Introducción a Python

Temario de la Clase 5

Graficación usando matplotlib

- Gráficos de curvas
- Labels - títulos - líneas - colores
- subplots
- guardado de gráficos
- Barplots
- Scatterplots
- Histogramas - densidades
- Contornos - imágenes - flujos
- Animaciones

Graficación en python

La librería de graficación se llama matplotlib. Dentro de esta tenemos dos opciones: pylab y pyplot.

- pylab. Interface de graficación inspirada en Matlab.
- pyplot. Librería de graficación basada en objetos.

pylab trabaja con procedimientos e importaba todo al namespace (no recomendado).
Son sublibrerías de matplotlib:

```
import matplotlib.pyplot as plt
```

Graficación con scripts

plot es para graficar curvas 2D.

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

x = np.linspace(0, 20, 1000)
y = np.sin(x)

plt.plot(x, y)
plt.savefig('fig.png')

plt.show() # Solo en la laptop NO en  servidores
os.system('okular fig.png&') # muestro la imagen desde el script en Linux
# !okular fig.png  # muestro la imagen en ipython
```

savefig para guardar la figura en un archivo (png,jpq,eps,etc.)

show para mostrar la figura. OJO NO usar en el servidor

Usar el show solo al final de los scripts (consume recursos).

Guardar el gráfico en un archivo

Para guardar la figura en un archivo de imagen tenemos el comando:
savefig(fname, dpi=None, facecolor='w', edgecolor='w',
orientation='portrait', papertype=None, format=None):
Ejemplo:

```
savefig('fig05.png', dpi=80)
```

Formatos recomendados: png o eps (conservan la resolución de fuentes y líneas cuando se cambia el tamaño).

Esta es la única opción en servidores remotos (en lugar del show/inline).

Graficación en jupyter

En las versiones viejas se solía poner explicitamente que haga los gráficos embebidos en el cuaderno:

```
[1] %matplotlib inline
```

Actualmente se puede graficar directamente (lo tiene por default):

```
[2] import numpy as np
    import matplotlib.pyplot as
                        plt

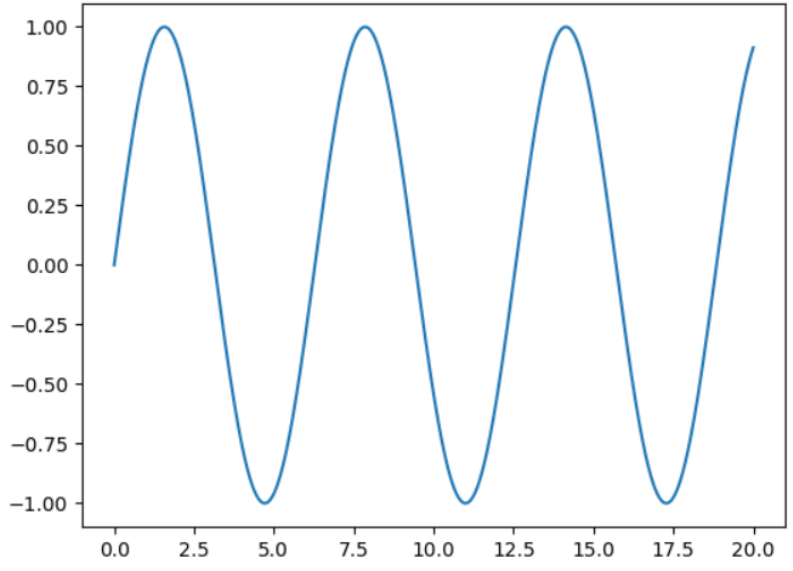
    x = np.linspace(0, 20, 1000)
    y = np.sin(x)

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

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

     x = np.linspace(0, 20, 1000)
     y = np.sin(x)

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




```
[ ]:
```

Algunos atajos en jupyter para una vida no tan miserable

[Esc] Pasa de verde (modo edicion) a azul la celda (modo comando).
[Enter] Pasa de azul a verde. Util para editar la celda.
[Shift] + [Enter] para ejecutar la celda corriente
[Ctrl]+ [Enter] para ejecutar las celdas seleccionadas.

En modo comando:

a genera celda arriba
b genera celda abajo
m celda con markdown
d + d borra celda
x corta las celdas seleccionadas
c copia las celdas seleccionadas
v pastea las celdas debajo
Ctrl + Home primera celda
Ctrl + End última celda

En modo de edición:

Tab completa el comando o tabula
Ctrl+] tabula
Ctrl+[destabula
Ctrl + Z deshace
Ctrl + Y vuelve a hacer
Ctrl + Home va al inicio celda
Ctrl + End va a lo ultimo celda

Ctrl + s Guarda el cuaderno (checkpoint)

Markdown mínimo

Tanto el jupyter/colab como el zulip utilizan markdown (procesado de textos ascii)

```
#  Titulos
## Secciones
### Subsecciones
Negritas: **Nota** __Nota__
Italicas: *Recordar* _Recordar_

Lista de items:
* Item
  * Subitem
- Item
  - Subitem

Agregado de imagenes:
![Funcion seno](img/plot_seno.png)
Links:
[GitHub](http://github.com)
```

````
Codigo en el texto `plt.plot`

Bloque de codigo:
``` codigos ```
Con resaltado de comandos:
```python
if a==0:
    print('termine')
```
Formulas matematicas:
\begin{equation}
\alpha=\int f(x) dx
 \end{equation}
$\alpha=\int f(x) dx$
````
````

Títulos de gráficos

xlabel('x'), ylabel('y'), title('Titulo')

```
plt.plot(x, y)

plt.xlabel('this is x!')
plt.ylabel('this is y!')
plt.title('My First Plot')
```

Entiende latex para escribir fórmulas:

```
y = np.sin(2 * np.pi * x)
plt.plot(x, y)
plt.title(r'$\sin(2 \pi x)$')
```

Tipos de curvas/líneas

```
plt.plot(x, y, '-r')
```

Colores disponibles:

'r' = red - rojo

'g' = green - verde

'b' = blue - azul

'c' = cyan - celeste

'm' = magenta - violeta

'y' = yellow - amarillo

'k' = black - negro

'w' = white - blanco

Las líneas pueden tener distintos estilos:

'-' = línea continua

'–' = línea a trazos

':' = línea punteada

'-.' = punteada a trazos

'.' = puntos

'o' = círculos

'^' = triángulos