Tutorial 3: Introduction to Matplotlib

EE2405

嵌入式系統與實驗

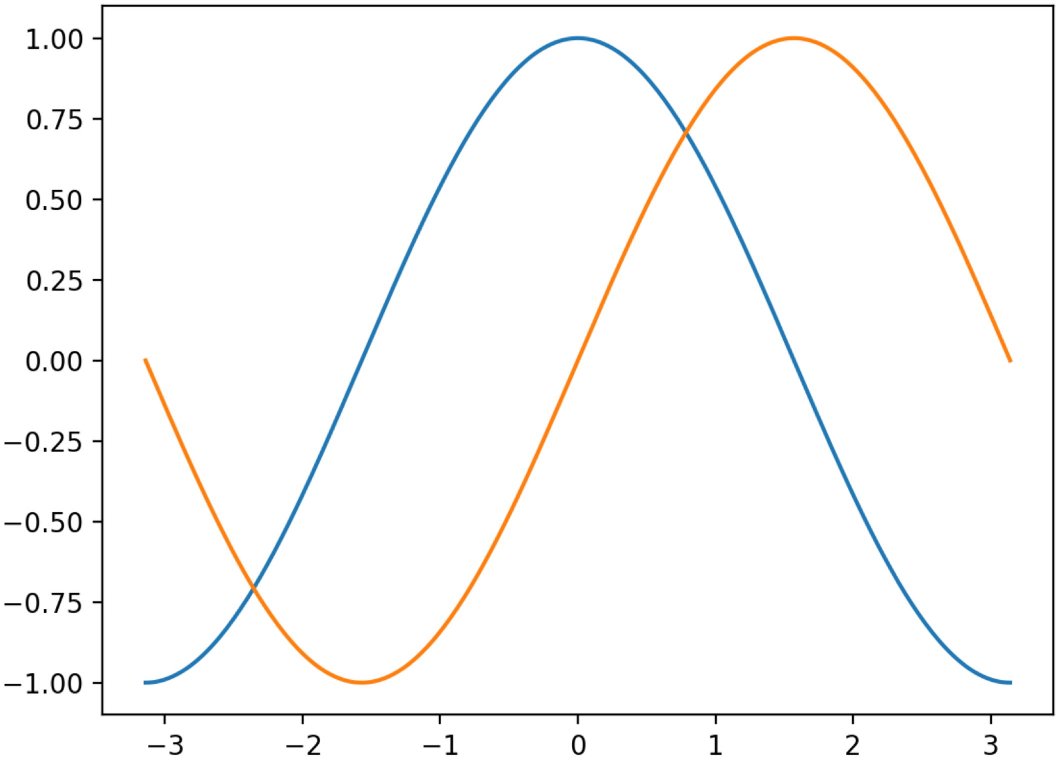
Embedded System Lab

Introduction

- Matplolib
  - A Python plotting library, inspired by MATLAB
  - To visualize scientific data in 2D
- Pyplot
  - PyPlot is a shell-like interface to Matplotlib
  - Pyplot maintains state across calls.

