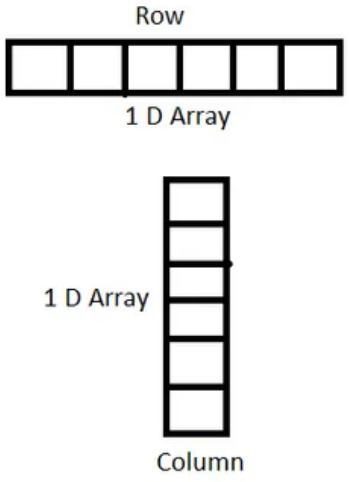
PYTHON APLICADO A LA CLIMATOLOGÍA

Bch. Javier Chiong Ravina

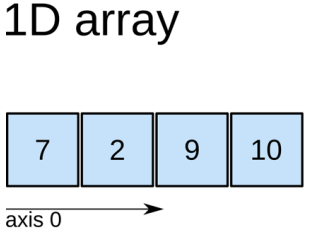
Numpy

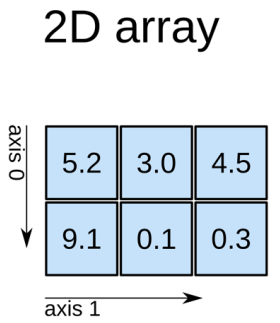
https://numpy.org/

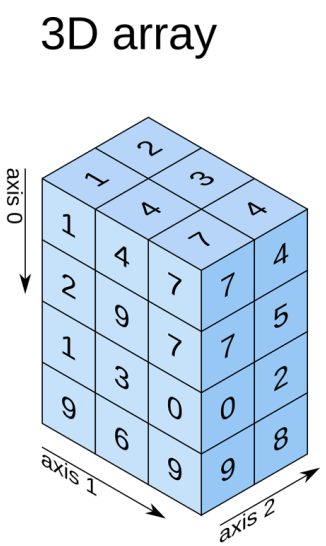
- Es una librería diseñada para trabajar arrays(matrices) 1-Dimensional y N-Dimensionales
- Es la base de desarrollo de otras librerías como Scipy, Scikit-Learn, etc.
- Tiene la capacidad de realizar operaciones complejas de los elementos como el álgebra lineal



