

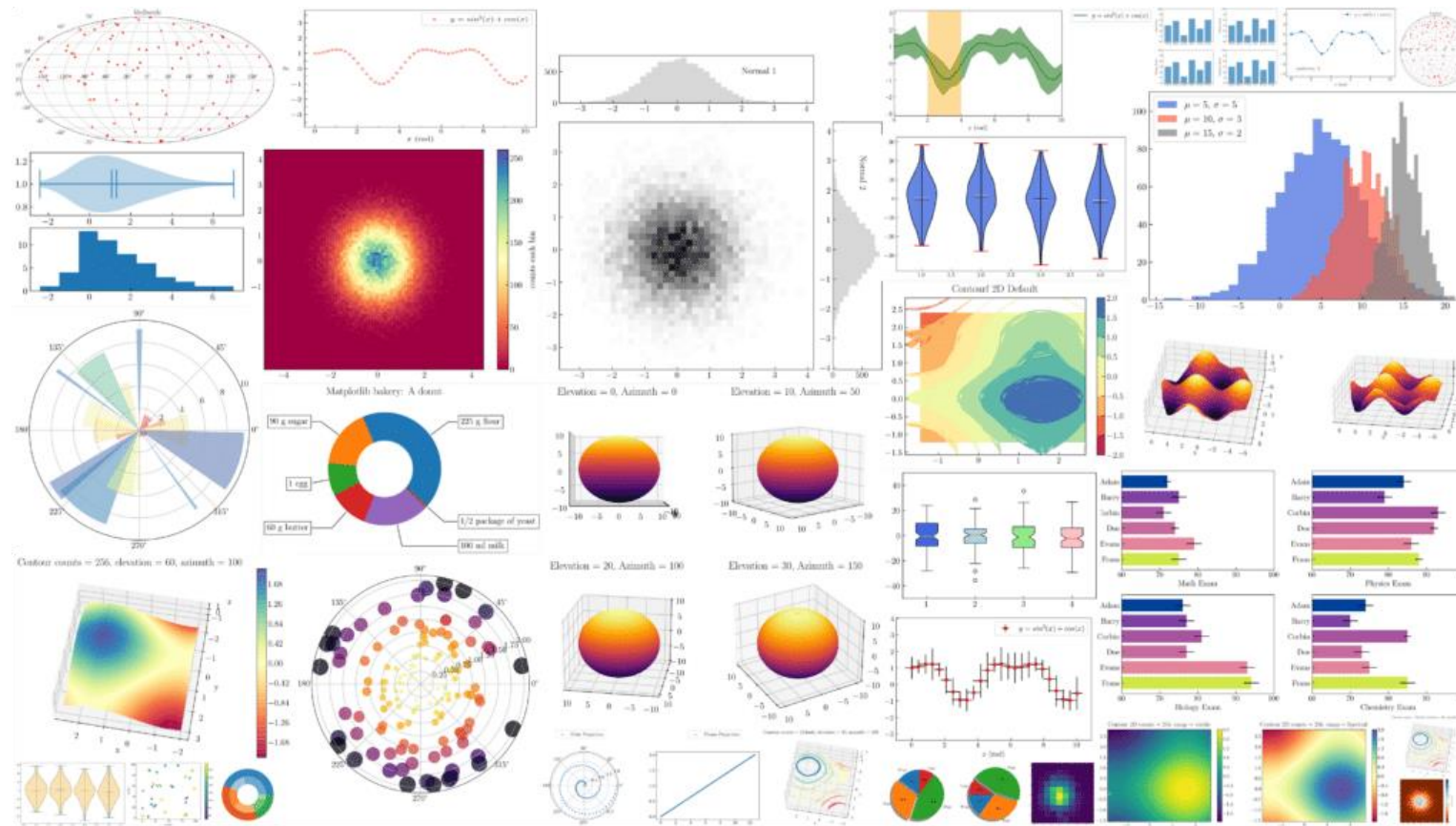
Graficas Modulo en python

POR: Guillermo Andres De Mendoza Corrales



Libreria

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



imports

```
import matplotlib.pyplot as plt  
import numpy as np
```



Graficas líneas

```
import matplotlib.pyplot as plt

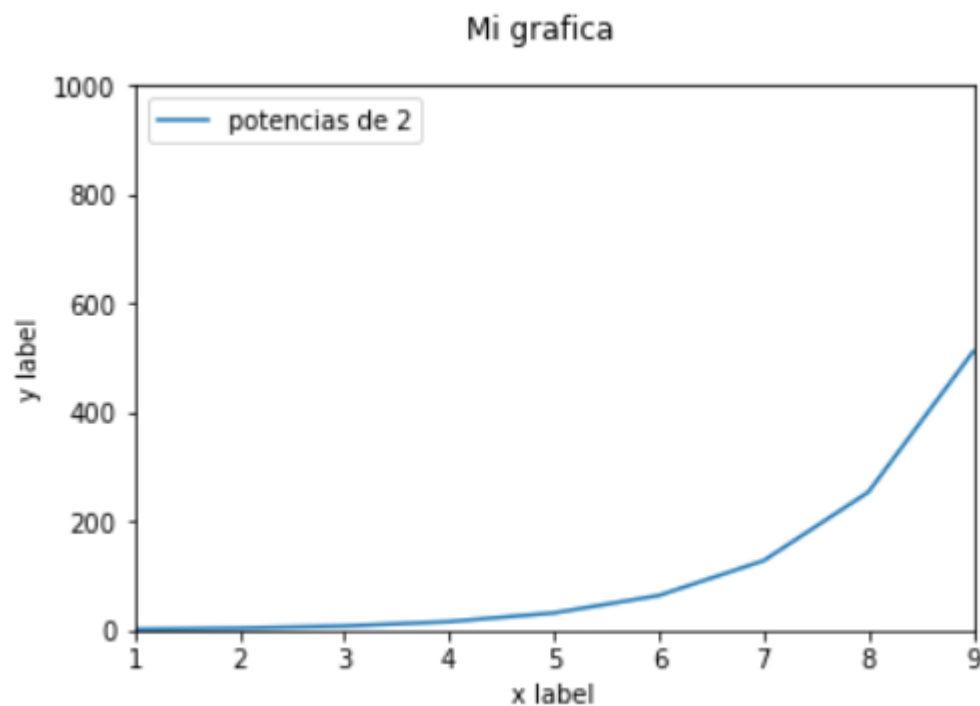
datosX = [1,2,3,4,5,6,7,8,9]
datosY = [2,4,8,16,32,64,128,256,512]

plt.suptitle('Mi grafica')
plt.ylabel('y label')
plt.xlabel('x label')
