

+++++Nombre de la práctica	Matplotlib			No.	3
Asignatura:	Simulación	Carrera:	ISIC	Duración de la práctica (Hrs)	8

Alumno (a): María Lucero Rodea Martínez

I. Lugar de realización de la práctica (laboratorio, taller, aula u otro): Aula

II. Material empleado:

Laptop

III. Desarrollo de la práctica:

Introducción a Matplotlib

Matplotlib es una biblioteca que permite la creación de figuras y gráficos de calidad mediante el uso de Python.

- Permite la creación de gráficos de manera sencilla y eficiente.
- Permite la integración de gráficos y figuras en un Jupyter Notebook.

```
[1]: import matplotlib
import matplotlib.pyplot as plt
```

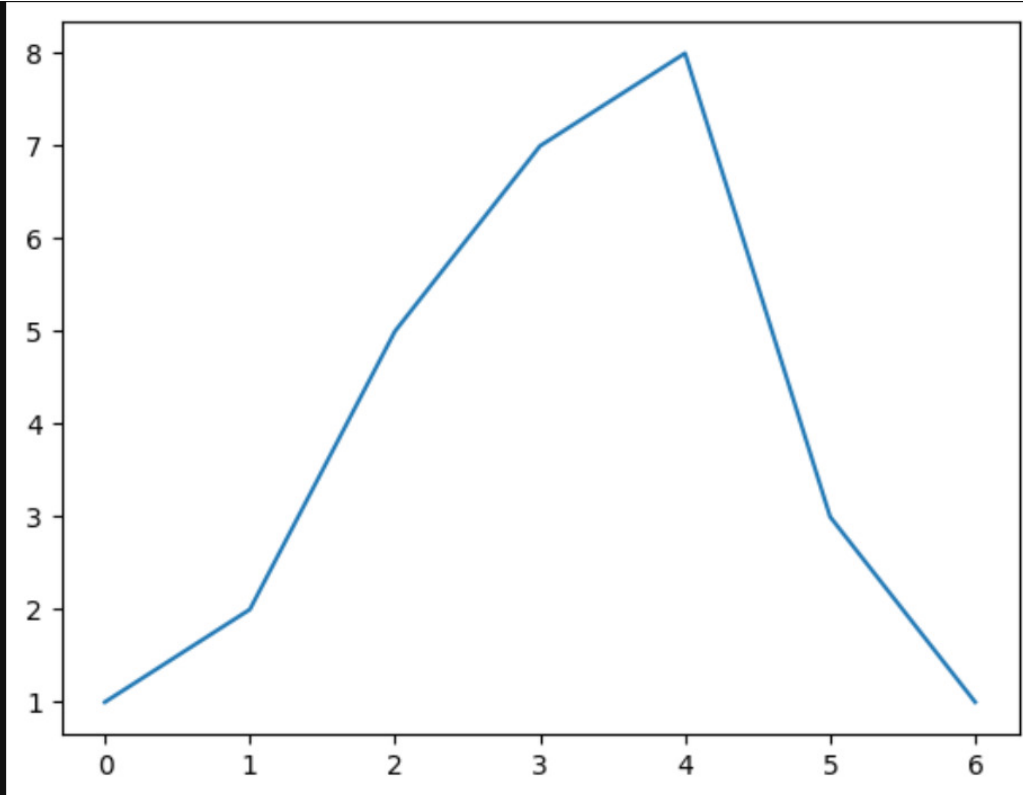
```
[2]: # Muestra los gráficos integrados dentro de Jupyter Notebook
%matplotlib inline
```

▼ Representación gráfica de Datos.

Si a la función de trazado se le da una matriz de datos, la usará como coordenadas en eje vertical, y utilizará el índice de cada punto de datos en el array como la coordenada horizontal.