https://learnprogramo.com/array-in-c-20/

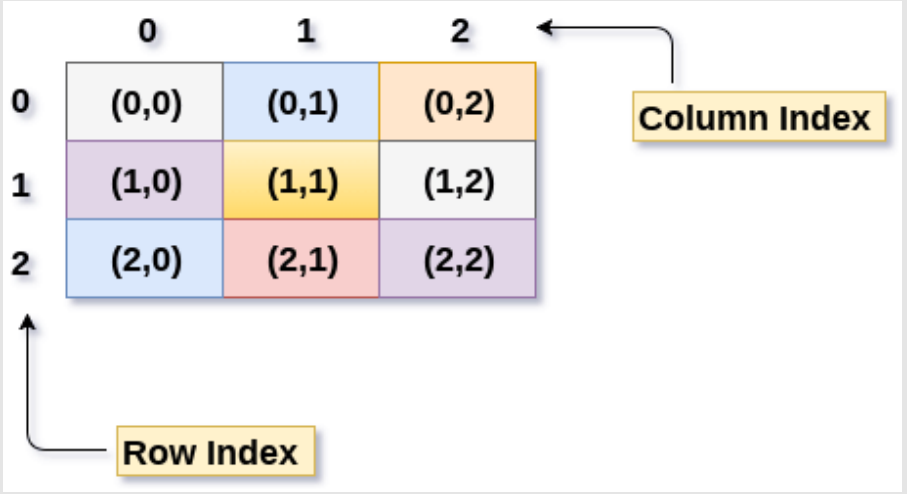




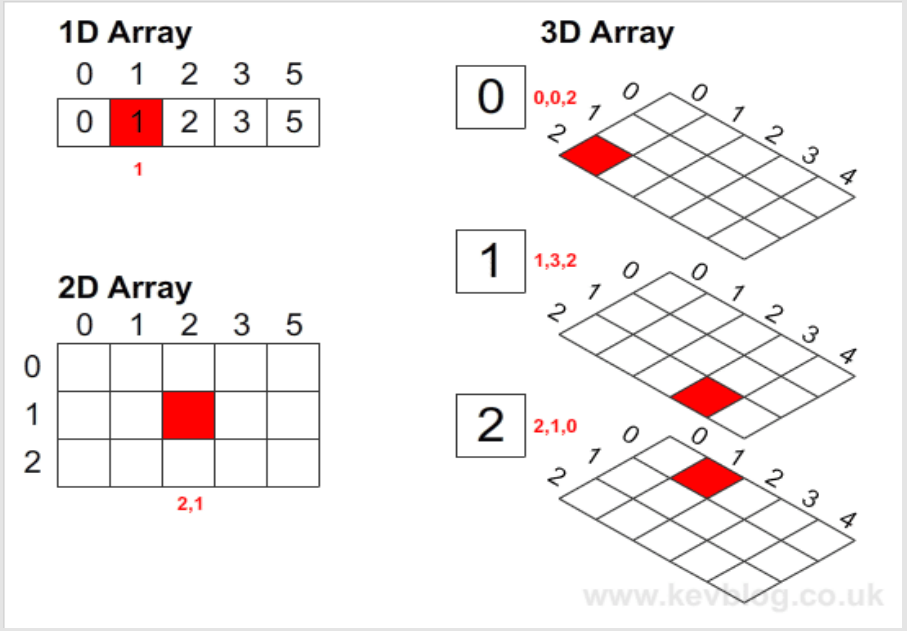


https://jovian.ai/abhianil96/python-numerical-computing-with-numpy

Índex de los arrays



https://iq.opengenus.org/2d-array-in-numpy/



https://www.kevblog.co.uk/create-3d-array-in-actionscript-3/

Atributos de los arrays

array.shape : retorna la forma del array

array.size : retorna el tamaño del array

array.ndim : retorna el número de dimensiones del array

array.dtype : retorna el formato del array

Módulos de Numpy

np.array() : crea un array	np.random.random() : crea un array con números decimales de forma aleatoria	np.random.randint() : crea un array con números enteros de forma aleatoria	np.linspace(a, b, n) : crea un array con n elementos, que van desde a hasta b
np.arange(a, b, Delta) : crea un array de a hasta b (sin considerarlo) cada Delta unidades	np.ones() : crea un array de valores uno	np.zeros() : crea un array de ceros	np.transpose() : transpone el array

Ejercicio 1

```
#importar la libreria
import numpy as np
#crear un array (min, max, incremento)
x = np.arange(0,16,1)
x  #matriz 1D
#explorar la forma
x.shape  #cambio de forma
x = x.reshape(4,4)
x  #matriz 2D
```

Ejercicio 1

```
#ejericios con índices
#Imprime algunos elementos
#(recordar que los índices comienzan en 0):
print(x[3])
print(x[1:3], x[1:4])
print(x[2,1])
print(x[1,:])
print(x[0:2,1:4])
# explorando atributos
x.ndim
x.shape
x.size
x.dtype
```

Estadísticas principales

array.sum() : suma todos los elementos del array

array.min() : retorna el mínimo valor del array

array.max() : retorna el máximo valor del array

array.mean() : retorna el promedio del array

array.std() : retorna la desviación estándar del array

array.sort() : ordena el array

array.astype(formato) : cambia el formato del array

array.reshape() : re-estructura la forma del array

array.flatten() : re-estructura el array a 1D

array.nanmean():Calcula la media aritmética ignorando los NaN.

array.var() : Calcula la variancia

Ejercicio 2

```
#Ejercicio 2: estadistica
np.mean(x, axis=0)
#operaciones aritmeticas
x = x - np.mean(x, axis=0)
x
#cambiar de formato un array
x = x.astype('float’)
x.dtype
x
```

Ejercicio 3

np.concatenate() : concatena arrays

```
#ejercicio 3 con operaciones
u = np.array([1,3,5,7,9,11])
w = np.array([0,2,4,6,8,12])
#dimensiones de los array
print(u.ndim, w.ndim)
#operacion concatenar
np.concatenate([u,w]) #array 1D
#cambiar de forma y concatenar
#Re-estructura los arrays y concatenar
u = u.reshape([2,3])
w = w.reshape([2,3])
np.concatenate([u,w], axis=0)
np.concatenate([u,w], axis=1)
```

Ejercicio 4

np.where() : realiza una operación con condicionales

```
#Ejercicio 4 con where
#crear array con numeros aleatorios enteros
m = np.random.randint(1,10,[8,6])
#crear un nuevo array condicion where
n = np.where(m<5,0,1)
print(n)
p = np.where((m<8)&(m>3),0,1)
#otro ejemplo
r = np.random.randint(1,10,[8,6])
s = np.random.randint(1,10,[8,6])
t = np.where(r>s,0,1)
print(t)
```

