

CURSOS  
INTERSEMESTRALES



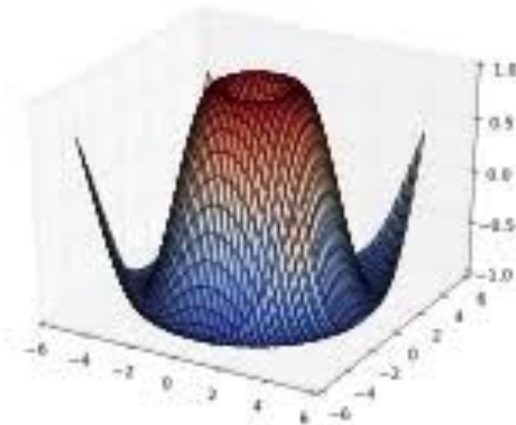
PROTECO

# Matplotlib

Python avanzado  
mayo 2019

# ¿Qué es Matplotlib?

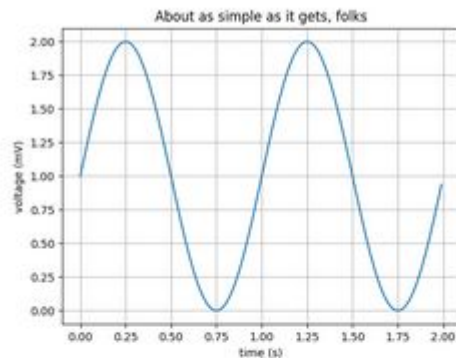
Matplotlib es una biblioteca de dibujo de gráficas 2D y 3D . Utilizaremos `matplotlib.pyplot` porque es una colección de funciones (módulo) de estilo de comando que hacen que matplotlib funcione como MATLAB.



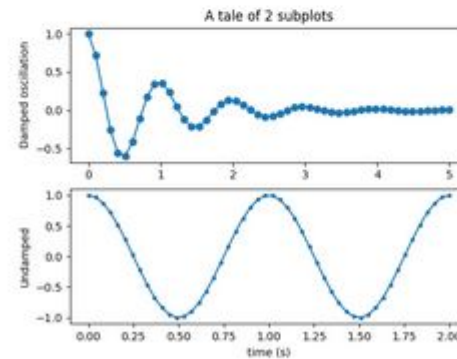
**matplotlib**

[https://matplotlib.org/tutorials/introductory/sample\\_plots.html](https://matplotlib.org/tutorials/introductory/sample_plots.html)

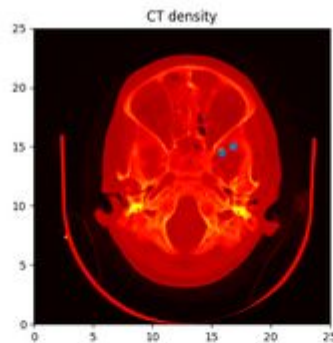
# Tipos de gráficas



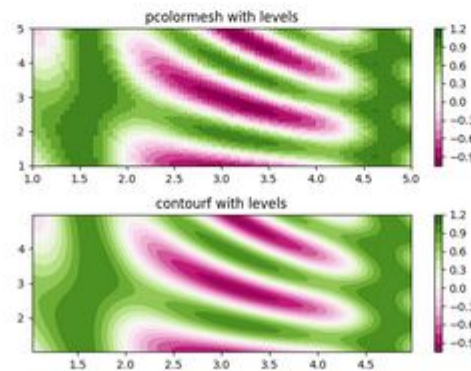
Simple Plot



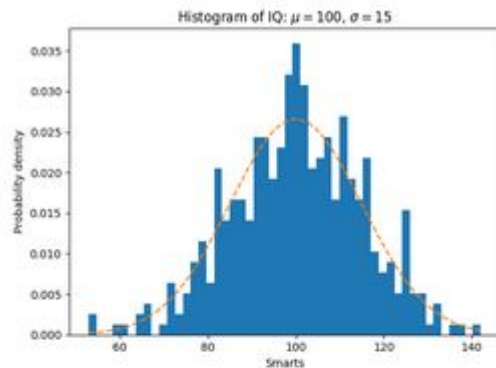
Subplot



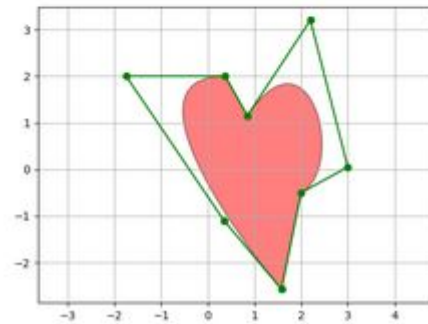
Example of using `imshow()` to display a CT scan



