 \cdot 1.0) \\ (0.632 \cdot 2.236 + 0.0 \cdot 0.0) & (0.632 \cdot 0.0 + 0.0 \cdot 1.0) \\ (x \cdot 2.236 + y \cdot 0.0) & (x \cdot 0.0 + y \cdot 1.0) \end{bmatrix} = \begin{bmatrix} -1.4131 & 0 \\ 0.7065 & -0.707 \\ -0.7065 & -0.707 \\ 1.4131 & 0 \\ x \cdot 2.236 & y \end{bmatrix}$$

E agora resolvendo $(U^* \times \Sigma) \times V^T$

$$(U^* \times \Sigma) \times V^T = \begin{bmatrix} -1.4131 \cdot -0.707 + 0.0 \cdot -0.707 & -1.4131 \cdot 0.707 + 0.0 \cdot -0.707 \\ 0.7065 \cdot -0.707 + -0.707 \cdot -0.707 & 0.7065 \cdot 0.707 - 0.707 \cdot -0.707 \\ -0.7065 \cdot -0.707 + -0.707 \cdot -0.707 & -0.7065 \cdot 0.707 - 0.707 \cdot -0.707 \\ 1.4131 \cdot -0.707 + 0 \cdot -0.707 & 1.4131 \cdot 0.707 + 0.0 \cdot -0.707 \\ x \cdot 2.236 \cdot -0.707 + y \cdot -0.707 & x \cdot 2.236 \cdot 0.707 + y \cdot -0.707 \end{bmatrix} =$$

$$= \begin{bmatrix} 0.9991 & -0.9991 \\ -0.4994 + 0.4998 & 0.4994 + 0.4998 \\ 0.4994 + 0.4999 & -0.4994 + 0.4998 \\ -0.9991 & 0.9991 \\ -1.5808x - 0.707y & 1.5808x - 0.707y \end{bmatrix} = \begin{bmatrix} 0.9991 & -0.9991 \\ 0.0004 & 0.9992 \\ 0.9992 & 0.0004 \\ -0.9991 & 0.9991 \\ -1.5808x - 0.707y & 1.5808x - 0.707y \end{bmatrix}$$

Assim, voltando ao problema que o novo usuário deu nota positiva aos dois sites, temos que:

$$\begin{bmatrix} -1.5808x - 0.707y & 1.5808x - 0.707y \end{bmatrix} = \begin{bmatrix} 1 & 1 \end{bmatrix}$$

Ou seja, temos o sistema:

$$\begin{cases} -1.5808x - 0.707y = 1 \\ 1.5808x - 0.707y = 1 \end{cases}$$

E resolvendo o sistema temos

$$\begin{cases} x = 0 \\ y = -1.4144 \end{cases}$$

Logo temos que:

$$U^* = \begin{bmatrix} -0.632 & 0.000 \\ 0.316 & -0.707 \\ -0.316 & -0.707 \\ 0.632 & 0.000 \\ 0 & -1.4144 \end{bmatrix}$$

E testando a nova matriz na fórmula $C^* = U^* \Sigma V^T$, temos que:

$$\begin{bmatrix} 0.9991 & -0.9991 \\ 0.0003 & 0.9994 \\ 0.9994 & 0.0003 \\ -0.9994 & 0.9994 \\ 0.9999 & 0.9999 \end{bmatrix} \approx \begin{bmatrix} 1 & -1 \\ 0 & 1 \\ 1 & 0 \\ -1 & 1 \\ 1 & 1 \end{bmatrix}$$

Exatamente como queriamos