Problemas em Equipe 11

Estudantes: Eduardo Eiji goto, Gustavo Hammerschmidt, João Vitor Andrioli de Souza.

1) Paulo está de bom humor (BH), mais ou menos (MM) ou de mau humor (MH). Se ele está BH hoje, então estará BH, MM ou MH amanhã, com as seguintes probabilidades: 0,5, 0,4, 0,1. Se ele está MM hoje, então estará BH, MM e MH amanhã com as seguintes probabilidades: 0,3, 0,4, 0,3. se ele está MH hoje, então ele estará BH, MM, e MH com as seguintes probabilidades: 0,2, 03, 0,5. O humor do Paulo pode ser modelado por uma CMTD com a seguinte matriz de transição:

$$P = \begin{vmatrix} 0,5 & 0,4 & 0,1 \\ 0,3 & 0,4 & 0,3 \\ 0,2 & 0,3 & 0,5 \end{vmatrix}$$

   a) Calcule a probabilidade de Paulo estar de mau humor hoje e ficar de humor mais ou menos daqui a 3 dias
   Dica: A probabilidade desejada estará será uma transição de mau humor (estado 3) para mais ou menos (estado 2) em 3 dias. Ou seja, o valor da probabilidade pode ser encontrado terceira linha da segunda coluna da matriz P elevada à terceira potência.
   Copie seu código aqui:

   Python:

```
import numpy as np

P = [[0.5, 0.4, 0.1],
     [0.3, 0.4, 0.3],
     [0.2, 0.3, 0.5] ]

def multipleOfP(matrix, times):

    aux = matrix

    for i in range(times-1):

        aux = np.dot(aux, matrix)

    return aux

print("P_21 ^ 3 = ", multipleOfP(P, 3)[2][1] )
```

   Output:

```
P_21 ^ 3 =  0.364
```

b) Calcule a matriz A, o vetor B e o vetor PI do regime permanente do humor do Paulo.
Dica 1: usar o notebook CMTD.ypynb
Dica 2: Equações do regime permanente
$\pi_1 = 0{,}5\pi_1 + 0{,}3\pi_2 + 0{,}2\pi_3$
$\pi_2 = 0{,}4\pi_1 + 0{,}4\pi_2 + 0{,}3\pi_3$
$\pi_3 = 0{,}1\pi_1 + 0{,}3\pi_2 + 0{,}5\pi_3$
$\pi_1 + \pi_2 + \pi_3 = 1$

Copie aqui o seu código:

Python:

```
import numpy as np

P = [[0.5, 0.4, 0.1],
     [0.3, 0.4, 0.3],
     [0.2, 0.3, 0.5] ]

P = np.array(P)

def cmtdP(P):
    # Testar se a matriz eh quadrada
    # e se a soma de todas as linhas eh igual a 1
    [r,c] = P.shape
    if ((r != c) | np.all(np.sum(P, 1) != 1)):
        raise Exception('Matriz P invalida!')

    # Calcular a matriz A
    A = np.transpose(P) - np.identity(r)
    A = np.vstack((A, [1 for _ in range(r)]) )

    print("A:\n",A,"\n")

    # Calcular o vetor B
    #    B = vetor de 0s de tamanho r (usar np.zeros)
    #    Concatenar o vetor B com o array [1] (usar np.hstack)
    B = np.hstack((np.zeros(r), [1]))

    print("B:",B,"\n")
    # Calcular o vetor PI
    PI = np.dot( np.linalg.pinv(A), B)

    return PI

print("PI: ",cmtdP(P),"\n")
```

Ouput:

A:
[[-0.5 0.3 0.2]
[0.4 -0.6 0.3]
[0.1 0.3 -0.5]
[1.   1.   1.]]

B: [0. 0. 0. 1.]

PI: [0.33870968 0.37096774 0.29032258]

---

c) Qual a proporção do tempo que o Paulo passa de bom humor?

PI[0] = 0.33870968

2) Implementar a função cmtdP para calcular o estado permanente de uma cadeia de Markov em tempo discreto.
A função recebe como argumento a matriz de probabilidades de um passo
Testa se a matriz está corretamente construída em relação às probabilidades (probabilidades de uma linha tem que somar 1)
Constrói as matrizes A e B
Retorna o vetor PI
Dica: Usar cmtd.py

Já implementado na questão acima:

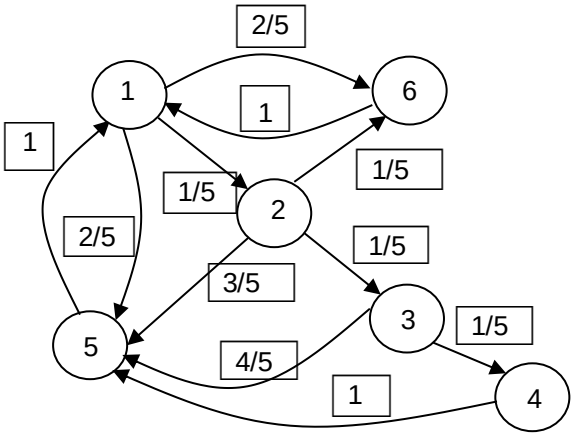
```
import numpy as np

P = np.array([[0.5, 0.4, 0.1], [0.3, 0.4, 0.3], [0.2, 0.3, 0.5]])

def cmtdP(P):

    [r,c] = P.shape
    if ((r != c) | np.all(np.sum(P, 1) != 1)):
        raise Exception('Matriz P invalida!')

    A = np.vstack(( (np.transpose(P) - np.identity(r)), [1 for _ in range(r)] ))

    B = np.hstack((np.zeros(r), [1]))

    print("A:\n", A, "\n\nB:", B, "\n")

    PI = np.dot( np.linalg.pinv(A), B)

    return PI

print("PI: ",cmtdP(P),"\n")
```

---

3) Pedro e Natália formam o casal perfeito, com apenas um probleminha: quem lava a louça hoje? Na maior parte das vezes ambos são voluntários, mas de vez em quando a louça fica para o dia seguinte. A Natália observou que os fatos ocorrem, mais precisamente, da seguinte maneira: (i) Quando não há louça acumulada, Natália e Pedro se apresentam na mesma proporção, mas em uma a cada cinco vezes, a louça fica para o dia seguinte. (ii) Quando a louça está um dia acumulada, Natália se apresenta três vezes mais do que Pedro, mas em uma a cada cinco vezes, a louça fica para o dia seguinte. (iii) Quando a louça está dois dias acumulada, apenas Natália se apresenta, mas em uma a cada cinco vezes, a louça fica para o dia seguinte. (iv) Quando a louça está três dias acumuladas, a Natália sempre se apresenta.

a) Modele o problema como uma CMTD e apresente o diagrama de transições.
Dica1: utilize os seguintes estados:
Estado 1 – louça lavada Estado 2 – 1 dia de atraso
Estado 3 – 2 dias de atraso Estado 4 – 3 dias de atraso
Estado 5 – Natália lava a louça Estado 6 – Pedro lava a louça

Dica 2: Complete as probabilidades no seguinte diagrama de transições:



b) Calcule a matriz de transições P.
Copie seu código aqui:

```
P = [ [0.0, 0.2, 0.0, 0.0, 0.4, 0.4],
      [0.0, 0.0, 0.2, 0.0, 0.6, 0.2],
      [0.0, 0.0, 0.0, 0.2, 0.8, 0.0],
      [0.0, 0.0, 0.0, 0.0, 1.0, 0.0],
      [1.0, 0.0, 0.0, 0.0, 0.0, 0.0],
      [1.0, 0.0, 0.0, 0.0, 0.0, 0.0] ]
```

c) Calcule o vetor PI do regime permanente.
Copie seu código aqui:

PI: [0.44483986 0.08896797 0.01779359 0.00355872 0.24911032 0.19572954]

d) Qual a proporção dos dias que a Natália e o Paulo lavam louça?
Natália:
Paulo: 0.19572954 (para conferir)

Natália = PI[4] = 0.24911032

Código usado na questão 3:

```
import numpy as np

P = [ [0.0, 0.2, 0.0, 0.0, 0.4, 0.4],
   [0.0, 0.0, 0.2, 0.0, 0.6, 0.2],
   [0.0, 0.0, 0.0, 0.2, 0.8, 0.0],
   [0.0, 0.0, 0.0, 0.0, 1.0, 0.0],
   [1.0, 0.0, 0.0, 0.0, 0.0, 0.0],
   [1.0, 0.0, 0.0, 0.0, 0.0, 0.0] ]

P = np.array(P)
```

```
def cmtdP(P):

    [r,c] = P.shape
    if ((r != c) | np.all(np.sum(P, 1) != 1)):
        raise Exception('Matriz P invalida!')

    A = np.vstack(( (np.transpose(P) - np.identity(r)), [1 for _ in range(r)] ))

    B = np.hstack((np.zeros(r), [1]))

    print("A:\n", A, "\n\nB:", B, "\n")

    PI = np.dot( np.linalg.pinv(A), B)

    return PI

print("PI: ",cmtdP(P),"\n")
```

Output:

```
A:
 [[-1.  0.  0.  0.  1.  1. ]
 [ 0.2 -1.  0.  0.  0.  0. ]
 [ 0.  0.2 -1.  0.  0.  0. ]
 [ 0.  0.  0.2 -1.  0.  0. ]
 [ 0.4  0.6  0.8  1.  -1.  0. ]
 [ 0.4  0.2  0.  0.  0.  -1. ]
 [ 1.  1.  1.  1.  1.  1. ]]

B: [0. 0. 0. 0. 0. 0. 1.]

PI:  [0.44483986 0.08896797 0.01779359 0.00355872 0.24911032 0.19572954]
```
