**Problemas em equipe 12**

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

1. Um ateliê comporta duas máquinas idênticas, cada uma com taxa de falha de 0,25 falhas por dia. Quando uma máquina falha ela precisa passar por dois reparadores (r1 e r2), necessariamente nessa ordem, ou seja, primeiro pelo reparador r1 e depois pelo reparador r2. A taxa de reparação do reparador r1 é 0,5 reparações por dia e a taxa de reparação do reparador r2 é 0,75 reparações por dia. Os estados do sistema são representados por uma tripla de variáveis (n, r1, r2), onde n é a quantidade de máquinas em reparação, ou seja, n = 0, 1 ou 2; r1é igual a zero se o reparador 1 não está ocupado e igual a 1 se o reparador 1 está ocupado; r2é igual a zero se o reparador 2 não está ocupado e igual a 1 se o reparador 2 está ocupado.

A cadeia de Markov que representa o processo de funcionamento do ateliê terá então 6 estados:

Estado 1: (0, 0, 0) = nenhuma máquina com defeito e nenhum reparador trabalhando;

Estado 2: (1, 1, 0) = uma máquina com defeito e reparador 1 trabalhando;

Estado 3: (1, 0, 1) = uma máquina com defeito e reparador 2 trabalhando;

Estado 4: (2, 1, 0) = duas máquinas com defeito e reparador 1 trabalhando;

Estado 5: (2, 1, 1) = duas máquinas com defeito e reparadores 1 e 2 trabalhando;

Estado 6: (2, 0, 1) = duas máquinas com defeito e reparador 2 trabalhando (a segunda máquina que falhou está aguardando o reparador 2 que ainda trabalha na primeira máquina que falhou).

O diagrama de transições do funcionamento e reparo das máquinas pode ser representado como segue:



Calcular a matriz Q

TranscricaoGrafo = np.array([[0.0, 0.25, 0.0, 0.0, 0.0, 0.0],

[0.0, 0.0, 0.5, 0.25, 0.0, 0.0],

[0.75, 0.0, 0.0, 0.0, 0.25, 0.0],

[0.0, 0.0, 0.0, 0.0, 0.5, 0.0],

[0.0, 0.75, 0.0, 0.0, 0.0, 0.5],

[0.0, 0.0, 0.75, 0.0, 0.0, 0.0]], dtype=np.float64)

Q = np.array([[0.0, 1.0, 0.0, 0.0, 0.0, 0.0],

[0.0, 0.0, 0.6667, 0.3333, 0.0, 0.0],

[0.75, 0.0, 0.0, 0.0, 0.25, 0.0],

[0.0, 0.0, 0.0, 0.0, 1.0, 0.0],

[0.0, 0.6, 0.0, 0.0, 0.0, 0.4],

[0.0, 0.0, 1.0, 0.0, 0.0, 0.0]], dtype=np.float64)

1. Implementar a função cmtcP para calcular o estado permanente de uma cadeia de Markov em tempo contínuo.

A função recebe como argumento a matriz Q

Testa se a matriz está corretamente construída em relação (todas as linhas têm que somar 1)

Constrói as matrizes A e B

Retorna o vetor PI

Usar cmtc.py

Problema já resolvido

def cmtcP(Q):

[r,c] = Q.shape

if ((r != c) | np.any(np.sum(Q, 1) != 1)):

raise Exception('Matriz P invalida!')

A = np.transpose(Q)

A = np.vstack((A, np.ones(r)))

B = np.zeros(r)

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

A\_pinv = np.linalg.pinv(A)

PI = np.dot(A\_pinv, B)

return PI

1. Calcular o vetor de probabilidades do regime permanente.

Utilize PE12.ipynb

Copie seu código e resultado aqui

import numpy as np

Q = np.array([[0.0, 1.0, 0.0, 0.0, 0.0, 0.0],

[0.0, 0.0, 0.6667, 0.3333, 0.0, 0.0],

[0.75, 0.0, 0.0, 0.0, 0.25, 0.0],

[0.0, 0.0, 0.0, 0.0, 1.0, 0.0],

[0.0, 0.6, 0.0, 0.0, 0.0, 0.4],

[0.0, 0.0, 1.0, 0.0, 0.0, 0.0]], dtype=np.float64)

def cmtcP(Q):

[r,c] = Q.shape

if ((r != c) | np.any(np.sum(Q, 1) != 1)):

raise Exception('Matriz P invalida!')

A = np.transpose(Q)

A = np.vstack((A, np.ones(r)))

B = np.zeros(r)

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

A\_pinv = np.linalg.pinv(A)

PI = np.dot(A\_pinv, B)

return PI

print("PI: ", cmtcP(Q))

Output:

PI: [-0.07142857 0.42861429 0.19047619 0.0952381 0.35714286 -0.1429 ]

Obs.: seu código continha uma instrução errada. O correto é np.any(np.sum(Q, 1) != 1).