Gráficos con Matplotlib

Análisis Numérico/Análisis Numérico I - FAMAF ‘21

Matplotlib

- Librería Esencial para gráficos en Python
- La mayoría de las librerías de visualización del lenguaje están construidas en Matplotlib.
- Basada en el graficador de MATLAB.

Arrancamos con lo básico, en terminal de Python

```
import numpy as np

import matplotlib.pyplot as plt

plt.plot([1, 2, 3, 4], [1, 4, 2, 3])

plt.show()
```

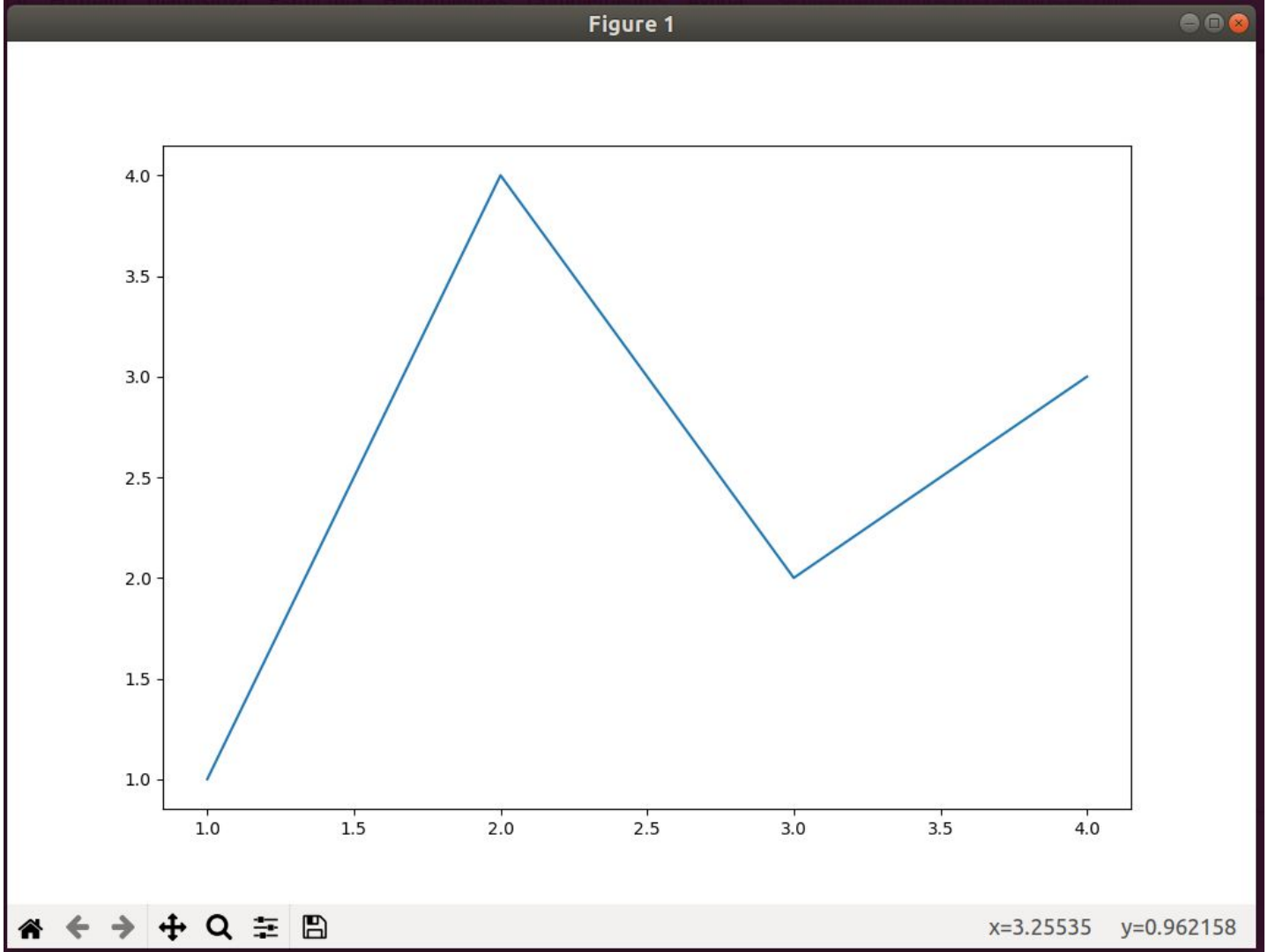
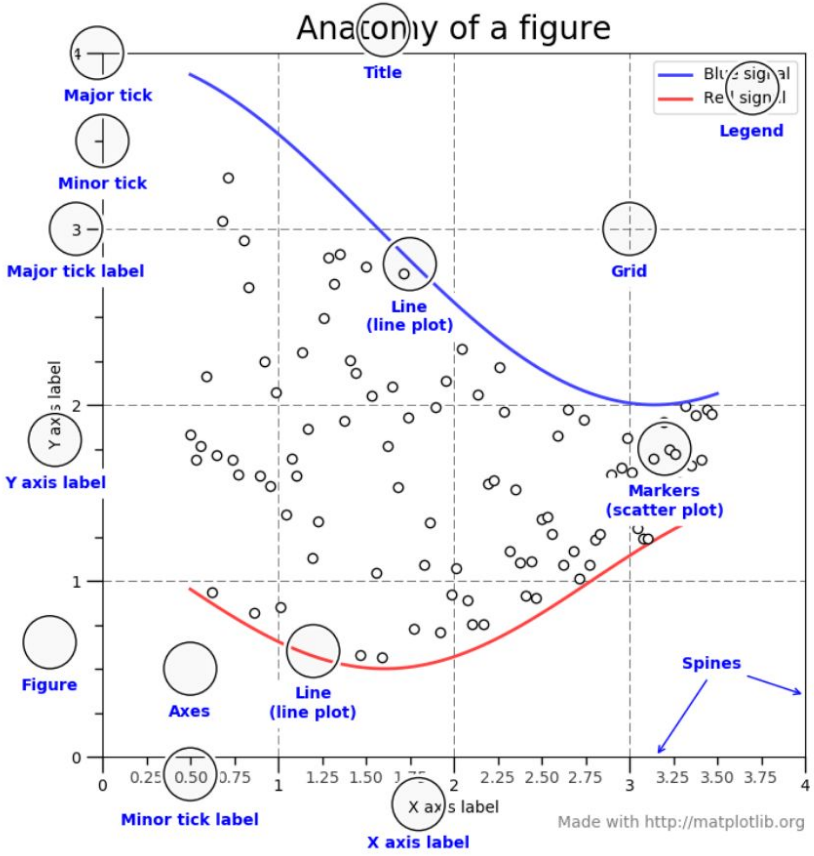