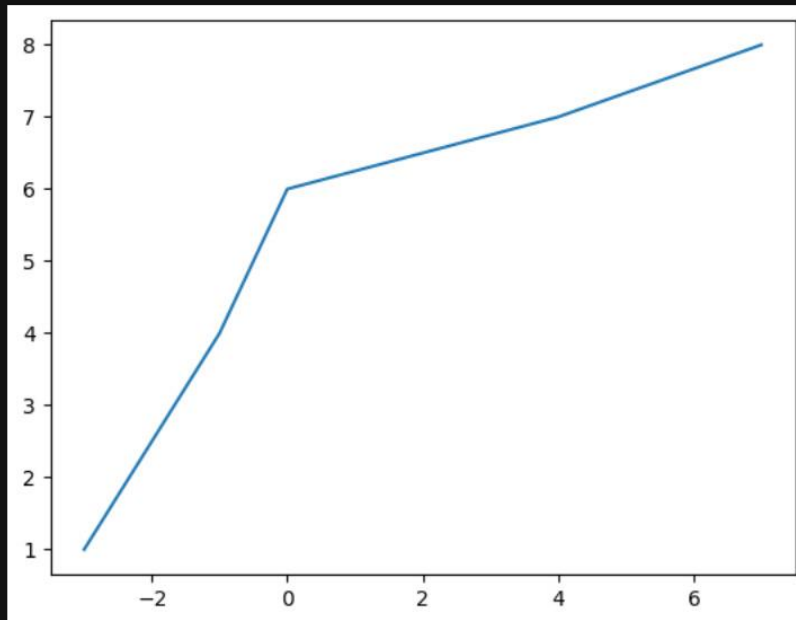
```
[3]: plt.plot([1, 2, 5, 7, 8, 3, 1])
```

```
[3]: plt.plot([1, 2, 5, 7, 8, 3, 1])
plt.show()
```



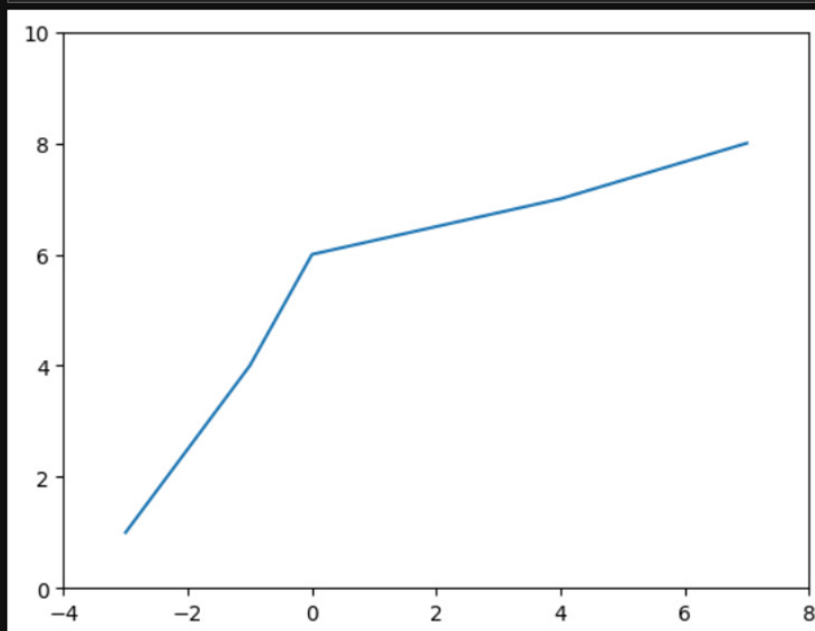
También se puede proporcionar dos matrices: una para el eje horizontal, y otra para el eje vertical.

```
[5]: plt.plot([-3, -1, 0, 4, 7], [1, 4, 6, 7, 8])  
plt.show()
```



Pueden modificarse las longitudes de los ejes para que la figura no se vea tan ajustada.