plt.legend("a")

plt.ylim(0, 1000)
plt.xlim(1, 9)

plt.plot(datosX,datosY,label="potencias de 2")
plt.legend(loc="upper left")

plt.show()
```

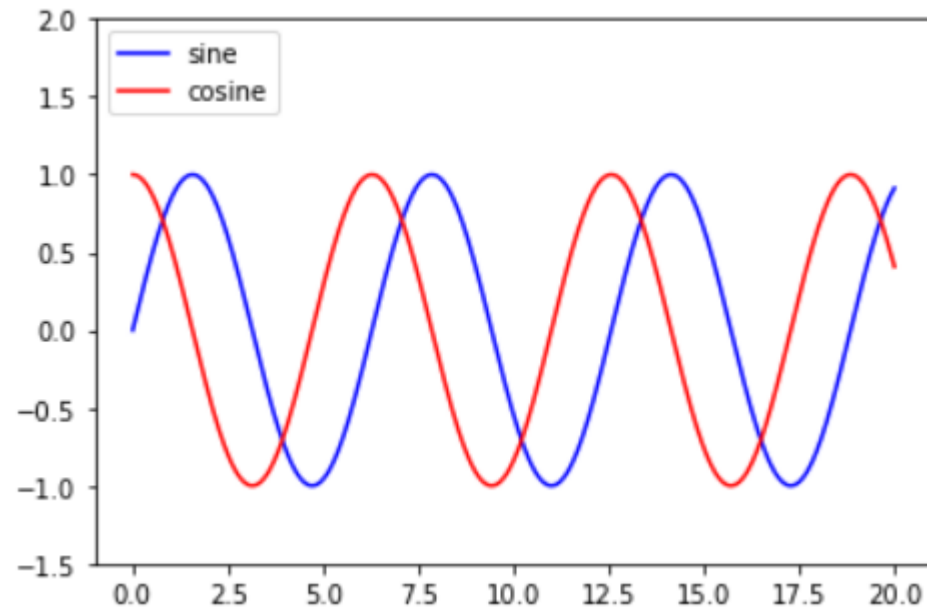


Graficas líneas multiples

```
import numpy as np
import matplotlib.pyplot as plt

x = np.linspace(0, 20, 1000)
y1 = np.sin(x)
y2 = np.cos(x)

plt.plot(x, y1, "-b", label="sine")
plt.plot(x, y2, "-r", label="cosine")
plt.legend(loc="upper left")
plt.ylim(-1.5, 2.0)
plt.show()
```