Figure 1
4.0
3.5
3.0
2.5
2.0
1.5
1.0
1.0
1.5
2.0
2.5
3.0
3.5
4.0
x=3.25535
y=0.962158

Partes del gráfico



Partes del gráfico

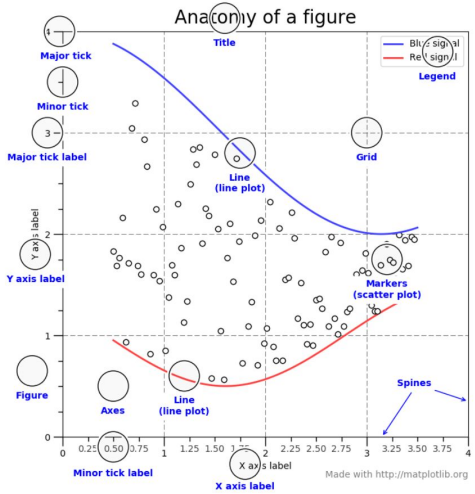


Figure:

- Lo que envuelve a todo el gráfico (todo lo que será visible)
- Para crearla y manipularla, deberiamos hacer

```
fig = plt.figure()
```

Axes:

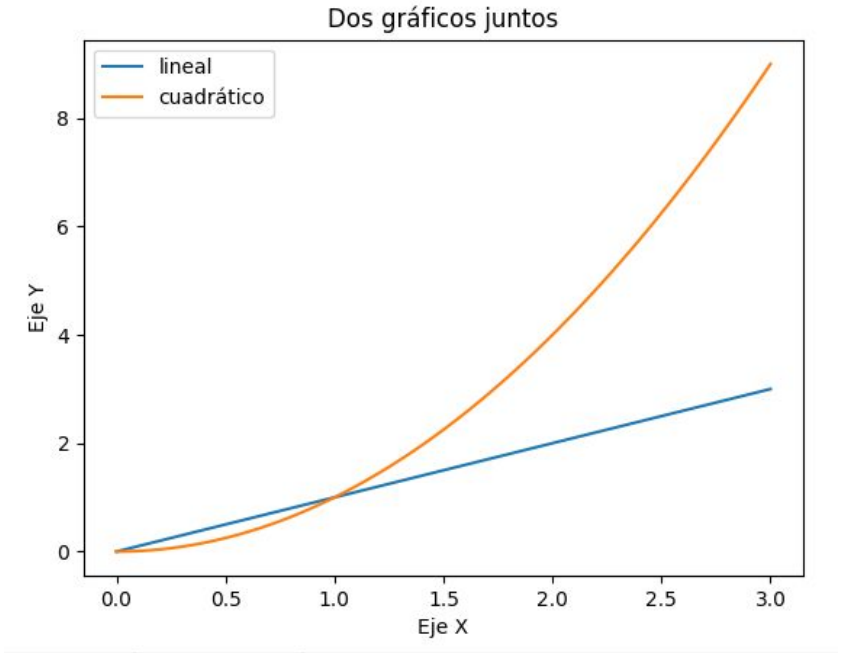
- Los límites de un gráfico. Es posible generar más de uno en una misma figura con subplots.

```
fig, ax = plt.subplots()
```

Dos formas de graficar

Crear explícitamente los objetos y llamar los métodos dentro de ellos (“funciones internas”). Se lo conoce como el “estilo orientado por objetos”.

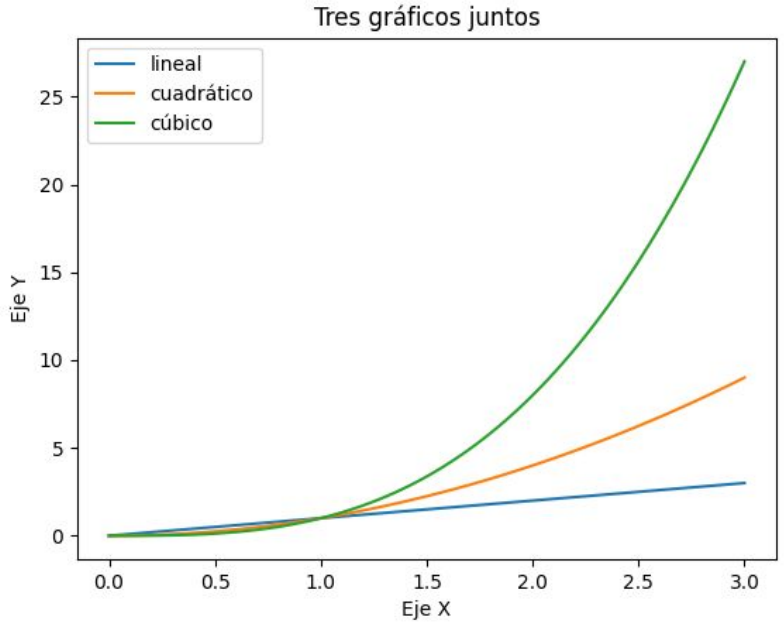
```
x = np.linspace(0, 3, 120)

fig, ax = plt.subplots()

ax.plot(x, x, label="lineal")

ax.plot(x, x**2, label="cuadrático")

ax.set_xlabel("Eje X")

ax.set_ylabel("Eje Y")

ax.set_title("Dos gráficos juntos")

ax.legend()

plt.show()
```



Dos formas de graficar

El estilo “py-plot” (un poco más vago, pero re-práctico para un sólo gráfico)

```
x = np.linspace(0, 3, 120)
plt.plot(x, x, label="lineal")
plt.plot(x, x**2, label="cuadrático")
plt.plot(x, x**3, label="cúbico")
plt.xlabel("Eje X")
plt.ylabel("Eje Y")
plt.title("Tres gráficos juntos")
plt.legend()
plt.show()
```



Qué más podemos agregar en la sentencia de plot?

- Varios gráficos en uno solo

```
plt.plot(x, x, x, x**2);plt.legend([“lineal”, “cuadrático”])
```

- Marcadores que quieran (estrellas, círculos, puntos, triángulos), están en https://matplotlib.org/3.2.1/api/markers_api.html