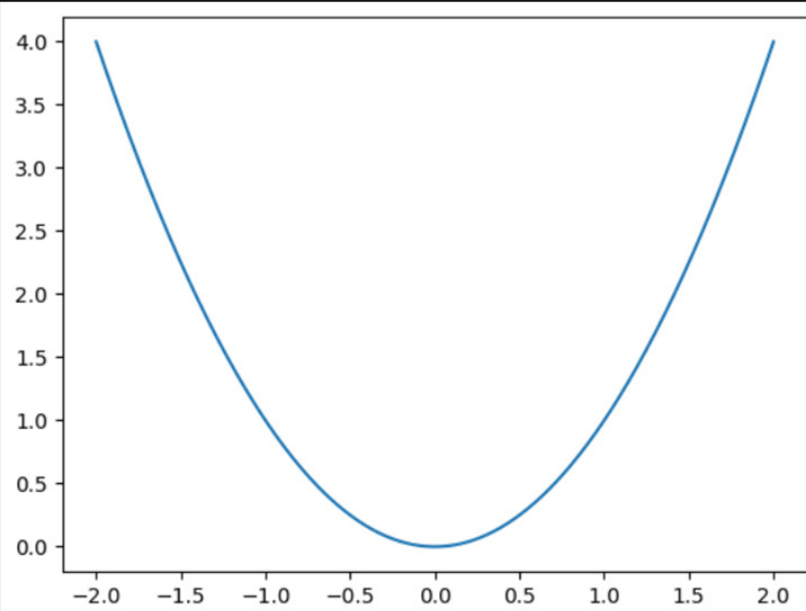
```
[6]: plt.plot([-3, -1, 0, 4, 7], [1, 4, 6, 7, 8])  
plt.axis([-4, 8, 0, 10]) # [xmin, xmax, ymin, ymax]  
plt.show()
```



Se sigue el mismo procedimiento para pintar una función matemática.

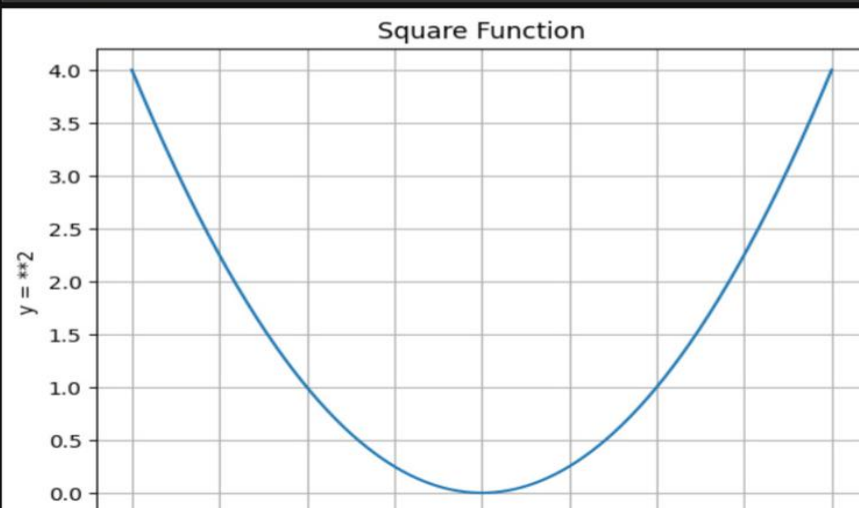
```
[9]: import numpy as np
x = np.linspace(-2, 2, 500)
y = x**2

plt.plot(x, y)
plt.show()
```



También puede modificarse el estilo de la gráfica para que contenga más información.

```
[10]: plt.plot(x, y)
plt.title("Square Function")
plt.xlabel("Chido")
plt.ylabel("y = **2")
plt.grid(True)
plt.show()
```

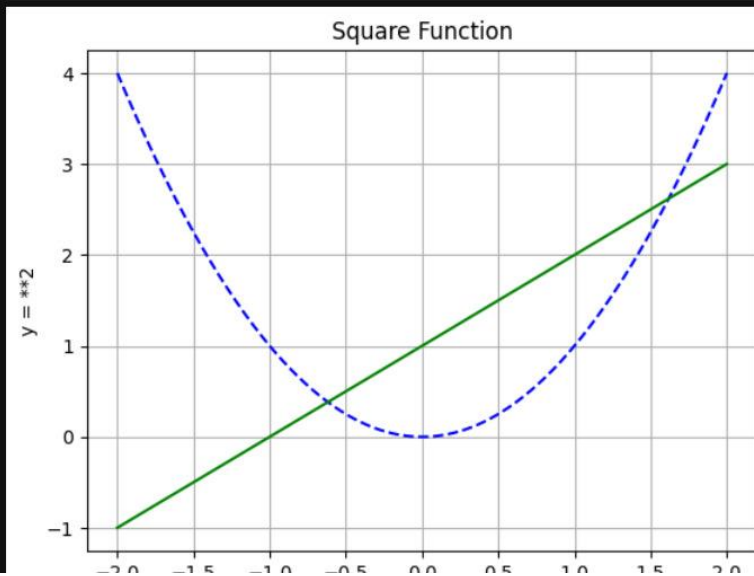




```
x = np.linspace(-2, 2, 500)
y = x**2
y2 = x + 1

plt.title("Square Function")
plt.xlabel("Chido")
plt.ylabel("y = **2")
plt.grid(True)

plt.plot(x, y, 'b--', x, y2, 'g')
plt.show()
```