Exercise 1

```
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()
```


1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
−3
−2
−1
0
1
2
3

Notes on exercise_1.py

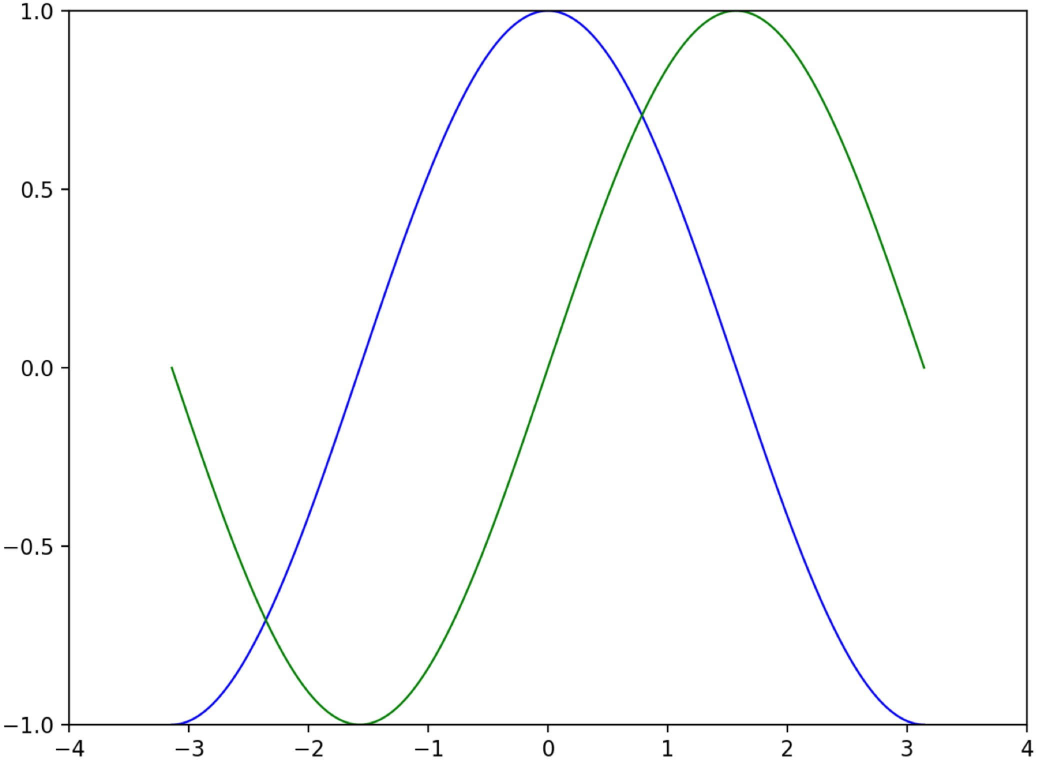
- **np.linspace(-np.pi, np.pi, 256, endpoint=True)**: linspace() will generate a list of numbers equally spaces between min and max. In this example, min=-np.pi and max=np.pi and number=256.
- **C = np.cos(X)**: cos() will generate a list of cosine values (C) from an input of list of numbers (X).
- **plt.plot(X,C)**: plot(X, C) will plot every pairs of C value vs. X value in the plt object. Note that you may plot several times on a plot object before actually show the figure on screen with show().

Exercise 2 (1/2)

```
# Create a new figure of size 8x6 points, using 100 dots per inch
plt.figure(figsize=(8,6), dpi=100)

# Create a new subplot from a grid of 1x1
plt.subplot(111)

X = np.linspace(-np.pi, np.pi, 256,endpoint=True)
C,S = np.cos(X), np.sin(X)

# Plot cosine using blue color with a continuous line of width 1 (pixels)
plt.plot(X, C, color="blue", linewidth=1.0, linestyle="-")

# Plot sine using green color with a continuous line of width 1 (pixels)
plt.plot(X, S, color="green", linewidth=1.0, linestyle="-")
```

Exercise 2 (2/2)

```
# 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))
```

