Sesión 1 - Matrices distribuidas

June 20, 2021

1 CURSO MATRICES DISTRIBUIDAS 2021

2 Sesión 1 - Fecha: 13 de Junio 2021

2.1 Inversa de una matriz para resolver sistemas de ecuaciones

[4]:
```
import numpy as np
from scipy import linalg

A = np.matrix([[1,1],[42,30]])
b = np.matrix([[500],[18600]])

print("Solución: \n ", linalg.inv(A)*b)
```

```
Solución:
  [[300.]
 [200.]]
```

2.2 Cadenas de Markov

[9]:
```
import numpy as np

def multiplicaN(A,B,n):
    for i in range(n):
        C = A*B
        B = C
    return C


if __name__ == '__main__':
    A = np.matrix([[0.5, 0.1, 0.1, 0.05, 0.2],[0.2, 0.3, 0.2, 0.15, 0.1],
                   [0.15, 0.2, 0.3, 0.3, 0.1],[0.1, 0.3, 0.2, 0.4, 0.1],
                   [0.05, 0.1, 0.2, 0.1, 0.5]])
    b = np.matrix([[10],[30],[40],[10],[10]])
    x = multiplicaN(A,b,10)
```

```
    print(A)
    print(x)
    print(6000/100*x)
    print(np.round(6000/10000*x))
```

```
[[0.5  0.1  0.1  0.05 0.2 ]
 [0.2  0.3  0.2  0.15 0.1 ]
 [0.15 0.2  0.3  0.3  0.1 ]
 [0.1  0.3  0.2  0.4  0.1 ]
 [0.05 0.1  0.2  0.1  0.5 ]]
[[17.89996204]
 [18.86929887]
 [21.67034545]
 [22.77249271]
 [18.78790093]]
[[1073.99772234]
 [1132.15793223]
 [1300.22072697]
 [1366.3495628 ]
 [1127.27405567]]
[[11.]
 [11.]
 [13.]
 [14.]
 [11.]]
```

2.3 Codificación de mensajes con la Inversa

[1]:
```
from scipy import linalg
import numpy as np

## ------------ Codificar mensaje ------------
def codificar(mensaje, m_codificadora):
    return(m_codificadora*mensaje)

## ------------ Decodificar mensaje ------------
def decodificar(mensaje,m_codificadora):
    m_inversa = linalg.inv(m_codificadora)
    print(m_inversa)
    return (m_inversa*mensaje)

## ------------ Mensaje a Texto ------------
def mensaje_texto (mensaje,n,m):
    texto = []
    for i in range(m):
        for j in range(n):
```

```
            if (int(round(mensaje[j,i]))+64)>90:
                texto.append('*')
            else:
                texto.append(chr(int(round(mensaje[j,i]))+64))

    print(texto)

matriz_codif = np.matrix([[3,6,2],[2,3,1],[3,1,1]])

m = np.matrix([[12, 20, 27, 18, 1, 3],[
    9, 15, 16, 1, 20, 1],[19, 19, 1, 27, 1, 18]])

m_codificado = codificar(m,matriz_codif)
print(m_codificado)
m_decodificado = decodificar(m_codificado,matriz_codif)
print(m_decodificado)
mensaje_texto(m_decodificado,3,6)
```

```
[[128 188 179 114 125  51]
 [ 70 104 103  66  63  27]
 [ 64  94  98  82  24  28]]
[[-1.00000000e+00  2.00000000e+00  1.11022302e-16]
 [-5.00000000e-01  1.50000000e+00 -5.00000000e-01]
 [ 3.50000000e+00 -7.50000000e+00  1.50000000e+00]]
[[12. 20. 27. 18.  1.  3.]
 [ 9. 15. 16.  1. 20.  1.]
 [19. 19.  1. 27.  1. 18.]]
['L', 'I', 'S', 'T', 'O', 'S', '*', 'P', 'A', 'R', 'A', '*', 'A', 'T', 'A', 'C',
'A', 'R']
```

[ ]: