Nombre de la práctica	Matplotlib			No.	3
Asignatura:	Simulación	Carrera:	Ingeniería en Sistemas Computacionales	Duración de la práctica (Hrs)	

NOMBRE DEL ALUMNO: Vanesa Hernández Martínez
GRUPO: 3501

II. Lugar de realización de la práctica (laboratorio, taller, aula u otro):
Actividades en aula de clases y en equipo personal

III. Material empleado:
- Laptop
- Anaconda

Matplotlib es una biblioteca de Python utilizada para la generación de gráficos estáticos, animados e interactivos. Su módulo más conocido, **Pyplot**, ofrece una interfaz sencilla para crear gráficos de líneas, barras, dispersión, histogramas, etc. La sintaxis de Pyplot es similar a la de MATLAB, lo que facilita su uso.

Características principales:
1. **Gráficos 2D**: Permite la creación de gráficos bidimensionales (líneas, dispersión, barras, histogramas, etc.).
2. **Personalización**: Puedes personalizar los ejes, leyendas, etiquetas, colores, estilos de línea y más.
3. **Soporte para LaTeX**: Posibilidad de integrar ecuaciones matemáticas en los gráficos usando sintaxis LaTeX.
4. **Interactividad**: Puedes agregar interactividad a tus gráficos en combinación con otras bibliotecas.
5. **Subgráficos**: Permite crear figuras con múltiples gráficos dentro, organizados en subtramas.

Introducción a Matplotlib

[Matplotlib] (https://matplotlib.org) 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 sencill y eficiente.
- Permite la integración de gráficos y figuras en un Jupyter Notebook.

Import

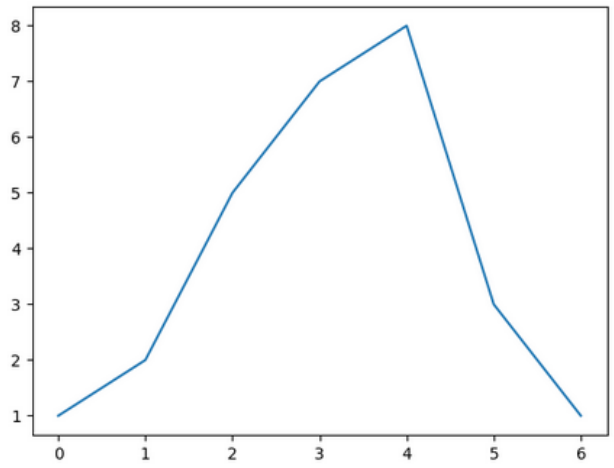
[3]:
```
import matplotlib
import matplotlib.pyplot as plt
```

[4]:
```
# Muestra los gráficos integrados dentro de Jupyter Notebook
%matplotlib inline
```

Representacion gráfica de los datos

Si a la función d etrazado se le va una matriz de datos, la usara como coordenadas en el eje vertical(y), y utilizara el índice de cada punto de datos en el array como la coordenada horizontal(x).

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

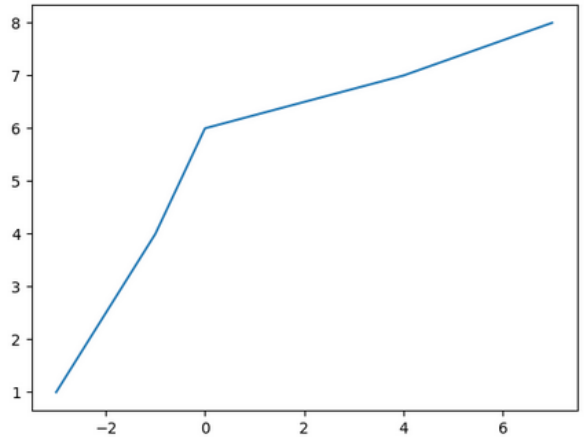


Tambien se pueden proporcionar dos matrices una para el eje horizontal y otra para el eje vertical.

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

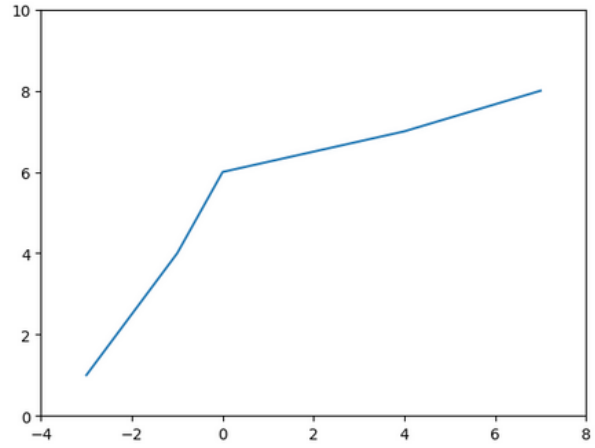
Tambien se pueden proporcionar dos matrices una para el eje horizontal y otra para el eje vertical.

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



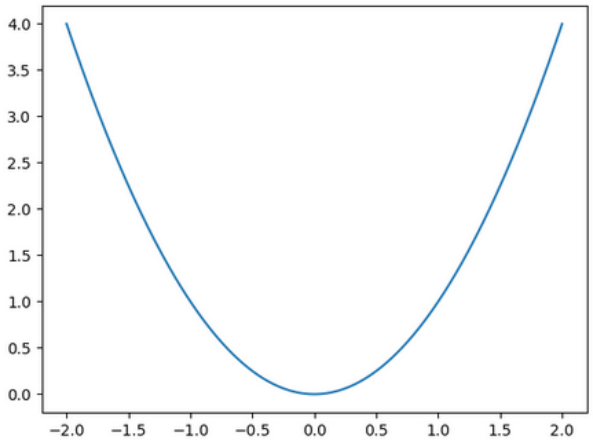
Pueden modificarse las longuitudes de los ejes para que la figura no se vea tan ajustada.

[9]:
```
plt.plot([-3, -1, 0, 4, 7], [1, 4, 6, 7, 8])
plt.axis([-4, 8, 0, 10]) #[xmin, xmax, ymin, ymax]
plt.show()
```



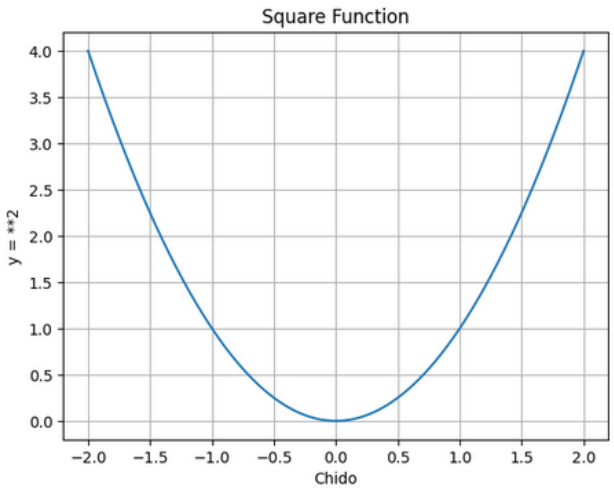
Se sigue el mismo procedimiento para pintar una función matemática.