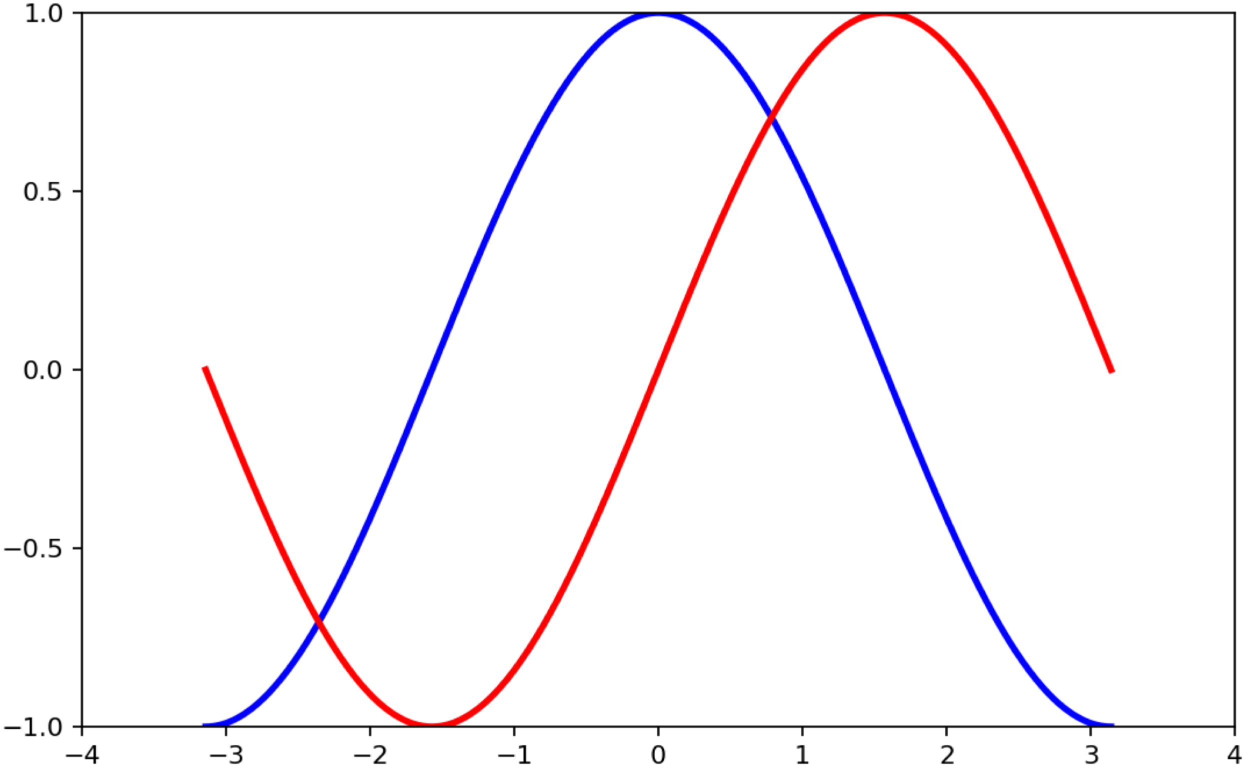

1.0
0.5
0.0
−0.5
−1.0
−4
−3
−2
−1
0
1
2
3
4

Notes on subplots

- **plt.subplot(111)**: "111" denotes the first subfigure (1) in (row, col)=(1, 1) sub-figures. As another example, "121" denotes the first subfigure in a side-by-side sub-figures (1X2). For more discussion, please refer to https://stackoverflow.com/questions/3584805/in-matplotlib-what-does-the-argument-mean-in-fig-add-subplot111