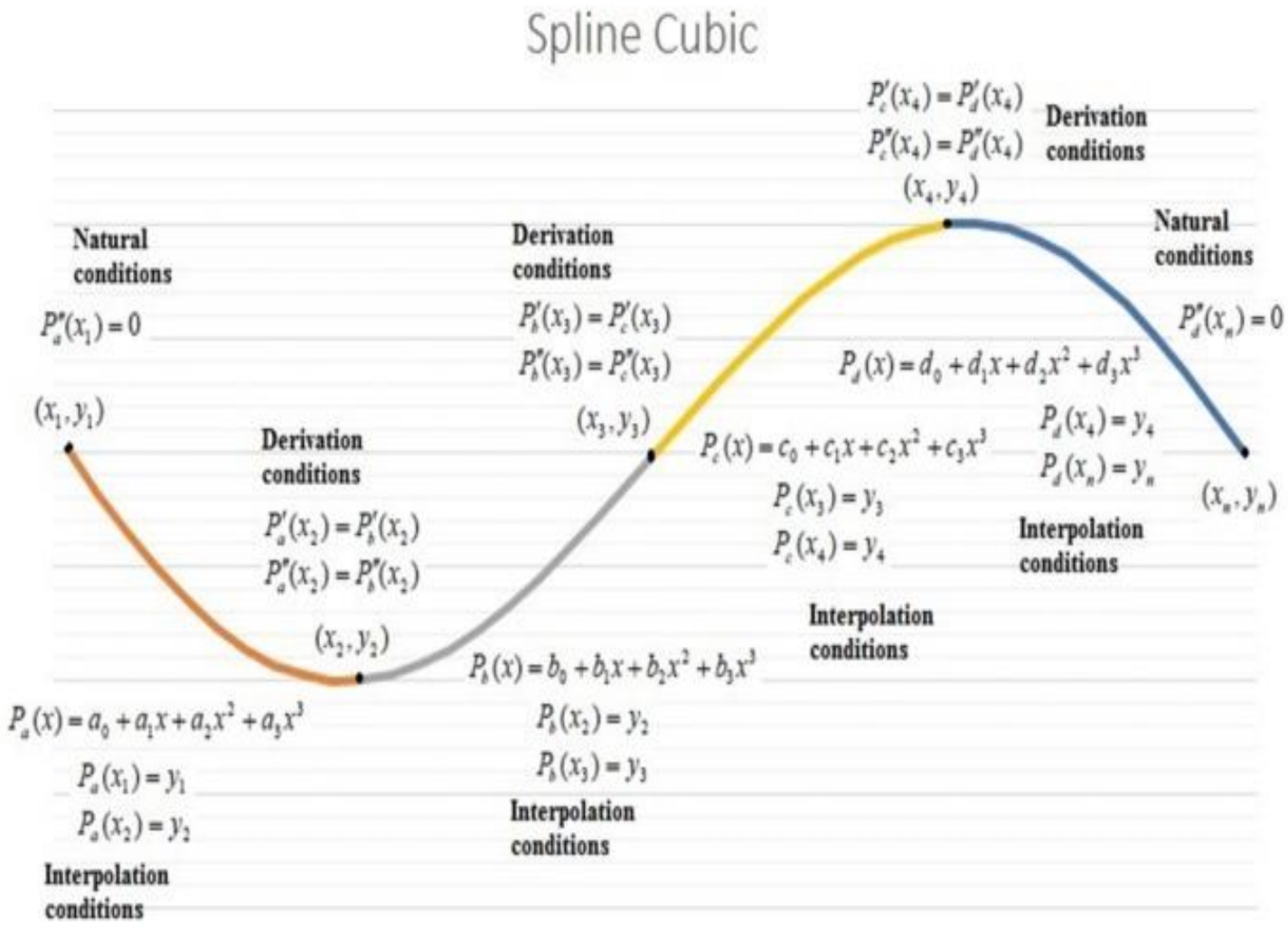
Ejercicio 5

np.meshgrid() : convierte un array 1D en 2D, utilizando el número de elementos de los arrays 1D.

```
#ejercicio 5 con meshgrid
#crear un array 1D
j = np.array([3,5,7,9])
j
j.shape
#crear otro array 1D
k = np.array([0,1,4])
k
k.shape
#operacion meshgrid
j2d, k2d = np.meshgrid(j, k)
j2d
j2d.shape # array 2D
k2d
k2d.shape #array 2D
```

Interpolación Spline

- Función que interpola funciones cúbicas por partes en un conjunto de puntos y garantiza la suavidad en los puntos (Ryan G. McClarren, 2018).
- Garantiza precisión, evita oscilaciones y obtiene funciones suavizadas (Ryan G. McClarren, 2018).