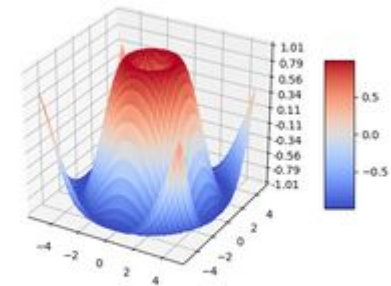
Example comparing `pcolormesh()` and `contour()` for plotting two-dimensional data



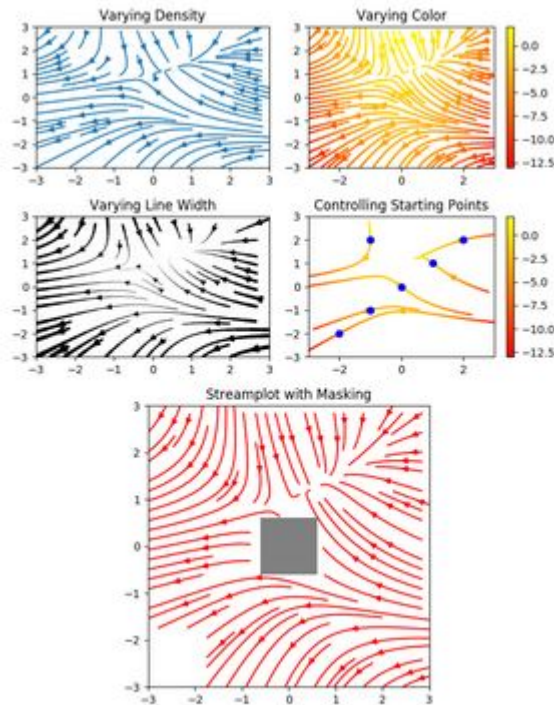
**Histogram Features**



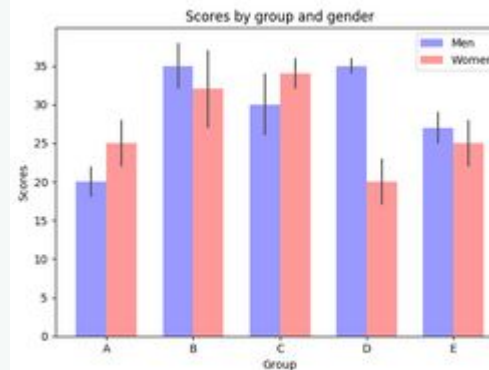
**Path Patch**



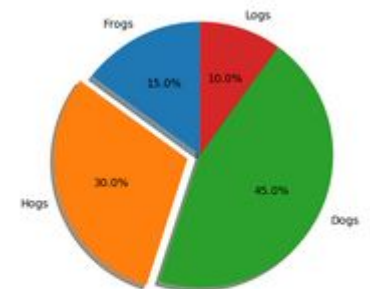
**Surface3d**



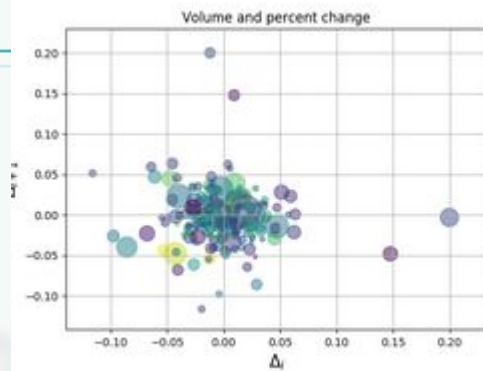
**Streamplot with various plotting options.**