Exercise 3

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

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

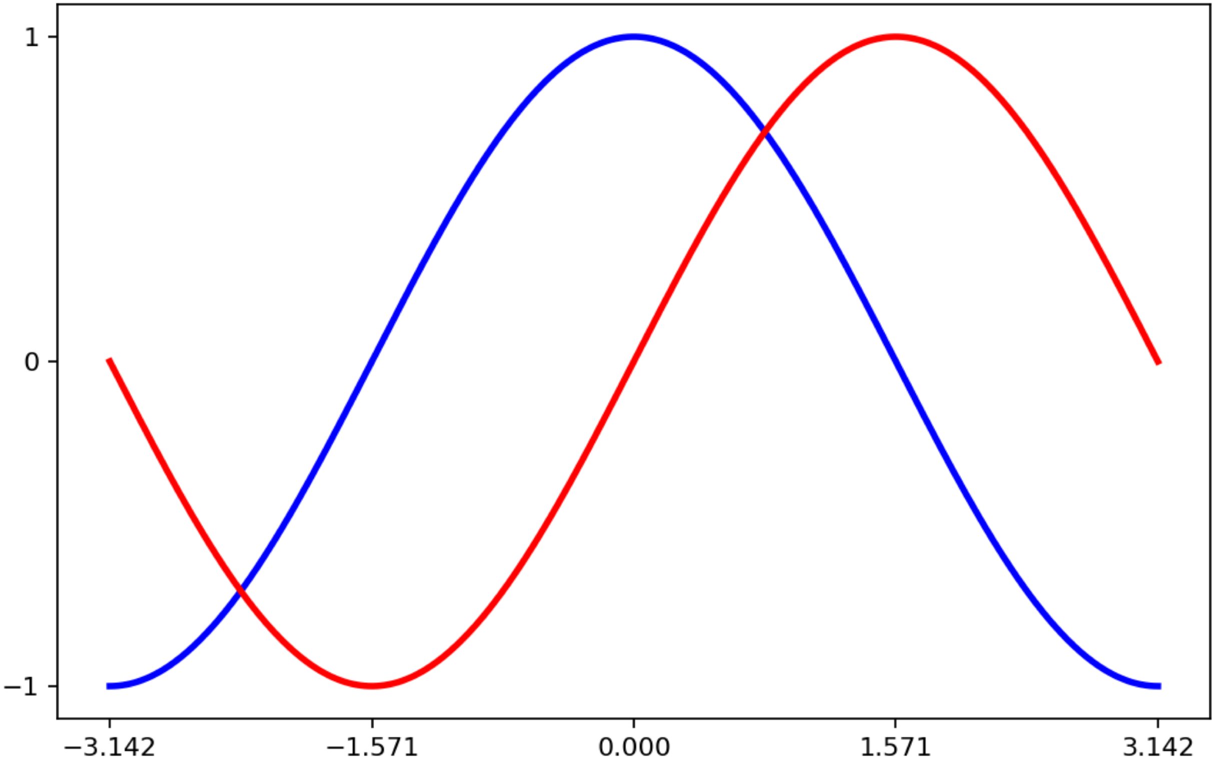

1.0
0.5
0.0
−0.5
−1.0
−4
−3
−2
−1
0
1
2
3
4

Exercise 4

```
plt.xlim(X.min()*1.1, X.max()*1.1)

plt.ylim(C.min()*1.1,C.max()*1.1)
```