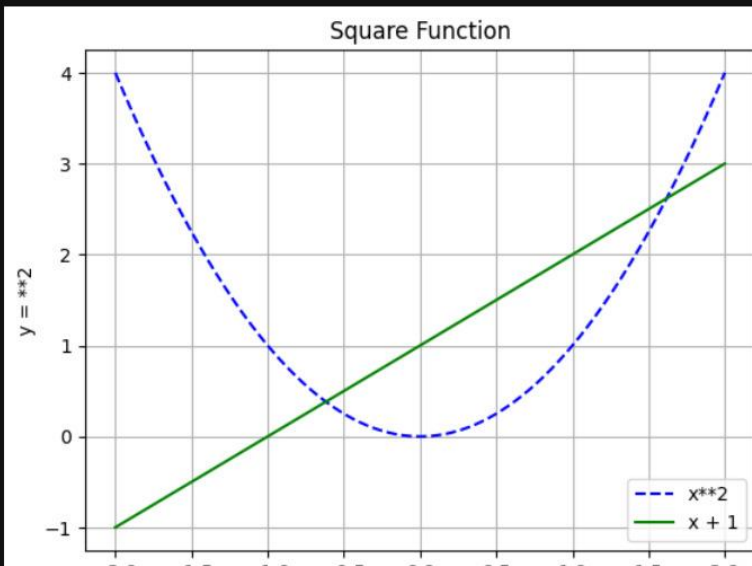




```
y2 = x + 1
```

```
plt.title("Square Function")  
plt.xlabel("Chido")  
plt.ylabel("y = **2")  
plt.grid(True)
```

```
plt.plot(x, y, 'b--', label = "x**2")  
plt.plot(x, y2, 'g', label = "x + 1")  
plt.legend(loc = "best") # La situa en la mejor localización  
plt.show()
```



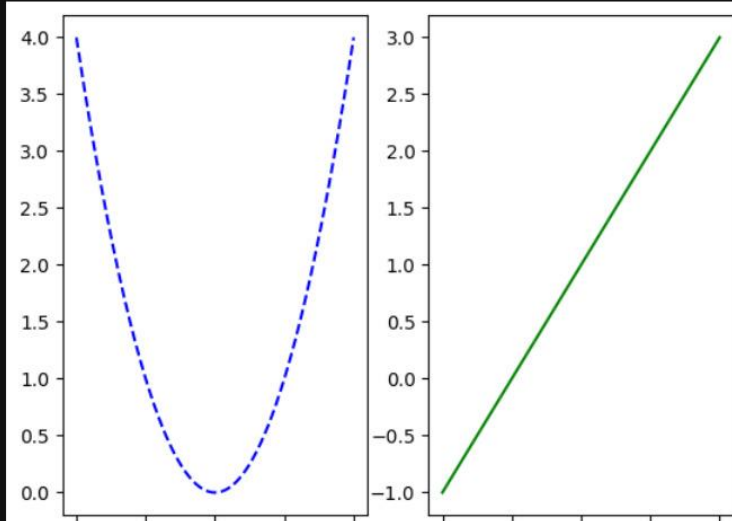


```
[19]: import numpy as np
x = np.linspace(-2, 2, 500)
y = x**2
y2 = x + 1

plt.subplot(1, 2, 1) # 1 Rows, 2 Columns, 1st Subplot
plt.plot(x, y, 'b--')

plt.subplot(1, 2, 2) # 1 Rows, 2 Columns, 2nd Subplots
plt.plot(x, y2, 'g')

plt.show()
```