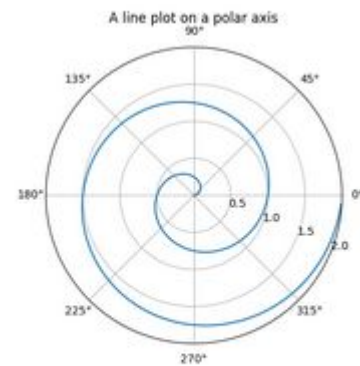
**Barchart Demo**



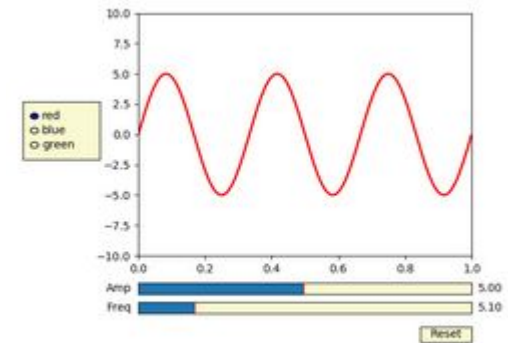
**Pie Features**



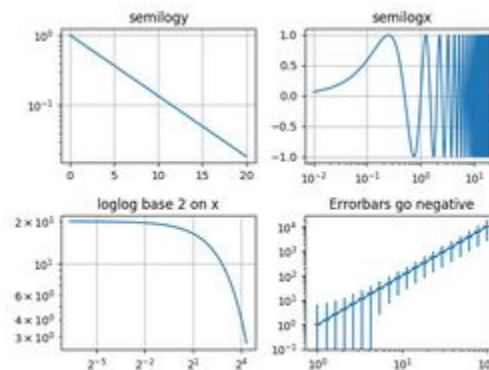
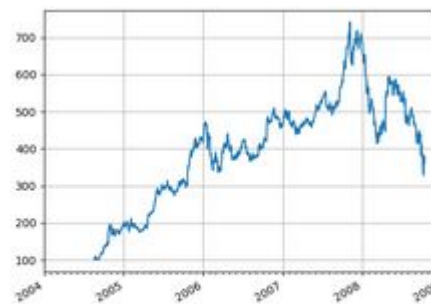
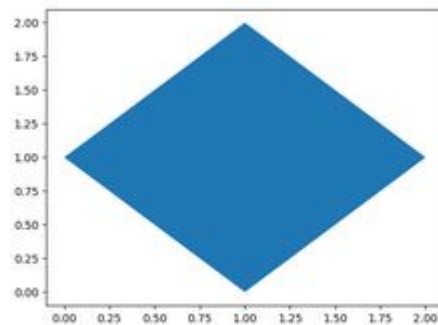
Scatter Demo2



Polar Demo



Slider and radio-button GUI.