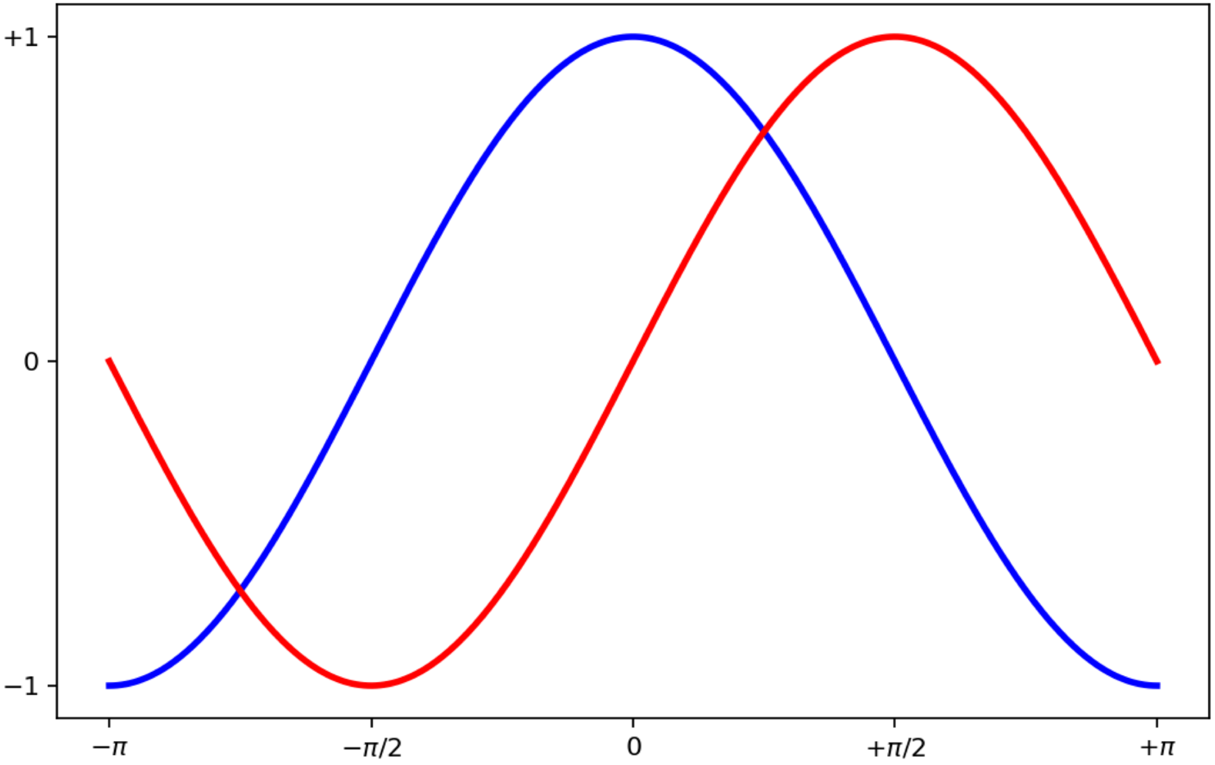
Exercise 5

```
# Set x ticks at interesting points
plt.xticks([-np.pi, -np.pi/2, 0, np.pi/2,
np.pi])

plt.yticks([-1, 0, +1])
```


1
0
−1
−3.142
−1.571
0.000
1.571
3.142

Exercise 6

```
# Set x ticks at interesting points with label
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.ylim(C.min()*1.1,C.max()*1.1)
plt.yticks([-1, 0, +1],
       [r'$-1$', r'$0$', r'$+1$'])
```