Para que las gráficas no queden tan ajustadas, se puede hacer la figura más grande.

```
[21]: plt.figure(figsize = (14, 6))

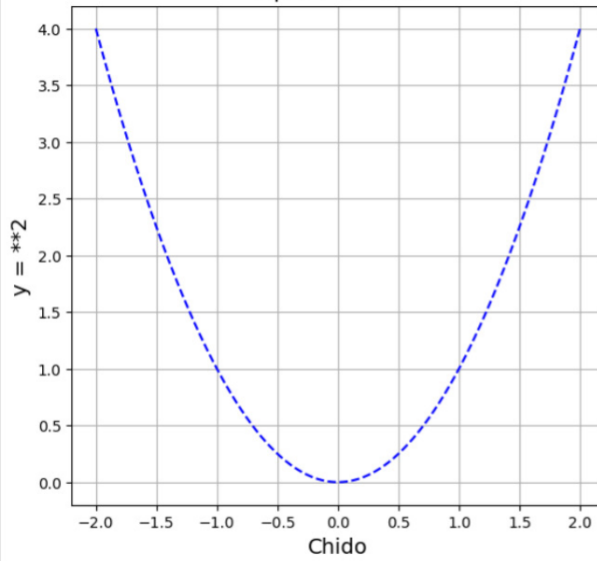
plt.subplot(1, 2, 1) # 1 Rows, 2 Columns, 1st Subplot
plt.plot(x, y, 'b--')
plt.title("Square Function", fontsize = 14)
plt.xlabel("Chido", fontsize = 14)
plt.ylabel("y = **2", fontsize = 14)
plt.grid(True)

plt.subplot(1, 2, 2) # 1 Rows, 2 Columns, 2nd Subplots
plt.plot(x, y2, 'g')
plt.title("Lineal Function", fontsize = 14)
plt.xlabel("Chido 2", fontsize = 14)
plt.ylabel("y = **2", fontsize = 14)
plt.grid(True)

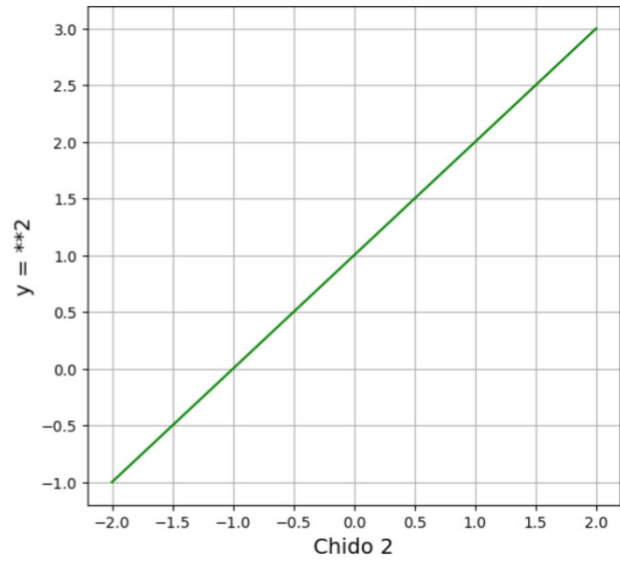
plt.show()
```