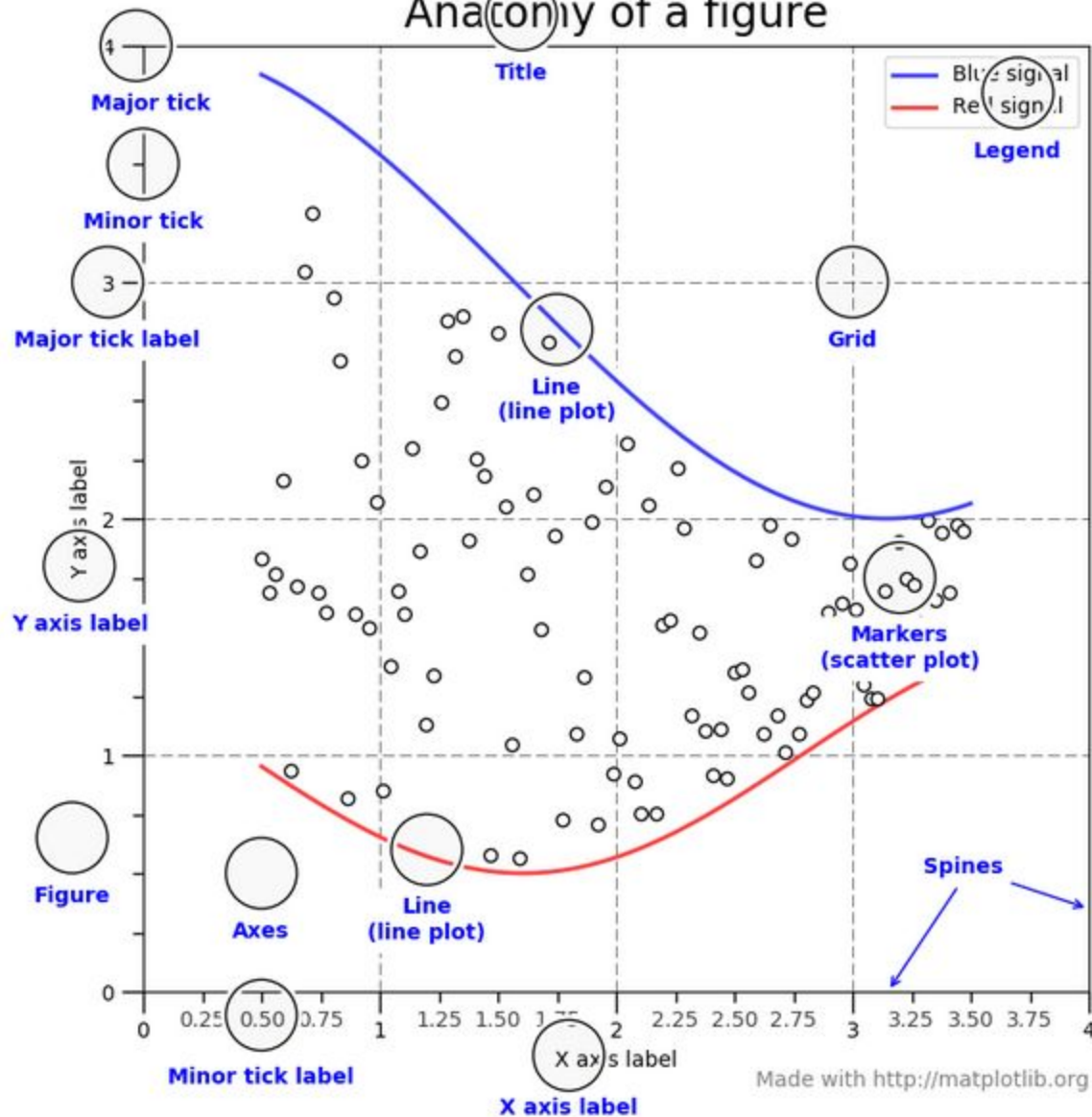


Log Demo

Date



# Anatomy of a figure



Figure

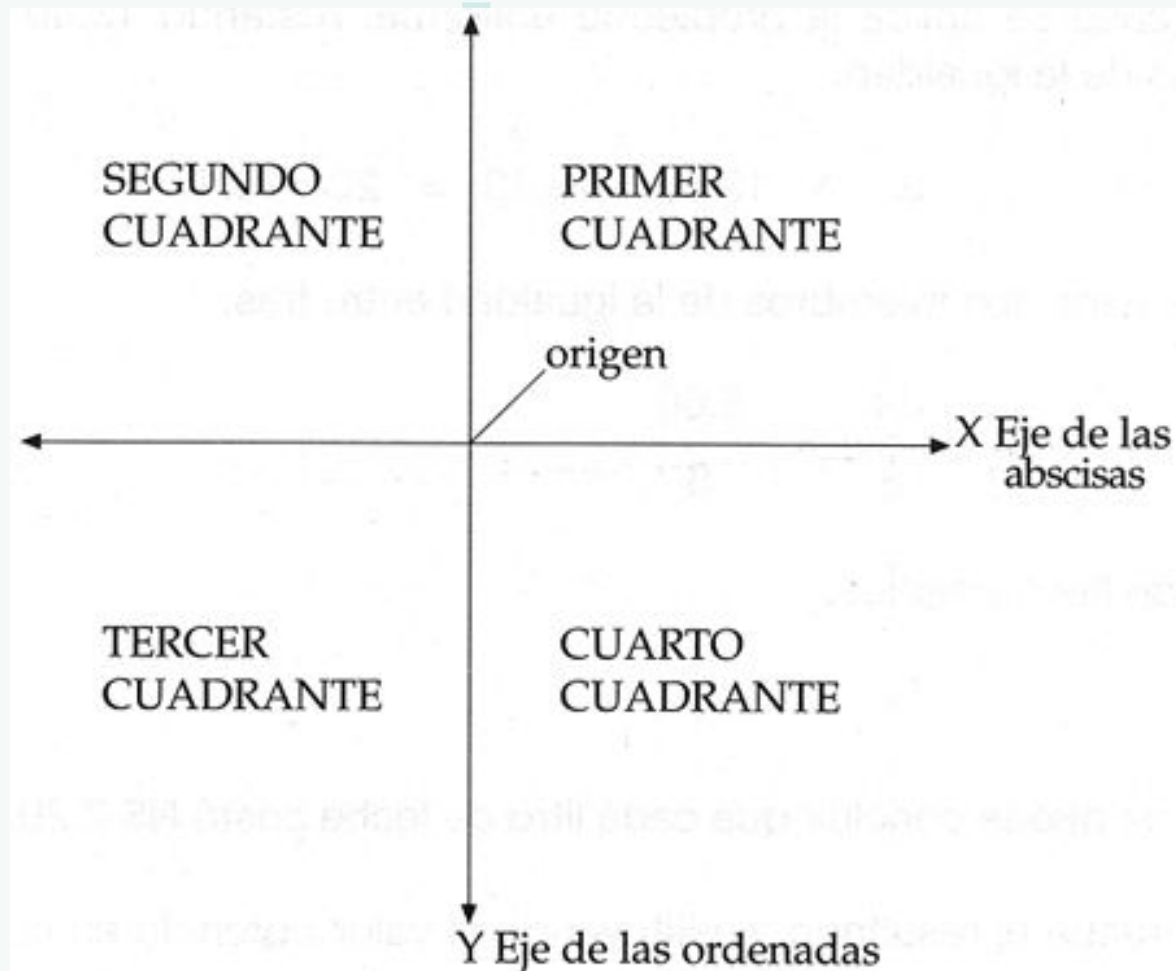


**plot(x, y, linestyle, linewidth, marker)** → Permite incluir varias gráficas en una única figura.

- x = Abscisas.
- y = Ordenadas. Tanto x como y pueden ser abscisas tuplas, listas o arrays. La única condición es que el tamaño de ambas debe ser el mismo ya que en caso contrario python nos devolverá un fallo de tipo dimesión. También se puede hacer una gráfica sin especificar la coordenada x.
- linestyle = color y tipo de dibujar la gráfica. Por ejemplo 'r'
- linewidth = ancho de línea.
- marker = Marcador.



# Ordenadas y Abscisas





# Linestyle

line styles

1.1 Dotted line

1.2 Dash-dot line

1.3 Dashed line

1.4 Solid line



PROTECO

# Linestyle

## Colores

'b' Azul

'g' Verde

'r' Rojo

'c' Cian

'm' Magenta

'y' Amarillo

'k' Negro

'w' Blanco

También se puede escribir el color de las siguientes formas: nombres ('green'); cadenas hexadecimales ('#008000'); tuplas con convención RGB (0,1,0); intensidades de escala de grises ('0.8').



# Marcadores

Los tipos principales son:

'+'

'\*'

'.'

'/'

'.'

'1'

'2'

'3'

'4'

'<'

'>'

'D'

'H'

'^'

' '

'd'

'h'

'o'

'p'

's'

'v'

'x'



# Subplot (Subgráficas)

**subplot(numRows/filas, numCols, plotNum)** → Permite incluir varias gráficas en una única figura.

- numRows/filas = Número de filas
- numCols = Número de columnas
- plotNum = Número de gráfica, 1,2,3,4 hasta llegar a el resultado de numRows\*NumCols

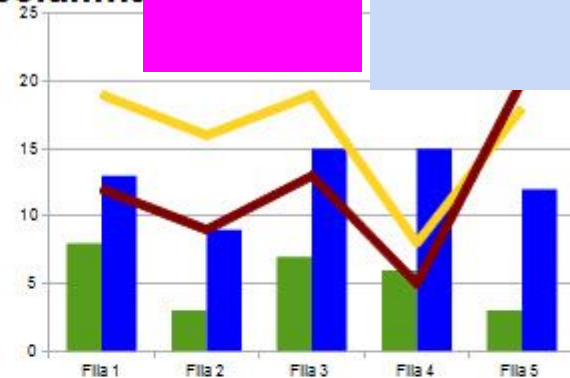
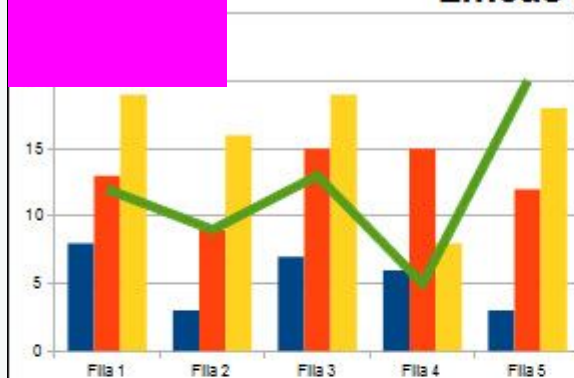


Columna 1

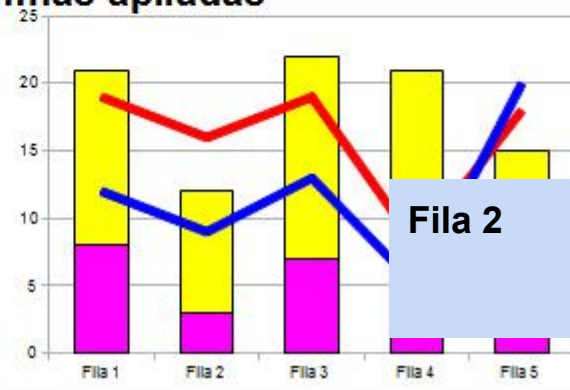
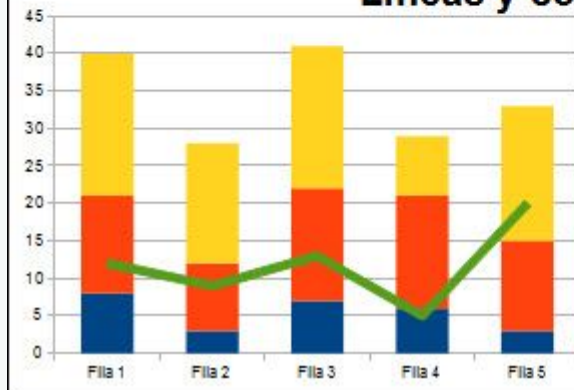
Columna 2

Fila 1

Líneas y columna



Líneas y columnas apiladas



Fila 2



PROTECO

# Pasos para subploteo

1.- Diseñar la cantidad de filas y columnas con `plt.subplot(filas,columnas,posición)`

2.-Indicar qué gráfica deseo que se coloque, es decir con `plt.plot(x,y,linestyle, width, marker)`





# Ejercicio 1

Ejercicio, hacer una gráfica de 3 filas y 2 columnas con los datos de clase.

`subplot(2,1,1)`

`subplot(2,1,2)`

`subplot(1,2,1)`

`subplot(1,2,2)`

`subplot(2,2,1)`

`subplot(2,2,2)`

`subplot(2,2,3)`

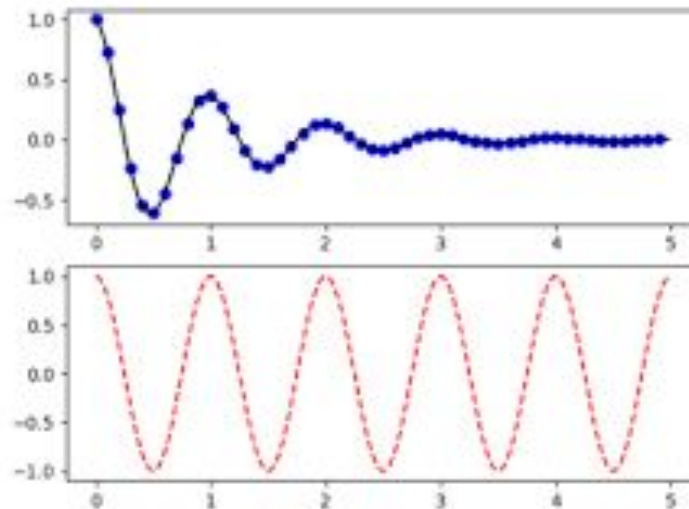
`subplot(2,2,4)`





# matplotlib.pyplot.figure()

[https://matplotlib.org/2.0.1/api/figure\\_api.html](https://matplotlib.org/2.0.1/api/figure_api.html)

El módulo de figura proporciona el nivel superior :“Artista”. La figura contiene todos los elementos de la gráfica.





```
matplotlib.pyplot.figure(num=None, figsize=None,  
dpi=None, facecolor=None, edgecolor=None,  
frameon=True, FigureClass=<class  
'matplotlib.figure.Figure'>, clear=False, **kwargs)
```



**num:** predeterminado: ninguno

Si no se proporciona, se creará una nueva figura y el número de la figura se incrementará. Los objetos de figura mantienen este número en un atributo de número.

**figsize:** Anchura, altura en pulgadas. Si no se proporciona, el valor predeterminado es = [6.4, 4.8].

**dpi :** entero, opcional, predeterminado:  
ninguno

Resolución de la figura. Si no se proporciona, el valor predeterminado es = 100.

**facecolor:** color de fondo. predeterminado  
:blanco.

**edgecolor:** color del borde.  
predeterminado:blanco  
etc...

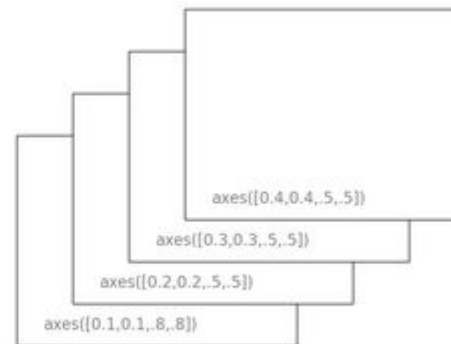
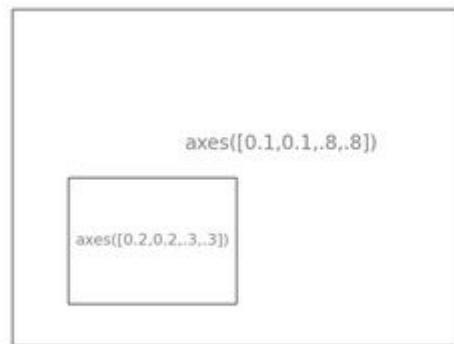


Argumento	Por defecto	Descripción
num	1	número de figura
figsize	figure.figsize	tamaño de figura en pulgadas (ancho, alto)
dpi	figure.dpi	resolución en puntos por pulgada
facecolor	figure.facecolor	color de fondo del dibujo
edgecolor	figure.edgecolor	color del borde alrededor del fondo del dibujo
frameon	True	dibujar figura en marcos o no



## `add_axes(* args, ** kwargs)`

Agregue un eje en una posición [izquierda, abajo, ancho, altura] donde todas las cantidades estén en fracciones de la figura ancho y alto. los kwargs son ejes que establece el tipo de proyección de los ejes (Polar, rectilinea,etc)





# Pasos para utilizar add\_axes()

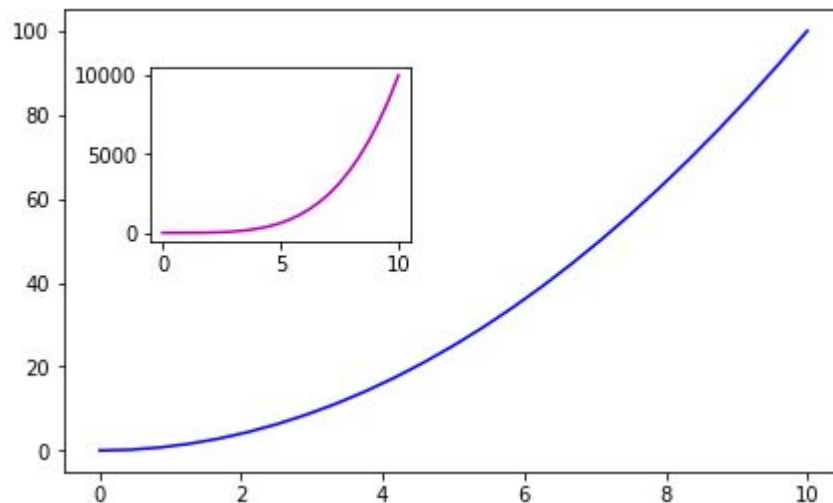
- 1.- Establecer que es de tipo figura
- 2.- Establecer la posición que tendrán los ejes (1 o más gráficas)
- 3.-Asignar las gráficas que serán visualizadas según la posición.



Ejemplos:

`facecolor='g'` # agrega fondo de color verde

`projection='lambert'`, `projection='polar'` #hace que la proyección sea polar.



# Títulos, leyendas y nombres de ejes

**plt.xlabel('s', comandos\_optativos)** Etiqueta el eje de abscisas de la gráfica actual.

**plt.ylabel('s', comandos\_optativos)** Etiqueta el eje de abscisas de la gráfica actual.

**plt.title('s', comandos\_optativos)** Etiqueta el eje de abscisas de la gráfica actual.

s = Texto que aparecerá en el título

comandos\_optativos = En esta etiqueta englobamos todos los modificadores de la fuente etc.



```
plt.legend( ('Etiqueta1', 'Etiqueta2', 'Etiqueta3'), loc  
= 'upper left')
```

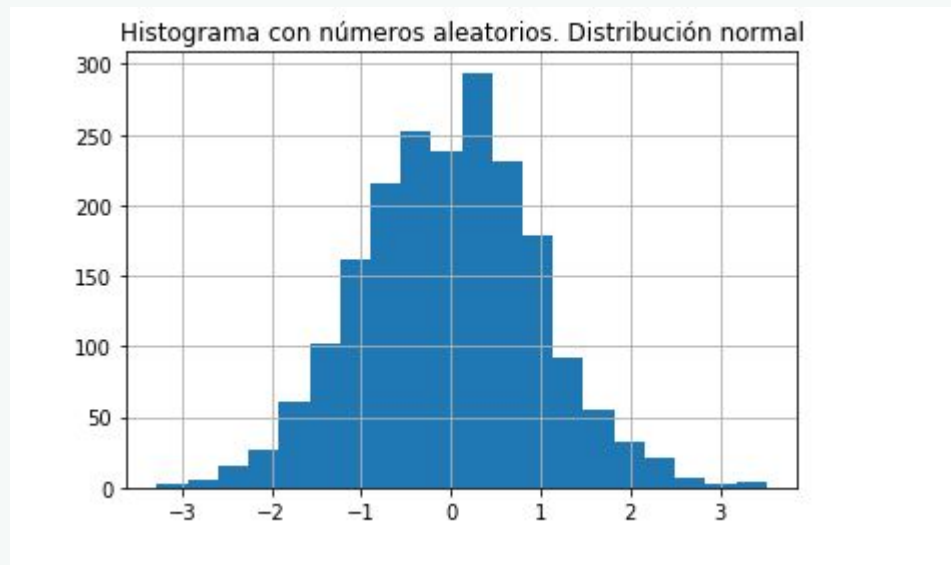
Opciones para loc:

```
best  
upper right  
upper left  
lower left  
lower right  
right  
center left  
center right  
lower center  
upper center  
center
```