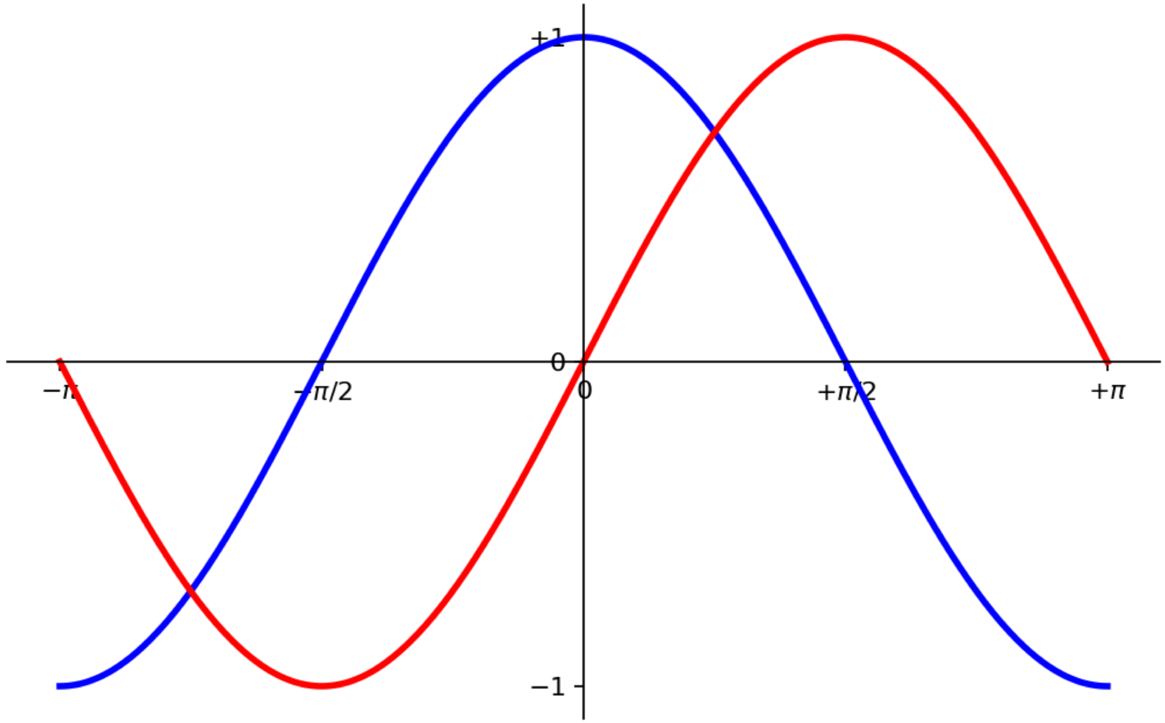

+1
0
−1
−π
−π/2
0
+π/2
+π

Notes on Latex Labels

- Label texts in matplotlib can be Latex equations. For more details on Latex math, please refer to https://en.wikibooks.org/wiki/LaTeX/Mathematics.
- As an example, "r'$\pi/2$'" is a Latex equation marked between two dollar signs "$". "\pi" denotes a predefined symbol of π. "r'$\pi/2$'" will show $\pi/2$.

Exercise 7

```
# get the subplot object
ax = plt.subplot(111)

# set right and top spines invisible
ax.spines['right'].set_color('none')
ax.spines['top'].set_color('none’)
# Show x ticks at bottom
ax.xaxis.set_ticks_position('bottom’)
# Move bottom spine to origin
ax.spines['bottom'].set_position(('data',0))
ax.yaxis.set_ticks_position('left')
ax.spines['left'].set_position(('data',0))
```