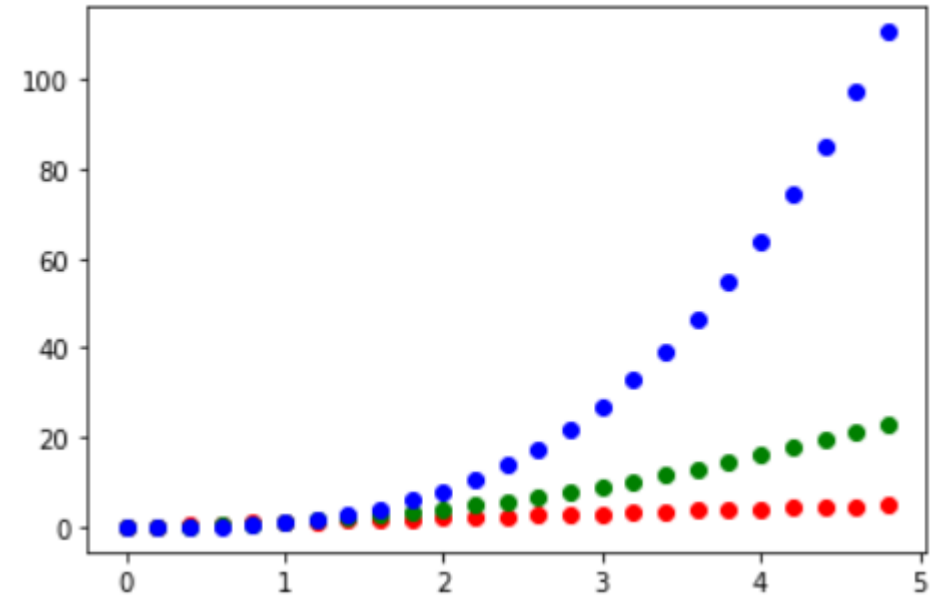


Graficas puntos

```
import numpy as np

# evenly sampled time at 200ms intervals
t = np.arange(0., 5., 0.2)

plt.scatter(t, t,color='r')
plt.scatter(t, t**2,color='g')
plt.scatter(t, t**3,color='b')
plt.show()
```

