Laboratorio2

September 10, 2020

MAT281 - 2° Semestre 2020

Profesor: Francisco Alfaro Medina

[1]:
```
import numpy as np
```

0.1 Problema 01

La función `SMA()` queda de la siguiente forma:

[2]:
```
def SMA(Array,Numero):
    stop = np.shape(Array)[0] - Numero + 1   #es para detenernos en ese punto, y no iterar demás
    AF = [np.cumsum(Array[i:i + Numero],dtype=float)[Numero - 1] / Numero for i in range(0,stop)] #Aca ponemos cada
                                                                                                  
            #media  considerando el
                                                                                                  
#largo Numero para volverlo un array
    return np.array(AF)  #Finalmente lo volvemos un arreglo
```

[3]:
```
a = np.array([5,3,8,10,2,1,5,1,0,2])
SMA(a,2)
```

[3]:
```
array([4. , 5.5, 9. , 6. , 1.5, 3. , 3. , 0.5, 1. ])
```

0.2 Problema 02

Notemos primero, que no siempre dará una matriz ya que nosotros elegimos el numero de columnas y desfase. Por ejemplo si escogemos ***Numero de Columnas***: 7 y ***Desfase***: 2 en el arreglo ***[1,2,3,4,5,6,7,8,9,10]*** la primera fila tendrá sus 7 columnas, pero la segunda tendrá solo 5 elementos.

Pensando en ello, así queda la funcion `Strides()`.

[4]:
```
def Strides(Arreglo, Columnas, Desfase):
    largo = np.shape(Arreglo)[0]    #Obtenemos el largo del arreglo
```

```
    Matriz = [Arreglo[i:i+Columnas] for i in range(0,largo - Desfase,Columnas -␣
↪Desfase)] #Agregamos el arreglo
                                                                                ␣
↪          #desde el desfase +
                                                                                ␣
↪         #la cantidad de columnas,
                                                                                ␣
↪     #y así agregamos el arreglo fila
                                                                                ␣
↪     #de largo columna con el desfase
    return np.array(Matriz)  #Finalmente lo volvemos un arreglo
```

[5]:
```
b = np.array([1,2,3,4,5,6,7,8,9,10])
Strides(b,4,2)
```

[5]:
```
array([[ 1,  2,  3,  4],
       [ 3,  4,  5,  6],
       [ 5,  6,  7,  8],
       [ 7,  8,  9, 10]])
```

0.3 Problema 03

Para la función `EsCuadradoMagico()` se crea la función `EsCuaYSonCon()`, para verificar si cumple con que es cuadrada y si los numeros en la matriz son consecutivos del 1 al n^2. Finalmente, ambas funciones quedan de la siguiente manera:

[6]:
```
def EsCuaYSonCon(Matrix):
    n,m = np.shape(Matrix)
    if n != m:
        return False      #Aquí vemos si es una matriz cuadrada
    for i in range(1,n ** 2 + 1):   #Con esto vemos si están todos los numeros␣
↪consecutivamente del 1 al n cuadrado
        if i not in Matrix:
            return False
    return True

def EsCuadradoMagico(Matrix):
    if EsCuaYSonCon(Matrix) == False:   #Vemos si se cumple o no lo pedido
        return False                    #(Matriz cuadrada + numeros␣
↪consecutivos del 1 al n cuadrado)
    n,m = np.shape(Matrix)
    SumasF, SumasC = Matrix.sum(axis = 1), Matrix.sum(axis = 0) #Arreglos con␣
↪la suma de cada fila y columna
    SumaD1, SumaD2 = sum(Matrix.diagonal()), sum(np.fliplr(Matrix).diagonal())␣
↪#Suma de la diagonal a la der e izq
```

```
    if SumaD1 == SumaD2:
        for i in range(0,n):
            if SumasF[i] != SumaD1 and SumasC[i] != SumaD1:  #Verificamos si se␣
↪cumple que es cuadrado magico o no
                return False
        return True
    return False
```

[7]:
```
A = np.array([[4,9,2],[3,5,7],[8,1,6]])
B = np.array([[4,2,9],[3,5,7],[8,1,6]])
print(EsCuadradoMagico(A))
print(EsCuadradoMagico(B))
```

```
True
False
```