¿POR QUÉ SUAVIZAR O QUITAR LA ALTA VARIABILIDAD EN UNA SERIE DE TIEMPO?

- Arguez y Applequist, 2003:

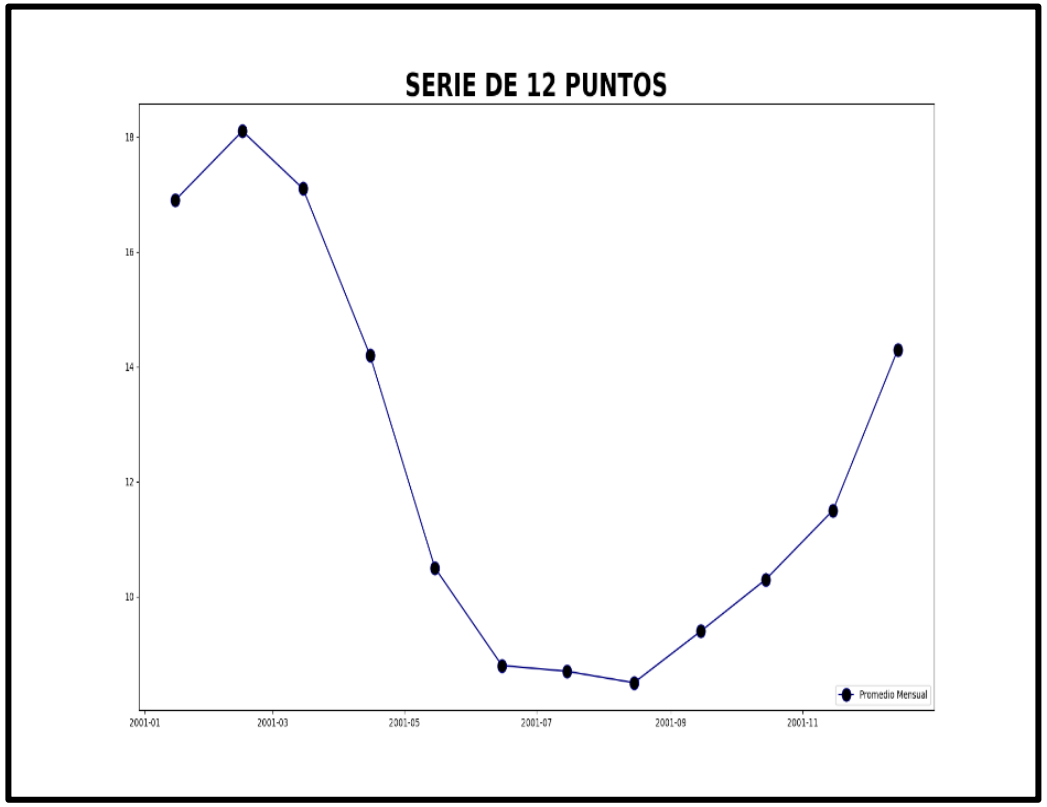
Porque las normales diarias calculadas promediando los 30 valores para cada día del año (es decir, "normales diarias crudas") constituyen una serie de tiempo bastante ruidosa debido a la variabilidad del muestreo. Este efecto se agrava si hay valores perdidos en el registro de datos.

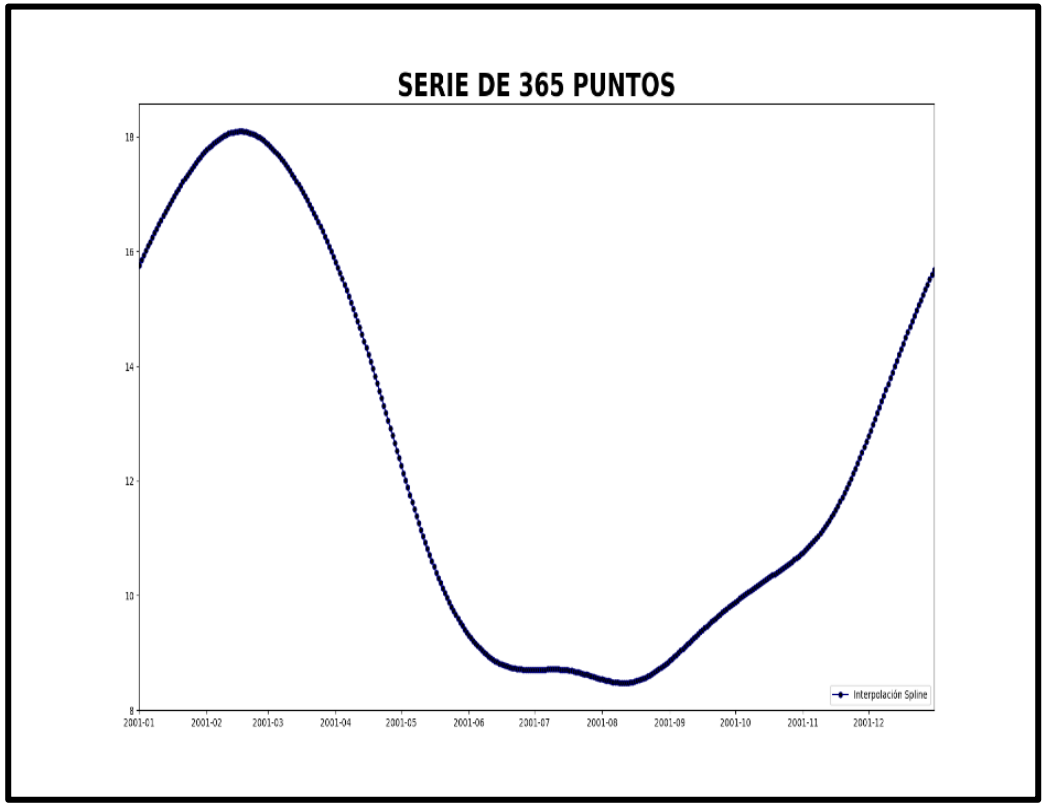
- Wilks, 2006:

Considerar que una serie de tiempo observada ha sido generada por un proceso teórico (modelo) es conveniente porque permite inferir características de valores futuros, aún no observados, de una serie de tiempo a partir de los inevitablemente limitados datos disponibles.

Interpolación Spline

- Generación de vector fecha triplicado (365 x 3), vector de promedios mensuales triplicados y vector posición del 15 de cada mes.
- Obtención de nodos y coeficientes de interpolación spline con los promedios mensuales triplicados y el vector posición del 15 de cada mes.
- Evaluación del polinomio con los valores del vector fecha triplicado y obtención de la función interpolada spline.





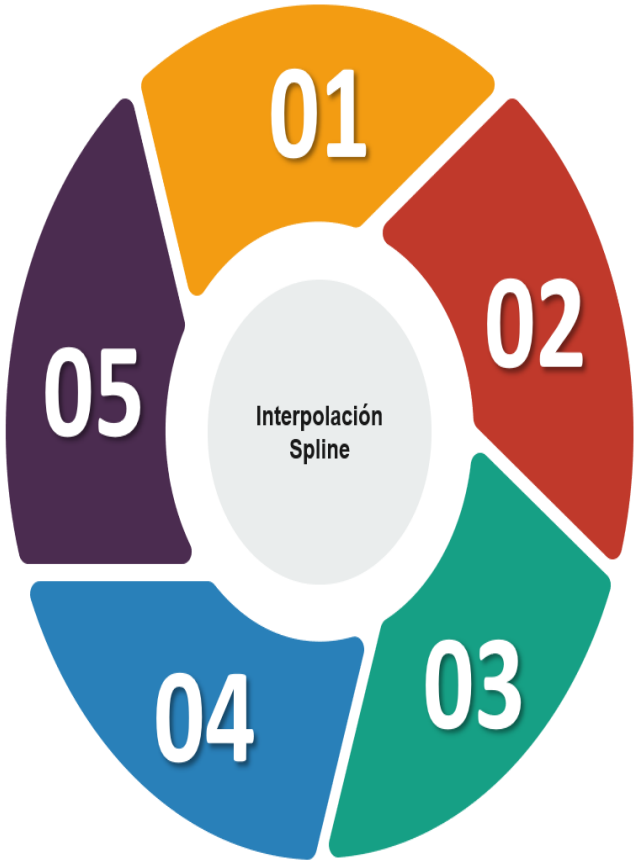
Interpolación Spline

Extracción de valores centrales

Extracción de los valores centrales de la función que van desde los valores de X_Q= 366 (367) hasta el valor X_Q = 729 (730) para un año normal (bisiesto).

Función interpolada

Evaluación del polinomio suavizado y sus derivadas para X_Q, que contiene 1095 (1098) días en un año normal (bisiesto) en donde finalmente se obtuvo la función interpolada mediante spline.



Normales Mensuales

Cálculo de normales mensuales de temperaturas extremas mediante la media de los promedios mensuales de las temperaturas extremas.

Generación de vectores

Generación de vectores de dimensión $Y = 36, X = 36, X_Q = 1095$.

Cálculo de coeficientes

Cálculo de aproximación de spline suavizado de grado 3 obteniendo los nodos y coeficientes de la representación spline.