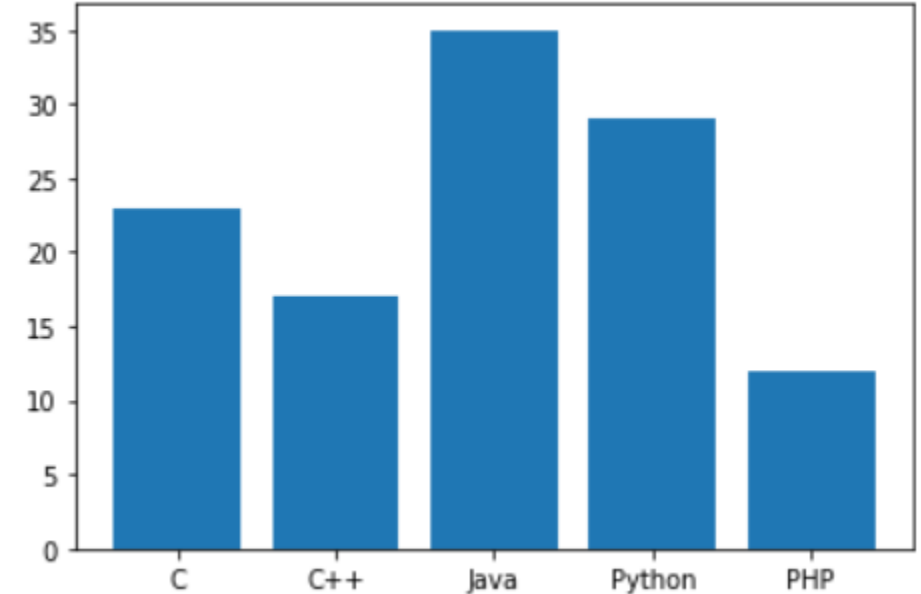


Barras verticales

```
import matplotlib.pyplot as plt

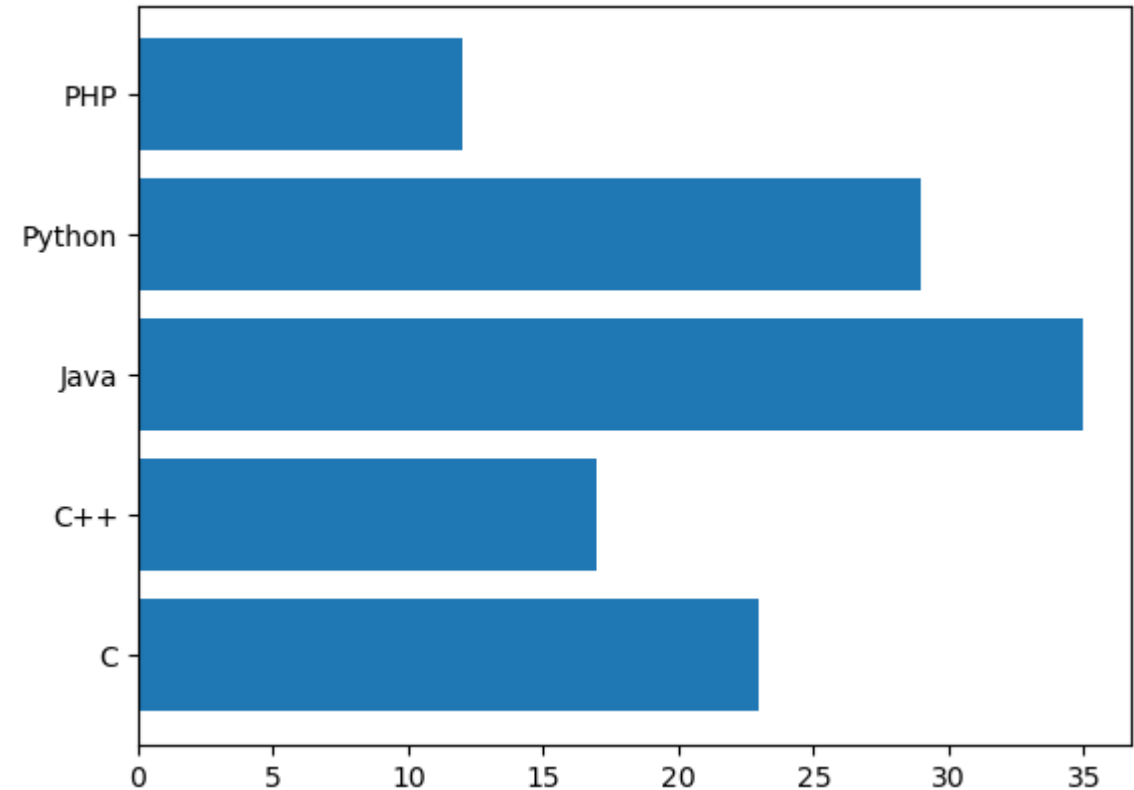
dataX = ['C', 'C++', 'Java', 'Python', 'PHP']
dataY = [23, 17, 35, 29, 12]

plt.bar(dataX, dataY)
plt.show()
```



Barras horizontales

```
dataX = ["C", "C++", "Java", "Python", "PHP"]  
dataY = [23, 17, 35, 29, 12]  
  
plt.barh(dataX, dataY)  
plt.show()
```



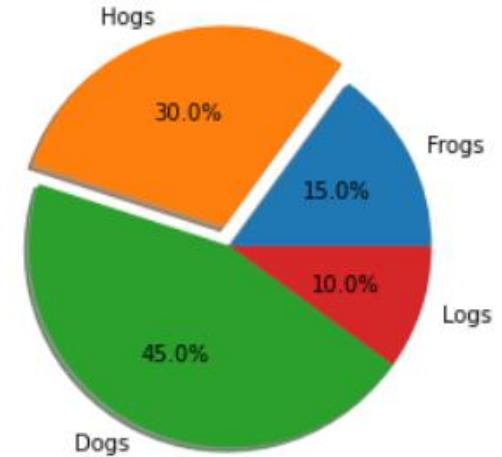
Graficas pastel

```
import matplotlib.pyplot as plt

labels = 'Frogs', 'Hogs', 'Dogs', 'Logs'
sizes = [15, 30, 45, 10]
explode = (0, 0.1, 0, 0) # only "explode" the 2nd slice (i.e. 'Hogs')

plt.pie(sizes, explode, labels, autopct='%1.1f%%', shadow=True)
plt.axis('equal') # Equal aspect ratio ensures that pie is drawn as a circle.

plt.show()
```

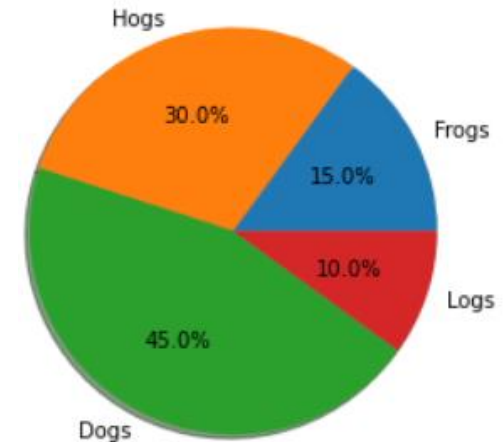


```
import matplotlib.pyplot as plt

labels = 'Frogs', 'Hogs', 'Dogs', 'Logs'
sizes = [15, 30, 45, 10]
explode = (0, 0, 0, 0) # only "explode" the 2nd slice (i.e. 'Hogs')

plt.pie(sizes, explode, labels, autopct='%1.1f%%', shadow=True)
plt.axis('equal') # Equal aspect ratio ensures that pie is drawn as a circle.

plt.show()
```

