Matplotlib: plotting

[Matplotlib](http://matplotlib.org/) is probably the most used Python package for 2D-graphics. It provides both a quick way to visualize data from Python and publication-quality figures in many formats.

**from** **matplotlib** **import** pyplot **as** plt

**Simple Plot:**

**Draw a Cosine and Sine Function on the plot**:

**import** **numpy** **as** **np**

**import** **matplotlib.pyplot** **as** **plt**

X = np.linspace(-np.pi, np.pi, 256, endpoint=**True**)

C, S = np.cos(X), np.sin(X)

plt.plot(X, C)

plt.plot(X, S)

plt.show()

Instantiating defaults

**import** **numpy** **as** **np**

**import** **matplotlib.pyplot** **as** **plt**

*# Create a figure of size 8x6 inches, 80 dots per inch*

plt.figure(figsize=(8, 6), dpi=80)

*# Create a new subplot from a grid of 1x1*

plt.subplot(1, 1, 1)

X = np.linspace(-np.pi, np.pi, 256, endpoint=**True**)

C, S = np.cos(X), np.sin(X)

*# Plot cosine with a blue continuous line of width 1 (pixels)*

plt.plot(X, C, color="blue", linewidth=1.0, linestyle="-")

*# Plot sine with a green continuous line of width 1 (pixels)*

plt.plot(X, S, color="green", linewidth=1.0, linestyle="-")

*# Set x limits*

plt.xlim(-4.0, 4.0)

*# Set x ticks*

plt.xticks(np.linspace(-4, 4, 9, endpoint=**True**))

*# Set y limits*

plt.ylim(-1.0, 1.0)

*# Set y ticks*

plt.yticks(np.linspace(-1, 1, 5, endpoint=**True**))

*# Save figure using 72 dots per inch*

*# plt.savefig("exercise\_2.png", dpi=72)*

*# Show result on screen*

plt.show()

### Changing colors and line widths

* ...
* plt.figure(figsize=(10, 6), dpi=80)
* plt.plot(X, C, color="blue", linewidth=2.5, linestyle="-")
* plt.plot(X, S, color="red", linewidth=2.5, linestyle="-")

### Setting limits

* ...
* plt.xlim(X.min() \* 1.1, X.max() \* 1.1)
* plt.ylim(C.min() \* 1.1, C.max() \* 1.1)

### Setting ticks

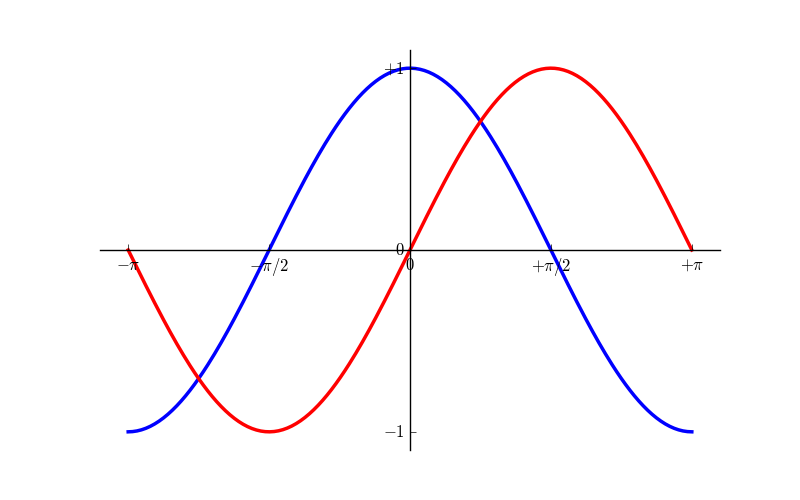
* plt.xticks([-np.pi, -np.pi/2, 0, np.pi/2, np.pi])
* plt.yticks([-1, 0, +1])

### Setting tick labels

* ...
* plt.xticks([-np.pi, -np.pi/2, 0, np.pi/2, np.pi],
* [r'$-\pi$', r'$-\pi/2$', r'$0$', r'$+\pi/2$', r'$+\pi$'])
* plt.yticks([-1, 0, +1],
* [r'$-1$', r'$0$', r'$+1$'])
* ...