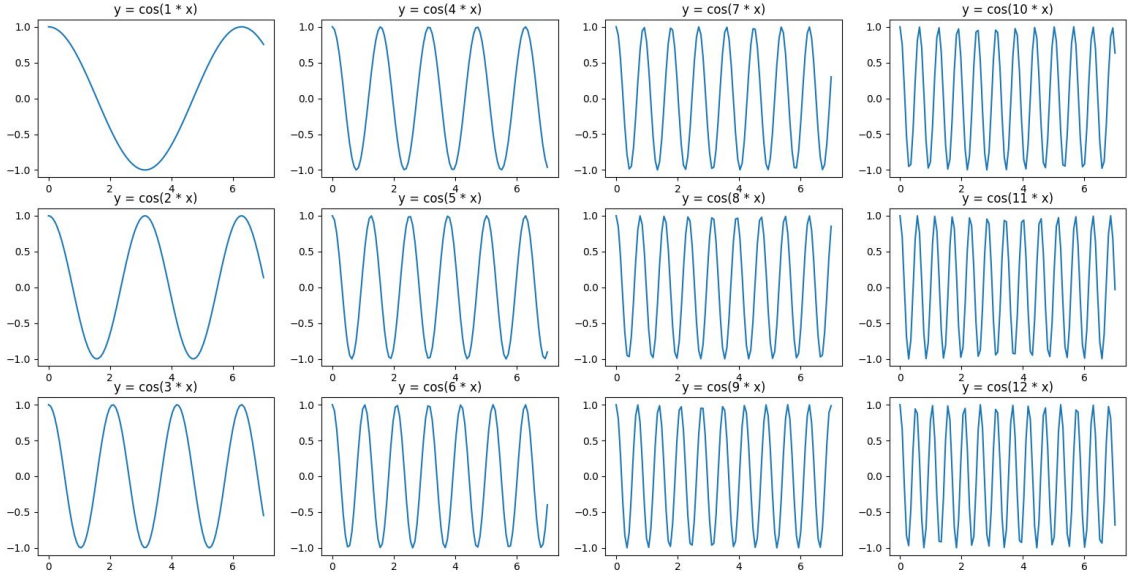
```
plt.plot(x, x, “*”, x, x**2, “-.”)
```

- Colores de las líneas (se puede escribir el nombre en inglés o código)

```
plt.plot(x, x, “red”, x, x**2, “g”)
```

- Todas las combinaciones (siempre y cuando estén bien organizadas)

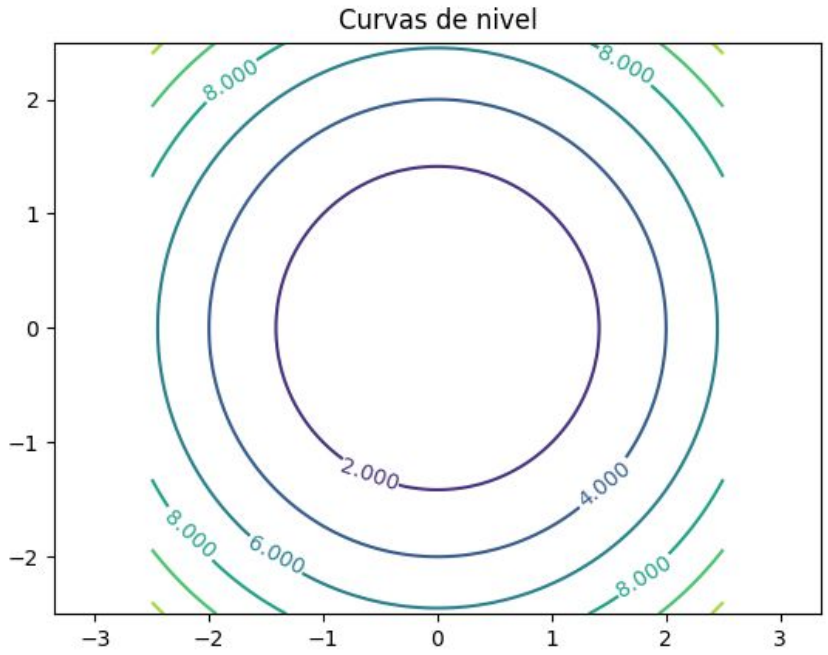
Gráficos múltiples




```
x = np.linspace(0, 7, 200)
fig, ax = plt.subplots(3, 4)
for idx in range(3):
    for idy in range(4):
        ax[idx][idy].plot(x, np.cos((idy * 3 + idx + 1) * x))
        ax[idx][idy].set_title("y = cos({} * x)".format(idy * 3 + idx + 1))
fig.show()
```

Curvas de nivel (para análisis multivariado dim 3)

```
x = np.linspace(-2.5, 2.5, 200)
y = np.linspace(-2.5, 2.5, 200)
xx, yy = np.meshgrid(x, y)
zz = xx ** 2 + yy ** 2
fig, ax = plt.subplots()
ax.axis('equal')  # ejes iguales
CS = ax.contour(xx, yy, zz, levels=[2,4,6,8])
ax.clabel(CS, inline=1, fontsize=10)
ax.set_title('Curvas de nivel')
fig.show()
```



Y mucho (muuuchoooo) más