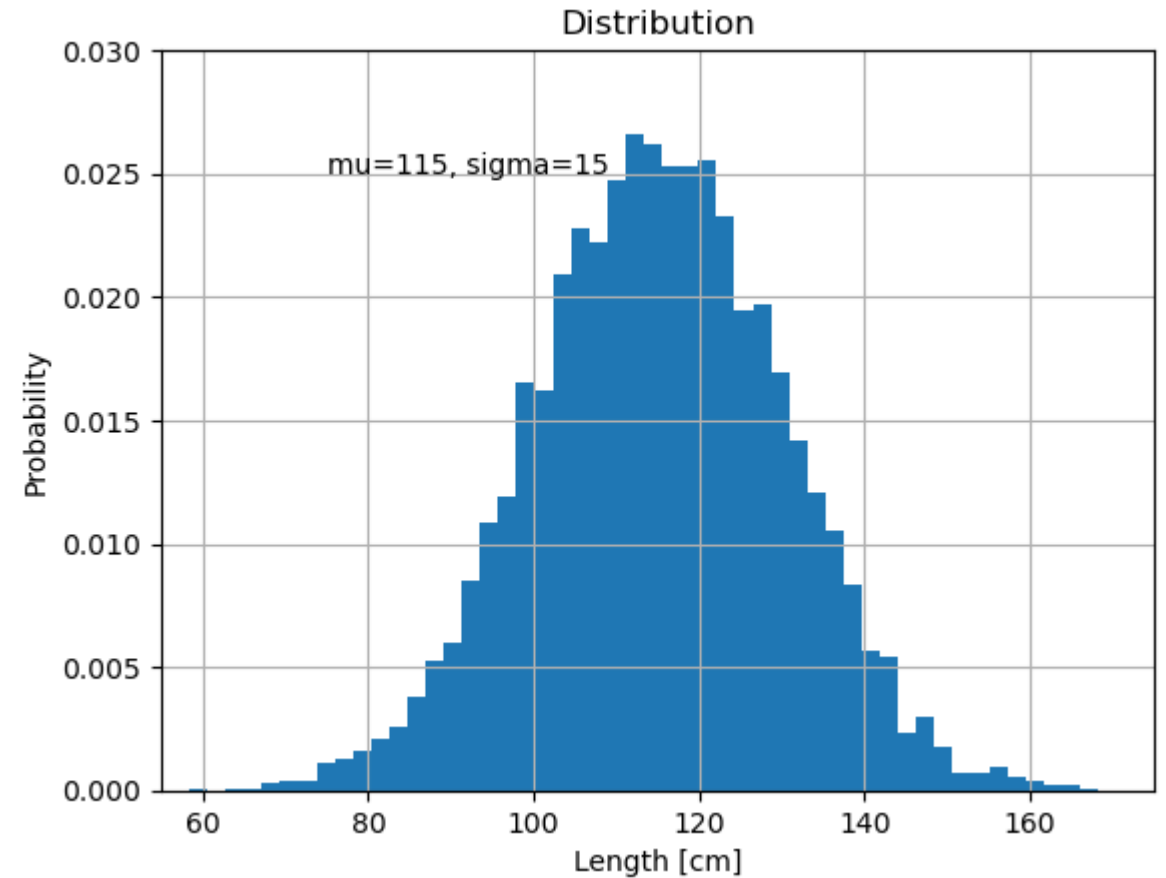


Histograma

```
#generate data
mu, sigma = 115, 15
x = mu + sigma * np.random.randn(10000)

# bins = periodo de muestras en x
# density = con True retorna el conteo
plt.hist(x, bins=50, density=True)
plt.xlabel('Length [cm]')
plt.ylabel('Probability')
plt.title('Distribution')
plt.text(75, .025, f'mu={mu}, sigma={sigma}')
plt.axis([55, 175, 0, 0.03])
plt.grid(True)

plt.plot()
```

