E04 - ANN

Vinicius Gasparini

17 de Setembro de 2019

1 Jacobi - Implementação

```
from pprint import pprint
from numpy import array, diag, diagflat, dot, linalg

def jacobi(A,b,N,x):
    D_ = diagflat(diag(A))
    L_U  = A - D_
    D_ = linalg.inv(D_)

    for i in range(N-1):
        x = dot(dot(-D_,L_U) , x)+dot(D_,b)
        print('X(%d) = '%(i+1),end='')
        print(x)
    return x

A = array([  [ 7.6 ,-0.2 ,-0.6 , 2   ,-1.3 , 1.2 , 2.3 ],
             [-2.4 , 18  ,-0.9 , 2.7 , 2.1 ,-2   , 0.4],
             [ 0.3 , 0.9 ,16.8 ,-1.9 , 1.3 , 2.2 ,-1.5],
             [-2   ,-1.2 , 2.4 ,11.8 ,-1.6 ,-1.6 ,-1.1],
             [-1.8 ,-2.5 , 2.4 ,-1.4 ,13.1 , 1.7 ,-2.7],
             [-2.2 , 1.8 , 1.9 ,-0.6 ,-0.1 , 10  , 1.3],
             [-1.5 ,-2   , 2.5 , 1.8 , 0.7 , 1.2 , 12.1]])

b = array([-0.8,-1.6,-4.6,0.8,0.8,4.6,4.0])
chute = array([-4.9,1.8,3.8,-2.8,-2.6,0.2,-2])

sol = jacobi(A,b,N=10,x=chute)
```

2 Resposta

```
A: array([[ 7.6, -0.2, -0.6,  2. , -1.3,  1.2,  2.3],
          [-2.4, 18. , -0.9,  2.7,  2.1, -2. ,  0.4],
          [ 0.3,  0.9, 16.8, -1.9,  1.3,  2.2, -1.5],
          [-2. , -1.2,  2.4, 11.8, -1.6, -1.6, -1.1],
          [-1.8, -2.5,  2.4, -1.4, 13.1,  1.7, -2.7],
          [-2.2,  1.8,  1.9, -0.6, -0.1, 10. ,  1.3],
          [-1.5, -2. ,  2.5,  1.8,  0.7,  1.2, 12.1]])

b: [-0.8 -1.6 -4.6  0.8  0.8  4.6  4. ]

x: [-0.30957065 -0.13108813 -0.28340396
     0.15146424  0.05709781  0.44505971
     0.25836421]
```

Pontando, a resposta correta é o ***item c***