[10]:
```
import numpy as np
x = np.linspace(-2, 2, 500)
y = x**2

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



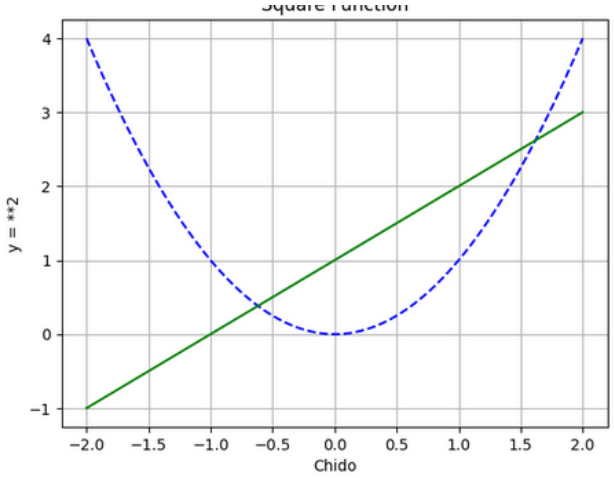
Tambien puede modificarse el estilo de la gráfica para que contenga más información.

[11]:
```
plt.plot(x, y)
plt.title("Square Function")
plt.xlabel("Chido")
plt.ylabel("y = **2")
plt.grid(True)
plt.show()
```

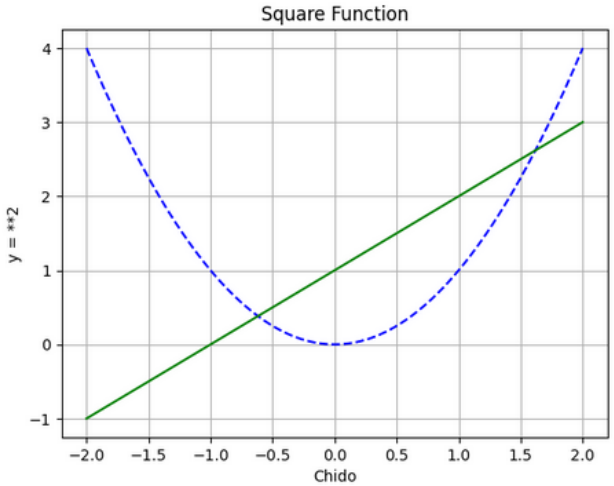


Pueden suporponerse gráficas y cambiar el estilo de las funciones.

[14]:
```
import numpy as np
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()
```

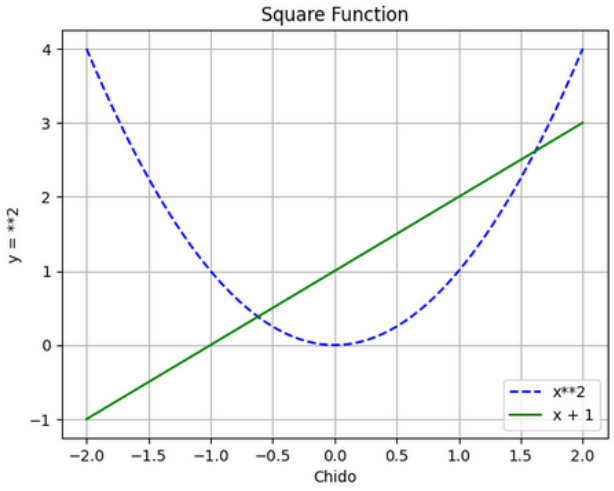


[15]:
```
# Separando en diferentes lineas las funciones.
import numpy as np
x = np.linspace(-2, 2, 500)
y = x**2
y2 = x + 1

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

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



[17]:
```
import numpy as np
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--',label = "x**2")
plt.plot(x, y2, 'g', label = "x + 1")
plt.legend(loc = "best") #Lo situa en la mejor localizacion
plt.show()
```

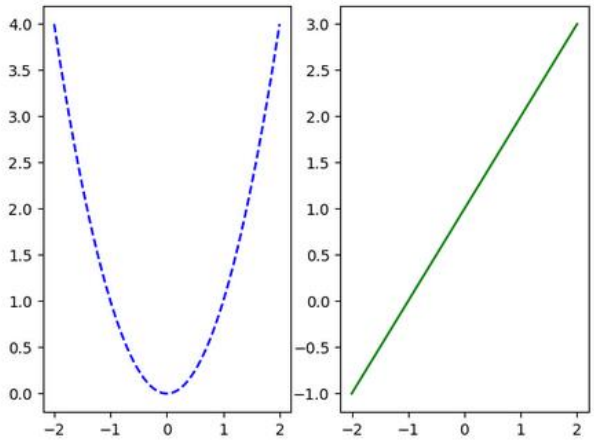
```
plt.show()
```



Tambien puede crearse dos gráficas que no se superpongan. Estas gráficas se organizan en un grid y se denominan **subplots**.