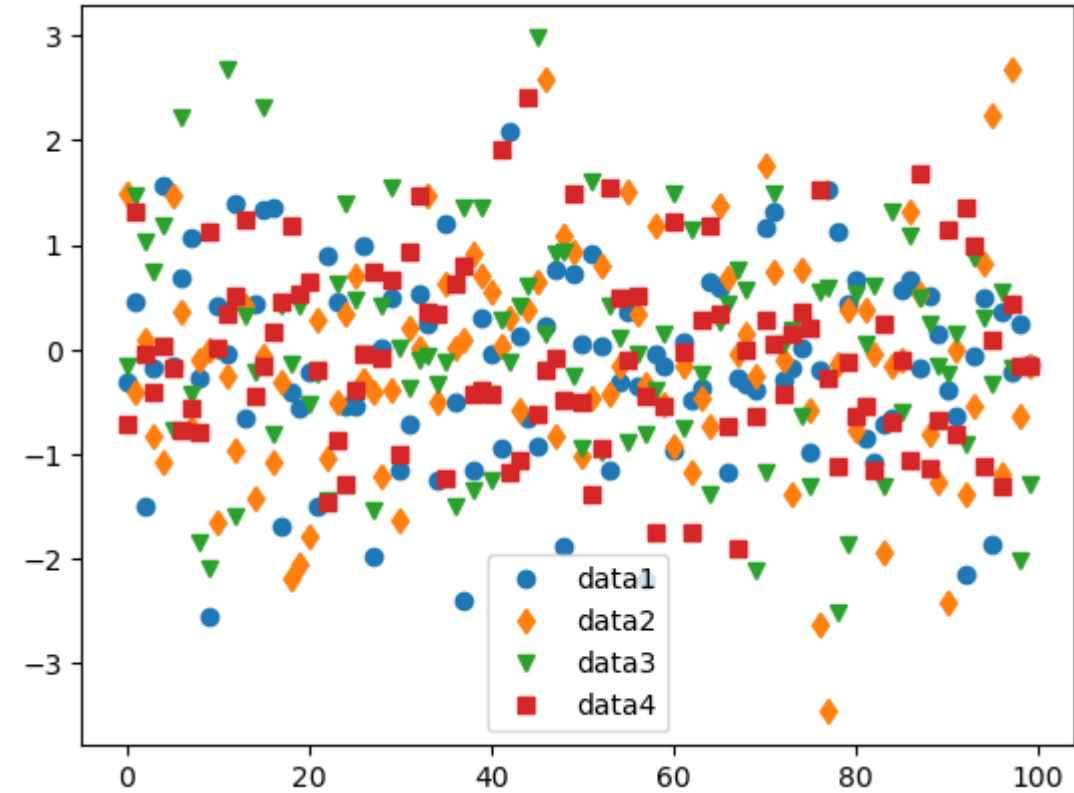


Simbolos

```
data1, data2, data3, data4 = np.random.randn(4, 100) # make 4 random data sets

plt.plot(data1, 'o', label='data1')
plt.plot(data2, 'd', label='data2')
plt.plot(data3, 'v', label='data3')
plt.plot(data4, 's', label='data4')
plt.legend()

plt.plot()
```



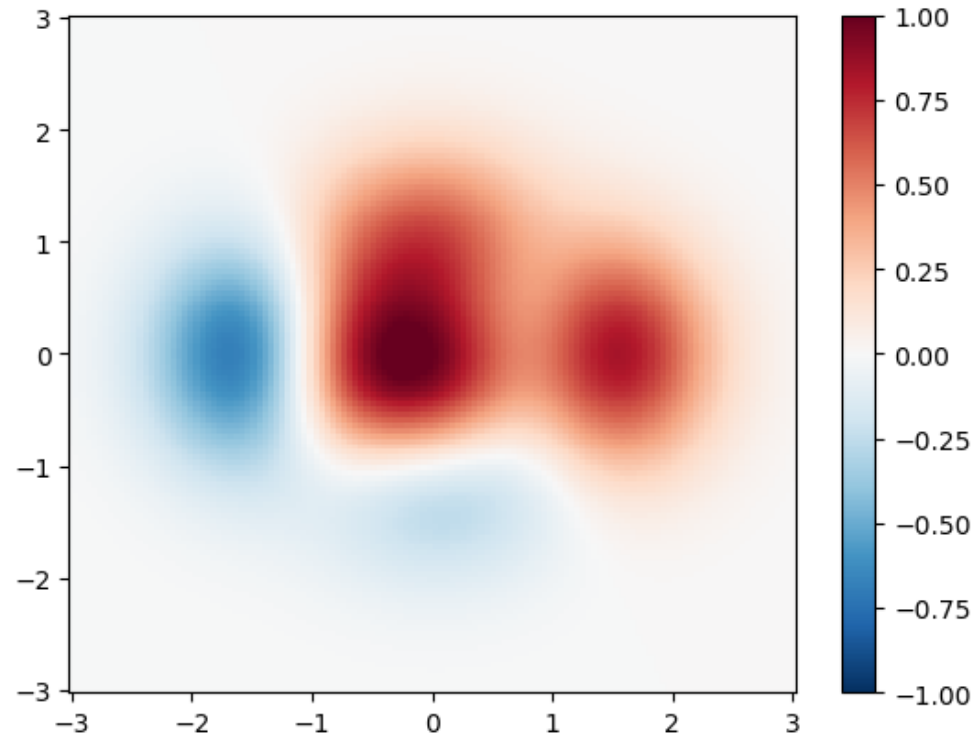
Grid: pcolormesh

```
import numpy as np
import matplotlib.pyplot as plt

from matplotlib.colors import LogNorm

X, Y = np.meshgrid(np.linspace(-3, 3, 128), np.linspace(-3, 3, 128))
Z = (1 - X/2 + X**5 + Y**3) * np.exp(-X**2 - Y**2) # se exageran los valores

# Create the RdBu_r
mygraph = plt.pcolormesh(X, Y, Z, vmin=-1, vmax=1, cmap='RdBu_r')
plt.colorbar(mygraph) # Add the colorbar
plt.show()
```