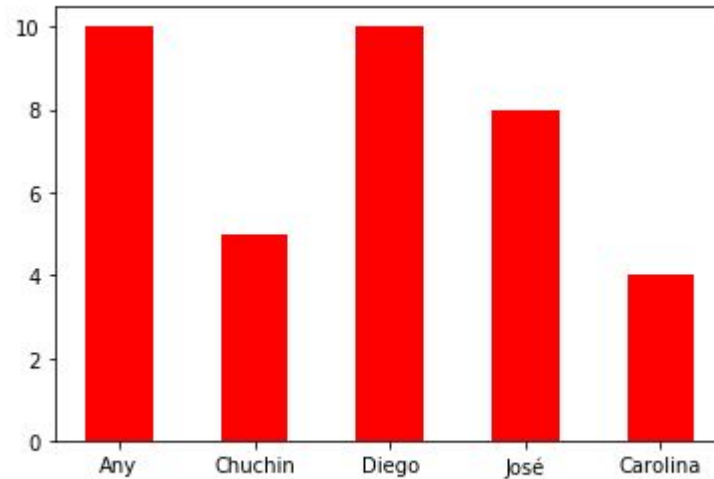
# Histograma

`matplotlib.pyplot.hist(datos, bin = separación)`



# Gráfica de barras

**`matplotlib.pyplot.bar(x, height, width=0.8, bottom=None, *, align='center', data=None, **kwargs)`**

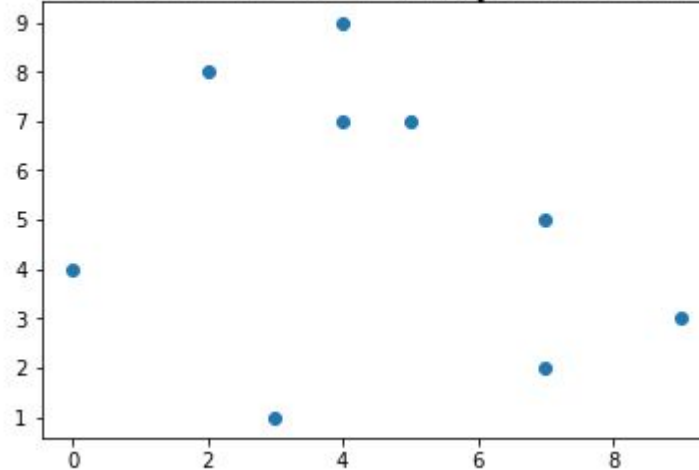




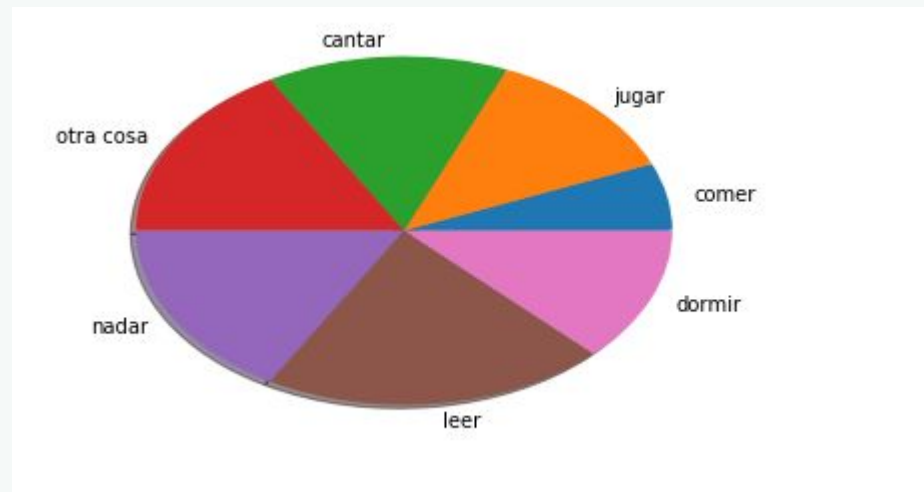
# Gráficas de dispersión

`plt.scatter(datosx, datosy)`

Gráfica de dispersión



```
matplotlib.pyplot.pie(x, explode=None,  
labels=None, colors=None, autopct=None,  
pctdistance=0.6, shadow=False, labeldistance=1.1,  
startangle=None, radius=None, counterclock=True,  
wedgeprops=None, textprops=None, center=(0, 0),  
frame=False, rotatelabels=False, *, data=None)
```



- \* x: como una matriz. Los tamaños.
- \* explode: similar a una matriz, opcional, predeterminado: ninguno
- \* labels s: lista, opcional, por defecto: ninguna
- \* colors: tipo matriz, opcional, por defecto: ninguno
- \* pctdistance: float, opcional, por defecto: 0.6. La proporción entre el centro de cada sector circular y el inicio del texto generado por autopct.
- \* shadow: dibuja una sombra debajo del pastel.



# Ejercicio

**Realiza un subploteo de 2 filas por 2 columnas.** En el espacio que desees coloca una gráfica simple, una gráfica de dispersión, una gráfica de barras y una gráfica de pastel. Cada gráfica debe tener un color diferente.

Hazlo lo más simple posible. si deseas puedes utilizar los siguientes datos:

Datos gráfica simple y de dispersión

equis = [1,2,3,4,5,6]

ye = [2,4,6,8,10,12]

Datos de barras y pastel

refrescos =

['Cocacola','Pepsi','Fanta','Mirinda','Peñafiel', 'Senzao']

popularidad = [7,2,3,2,1,10]



# Referencias

MTPLOTLIB

<https://matplotlib.org/contents.html>

Markdown

<https://joedicastro.com/pages/markdown.html>

MATPLOTLIB

<https://claudiovz.github.io/scipy-lecture-notes-ES/intro/matplotlib/matplotlib.html>



PROTECO