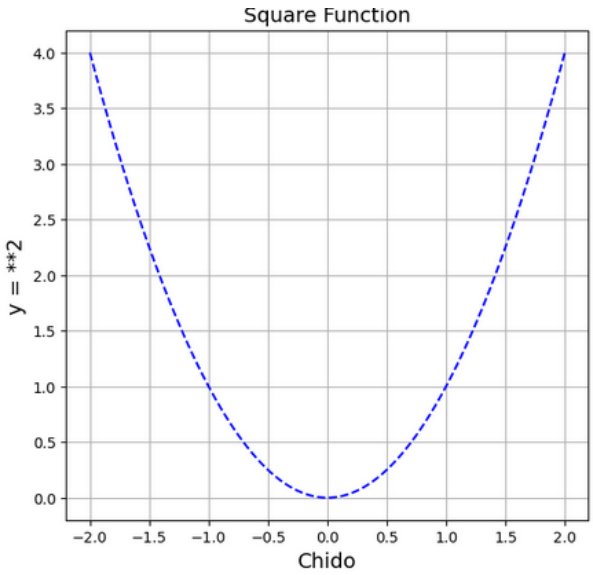
[18]:
```
import numpy as np
x = np.linspace(-2, 2, 500)
y = x**2
y2 = x + 1

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

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

plt.show()
```

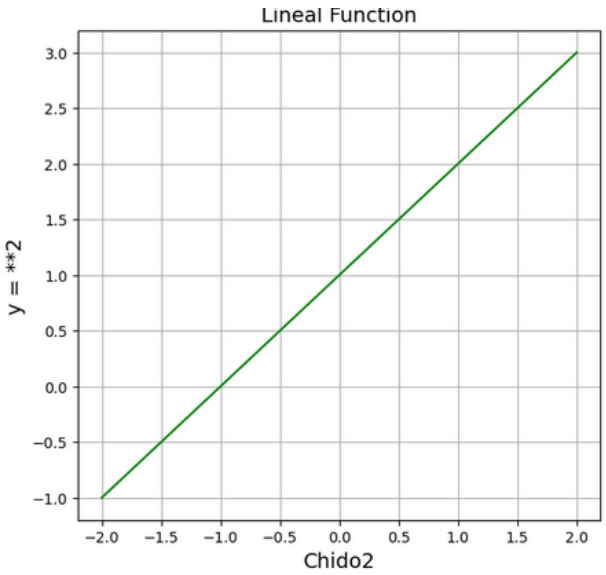
```
plt.show()
```



Para que las gráficas no queden tan agustadas se pueden hacer las gráficas más grandes.

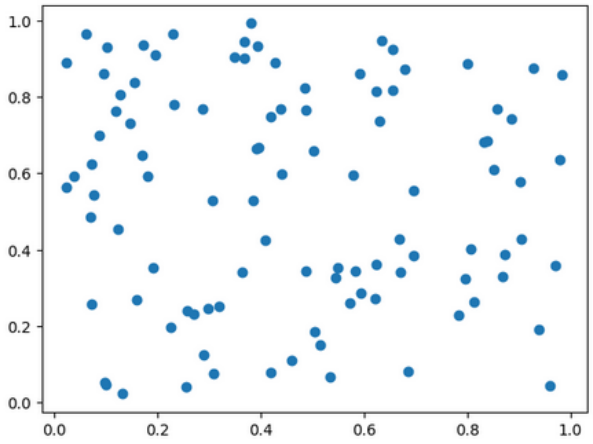
[20]:
```
plt.figure(figsize = (14, 6))

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(Fila), 2 Columns(Columna), 2nd Subplots
plt.plot(x, y2, 'g')
plt.title("Lineal Function", fontsize = 14)
plt.xlabel("Chido2", fontsize = 14)
plt.ylabel("y = **2", fontsize = 14)
plt.grid(True)

plt.show()
```



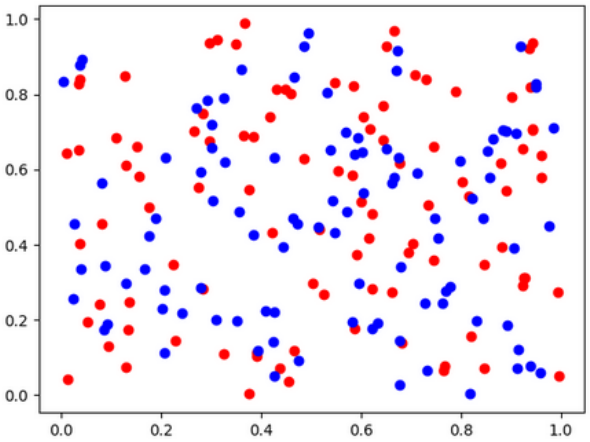


Scatter Plots

[21]:
```
from numpy.random import rand
x, y = rand(2, 100)
plt.scatter(x, y)
plt.show()
```