### Moving spines

Spines are the lines connecting the axis tick marks and noting the boundaries of the data area. They can be placed at arbitrary positions and until now, they were on the border of the axis.



### Adding a legend

Let’s add a legend in the upper left corner. This only requires adding the keyword argument label (that will be used in the legend box) to the plot commands.

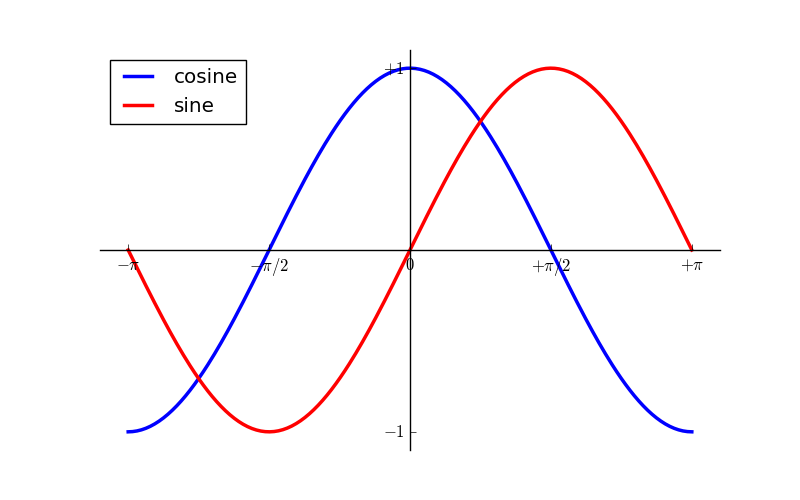
...

plt.plot(X, C, color="blue", linewidth=2.5, linestyle="-", label="cosine")

plt.plot(X, S, color="red", linewidth=2.5, linestyle="-", label="sine")

plt.legend(loc='upper left')

...



Figures, Subplots

A **“figure”** in matplotlib means the whole window in the user interface. Within this figure there can be **“subplots”.**

**Subplot(m,n,p)**

**m**: number of rows

**n**:number of columns

**p**:it determines where you want to place your plot within the grid, it increases from 1 to m\*n from left to right and top to bottom**.**

* **First create subplot and then call plot- when we call plt.show() that plot will be shown on the subplot grid only , see below Example and diagram**

**>>> plt.subplot(2,2,1)**

**<matplotlib.axes.\_subplots.AxesSubplot object at 0x000000FD4A330E48>**

**>>> plt.plot(X,S)**

**[<matplotlib.lines.Line2D object at 0x000000FD4A356D68>]**

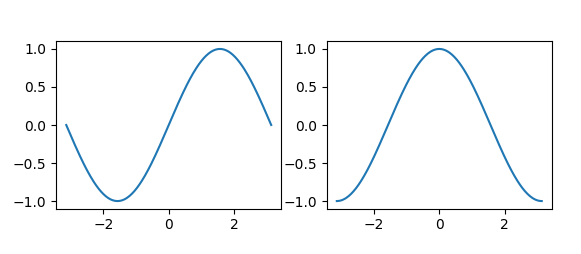
**>>> plt.subplot(2,2,2)**

**<matplotlib.axes.\_subplots.AxesSubplot object at 0x000000FD4A525A58>**

**>>> plt.plot(X,C)**

**[<matplotlib.lines.Line2D object at 0x000000FD4A52D4A8>]**

**>>> plt.show()**

****

Other types of plots:

1. **Regular plots – as shown above (plt.plot)**
2. **Scatter plots –**

n = 1024

X = np.random.normal(0,1,n)

Y = np.random.normal(0,1,n)

plt.scatter(X,Y)

1. **Bar plots:**
2. **Pie Charts:**
3. Z = np.random.uniform(0, 1, 20)
4. plt.pie(Z)

**References:** [**http://www.scipy-lectures.org/intro/matplotlib/index.html**](http://www.scipy-lectures.org/intro/matplotlib/index.html)