Square Function

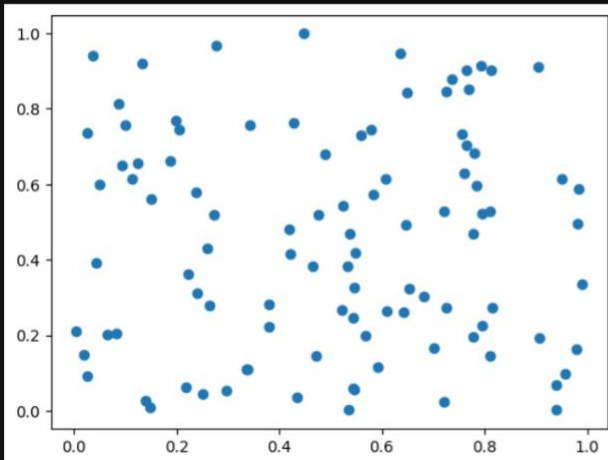


Lineal Function



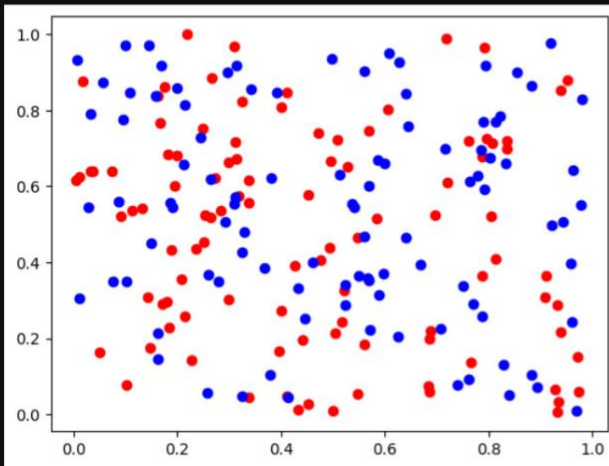
Scatter Plots

```
[22]: from numpy.random import rand
x, y = rand(2, 100)
plt.scatter(x, y)
plt.show()
```





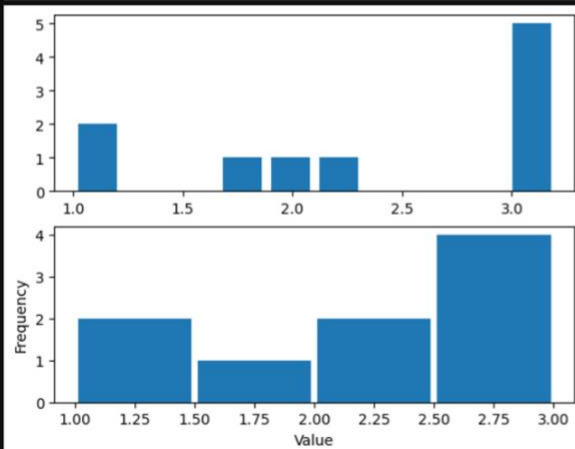
```
x, y = rand(2, 100)
x2, y2 = rand(2, 100)
plt.scatter(x, y, c = 'red')
plt.scatter(x2, y2, c = 'blue')
plt.show()
```



Histogramas

```
[26]: data = [1, 1.1, 1.8, 2, 2.1, 3.2, 3, 3, 3, 3]
plt.subplot(211)
plt.hist(data, bins = 10, rwidth = 0.8)

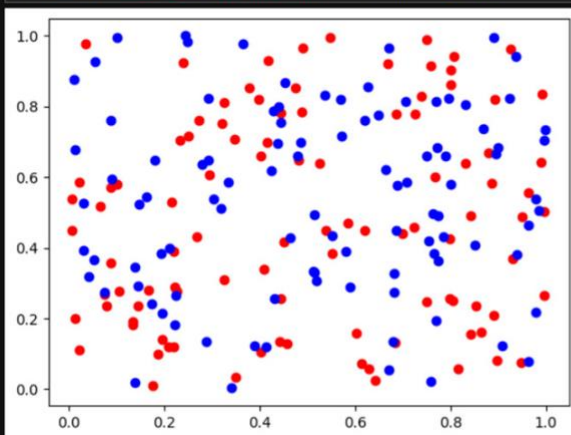
plt.subplot(212)
plt.hist(data, bins = [1, 1.5, 2, 2.5, 3], rwidth = 0.95)
plt.xlabel('Value')
plt.ylabel('Frequency')
plt.show()
```



Guardar las figuras

```
[28]: from numpy.random import rand
x, y = rand(2, 100)
x2, y2 = rand(2, 100)
plt.scatter(x, y, c = 'red')
plt.scatter(x2, y2, c = 'blue')

plt.savefig("3501_Mi_Grafica_Chida.png", transparent = True)
```



V. Conclusiones:

En esta práctica, hemos explorado las capacidades de Matplotlib para la visualización de datos dentro del entorno Anaconda. A lo largo del proceso, hemos aprendido a importar bibliotecas, crear gráficos básicos y personalizarlos para mejorar la presentación de la información.

Los resultados obtenidos demostraron que Matplotlib es una herramienta altamente versátil que permite generar visualizaciones informativas, desde gráficos de líneas y barras hasta diagramas de dispersión. La integración con Anaconda facilitó la gestión de dependencias y entornos, lo que garantizó un flujo de trabajo más eficiente.