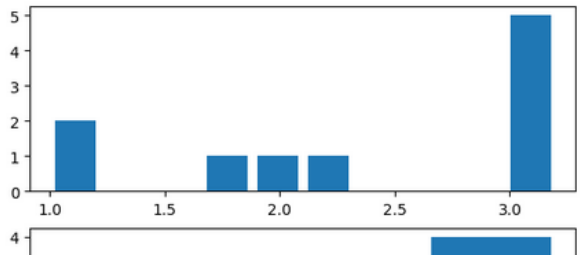


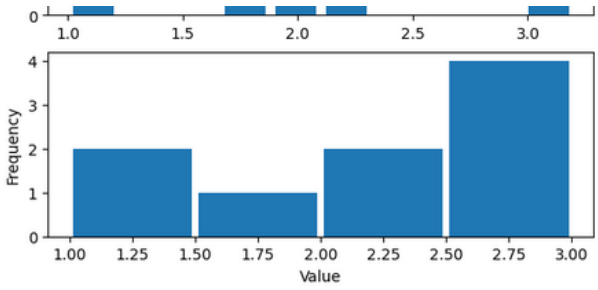
[22]:
```
from numpy.random import rand
x, y = rand(2, 100)
x2, y2 = rand(2, 100)
plt.scatter(x, y, c = 'red')
plt.scatter(x2, y2, c= 'blue')
plt.show()
```



Hostogramas

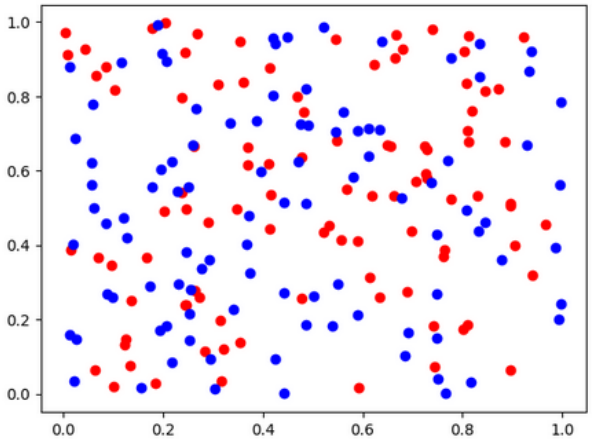
[23]:
```
data = [1, 1.1, 1.8, 2, 2.1, 3.2, 3, 3, 3, 3]
plt.subplot(211)
plt.hist(data, bins = 10, rwidth = 0.8)

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

[25]:
```
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)
```



[ ]:

- **Matplotlib** es una biblioteca de Python que facilita la creación de gráficos y figuras de calidad.
- Permite integrar gráficos fácilmente en **Jupyter Notebooks**.
- Para crear un gráfico básico, puedes proporcionar una matriz de datos que se usa como coordenadas en el eje **Y**, mientras que el índice del array se usa en el eje **X**.
- También es posible proporcionar dos matrices, una para el eje **X** y otra para el eje **Y**, personalizando más el gráfico.
- Se pueden ajustar las longitudes de los ejes para evitar que los gráficos queden ajustados.
- Es posible superponer varias gráficas en una sola figura y modificar el estilo para añadir más información visual.
- Para gráficos más complejos, se pueden crear múltiples gráficos organizados en un **grid** mediante **subplots**, evitando que se superpongan.
- El estilo y tamaño de los gráficos pueden ajustarse para mayor claridad.
- Tipos de gráficos comunes:

- **Scatter plots** (gráficos de dispersión).

- **Histogramas**.

- Las figuras creadas con Matplotlib se pueden guardar fácilmente en diferentes formatos.

Conclusión

Matplotlib es una herramienta muy útil para crear gráficos en Python de forma fácil. Nos permite hacer gráficos simples o más complejos, donde se pueden ver los resultados de forma interactiva. Se pueden personalizar los ejes y los estilos, además de superponer varios gráficos en uno solo o crear varios gráficos separados en una misma figura. También nos permite ajustar el tamaño de los gráficos para que no queden apretados. En general, es una biblioteca muy práctica para visualizar y entender datos en Python.