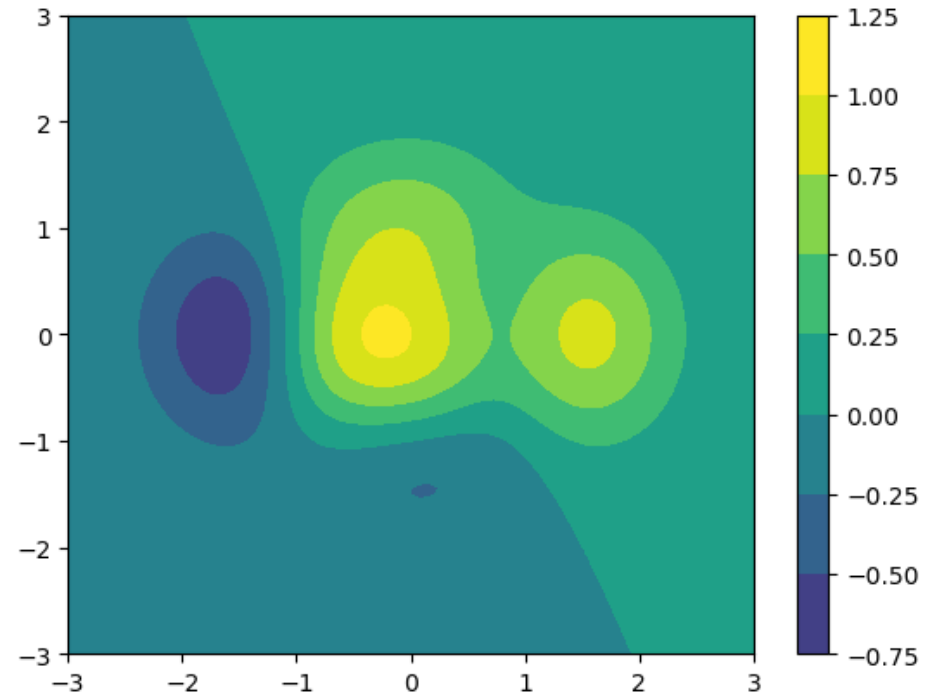
Grid: pcolormesh

```
import numpy as np
import matplotlib.pyplot as plt

from matplotlib.colors import LogNorm

X, Y = np.meshgrid(np.linspace(-3, 3, 128), np.linspace(-3, 3, 128))
Z = (1 - X/2 + X**5 + Y**3) * np.exp(-X**2 - Y**2) # se exageran los valores

# Create the RdBu_r
mygraph = plt.contourf(X, Y, Z, vmin=-1, vmax=1)
plt.colorbar(mygraph) # Add the colorbar
plt.show()
```



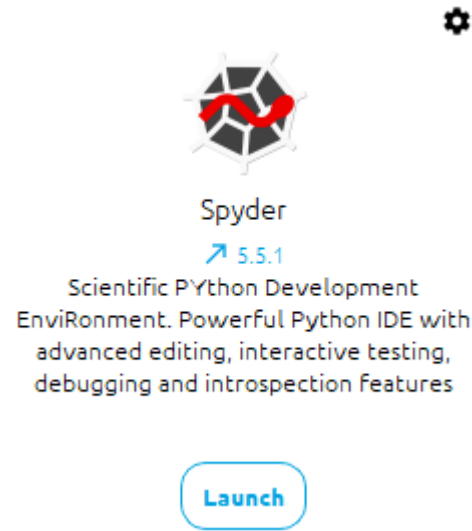
3D in Spider

Spider

Permite a los plots ser interactivos

SPIDER COMMAND:

`%matplotlib auto`



```
Console 1/A x
Python 3.12.7 | packaged by Anaconda, Inc. | (main, Oct 4 2024, 13:17:27) [MSC v.1929 64 bit (AMD64)]
Type "copyright", "credits" or "license" for more information.

IPython 8.27.0 -- An enhanced Interactive Python.

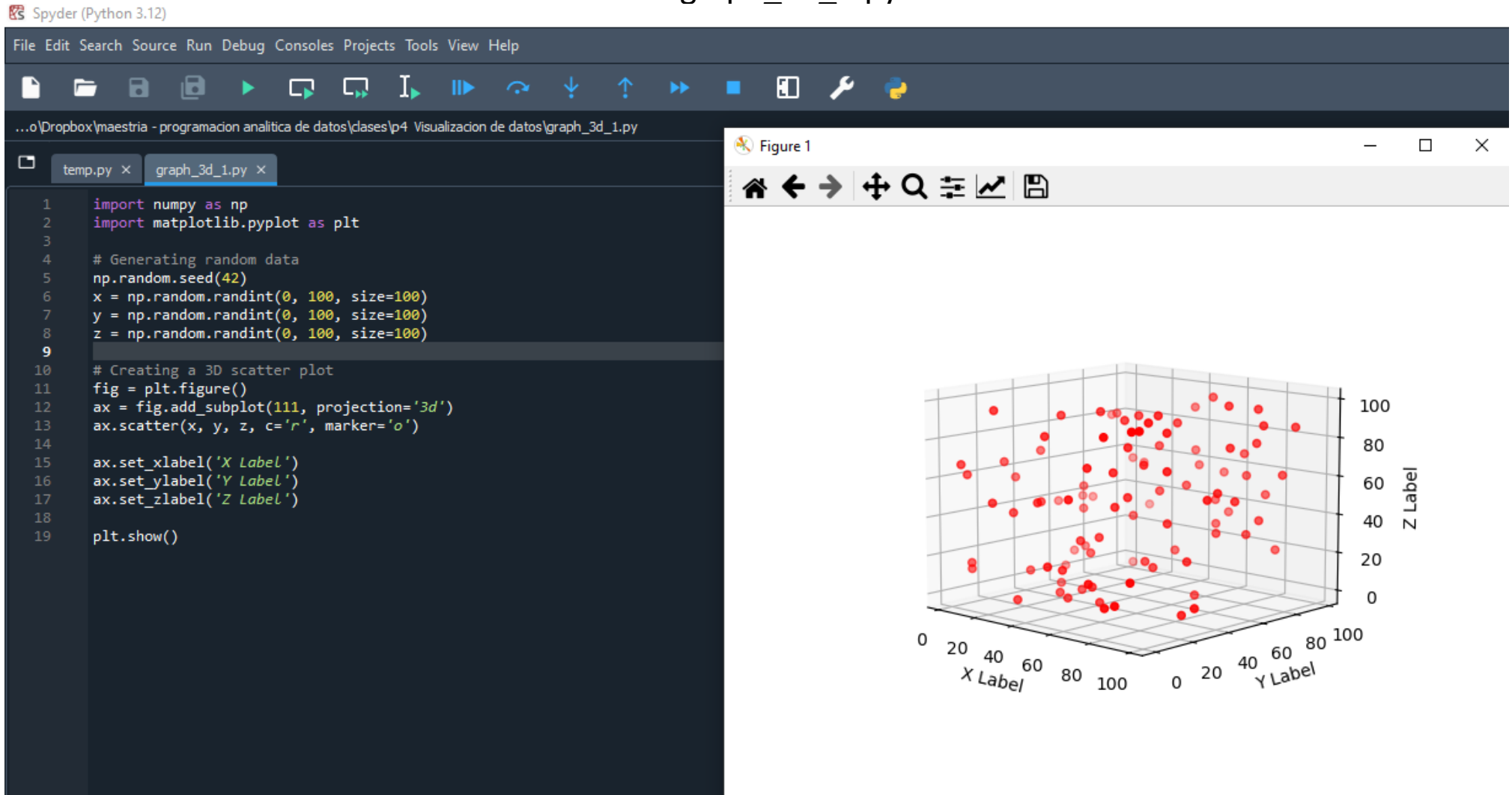
In [1]: %matplotlib auto
Using matplotlib backend: qtagg
Using matplotlib backend: qtagg

In [2]:
```



3D Graphs

File: graph_3d_1.py



3D Graphs

File: graph_3d_2.py

Spyder (Python 3.12)

File Edit Search Source Run Debug Consoles Projects Tools View Help



...o\Dropbox\maestria - programacion analitica de datos\clases\p4 Visualizacion de datos\graph_3d_2.py

temp.py x graph_3d_2.py x

```
1 import numpy as np
2 import matplotlib.pyplot as plt
3
4 # Data for a three-dimensional line
5 zline = np.linspace(0, 15, 1000)
6 xline = np.sin(zline)
7 yline = np.cos(zline)
8
9 # Data for three-dimensional scattered points
10 zdata = 15 * np.random.random(100)
11 xdata = np.sin(zdata) + 0.1 * np.random.randn(100)
12 ydata = np.cos(zdata) + 0.1 * np.random.randn(100)
13
14 # Create a 3D axis
15 fig = plt.figure()
16 ax = fig.add_subplot(111, projection='3d')
17
18 # Plot the 3D line
19 ax.plot3D(xline, yline, zline, 'gray')
20
21 # Scatter the 3D points
22 sc = ax.scatter3D(xdata, ydata, zdata, c=zdata, cmap='viridis') #
23
24 fig.colorbar(sc) # Add the colorbar
25
26 # Set the labels for the axes
27 ax.set_xlabel("X")
28 ax.set_ylabel("Y")
29 ax.set_zlabel("Z")
30
31 # Show the plot
32 plt.show()
```

Figure 2