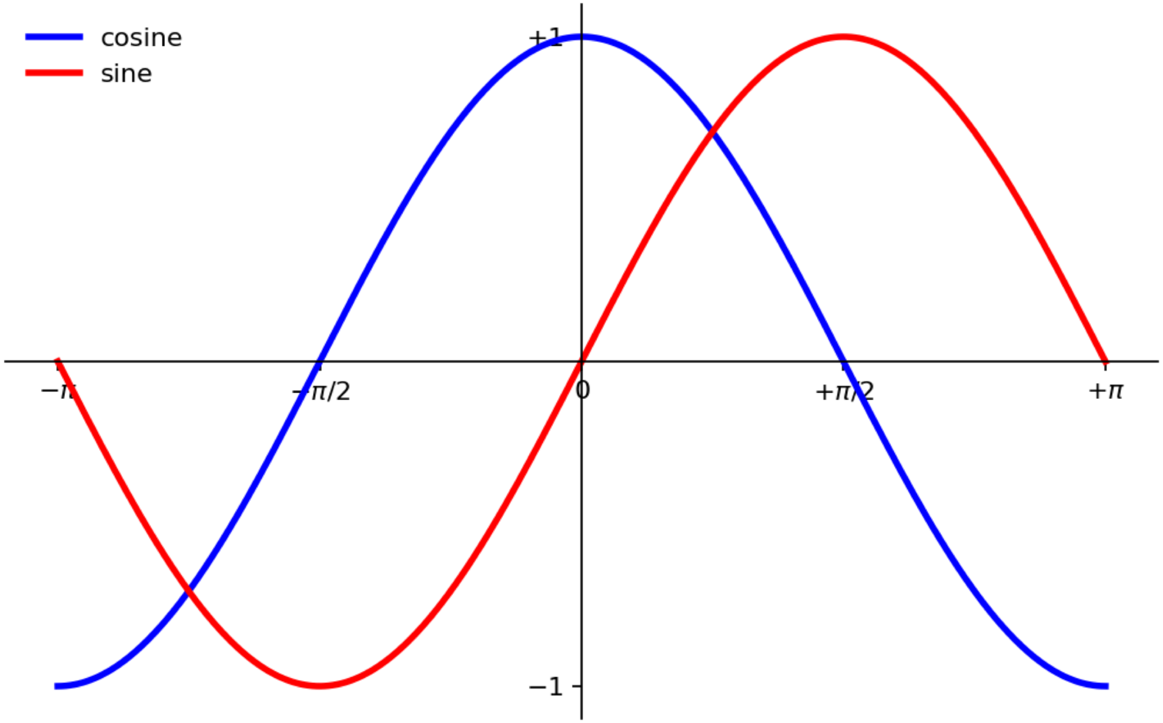

+1
0
−1
−π
−π/2
0
+π/2
+π

Notes on Spines

- Each sub plot has four spines at boundaries: top, bottom, left, right
- Please check https://matplotlib.org/examples/pylab_examples/spine_placement_demo.html for other examples.

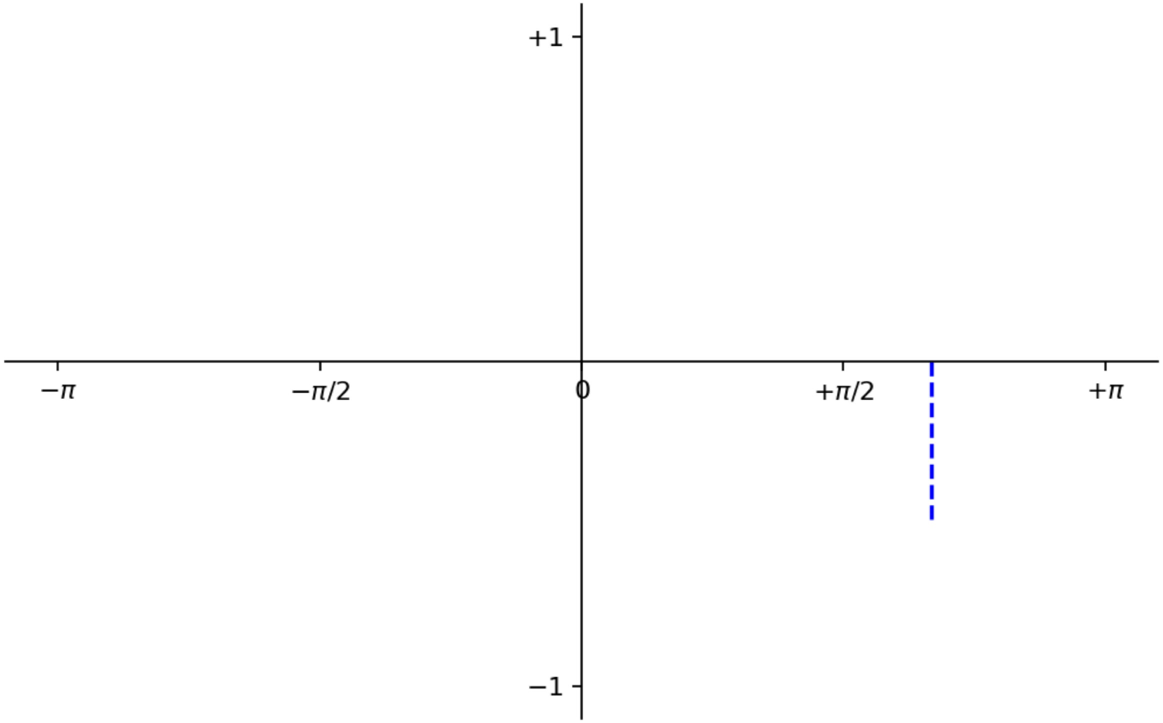
Exercise 8

```
# Add labels (legends) to data plots
plt.plot(X, C, color="blue", linewidth=2.5,
linestyle="-", label="cosine")
plt.plot(X, S, color="red", linewidth=2.5,
linestyle="-", label="sine")


…
# Show legends
plt.legend(loc='upper left', frameon=False)
```


cosine
sine
+1
−1
−π
−π/2
0
+π/2
+π

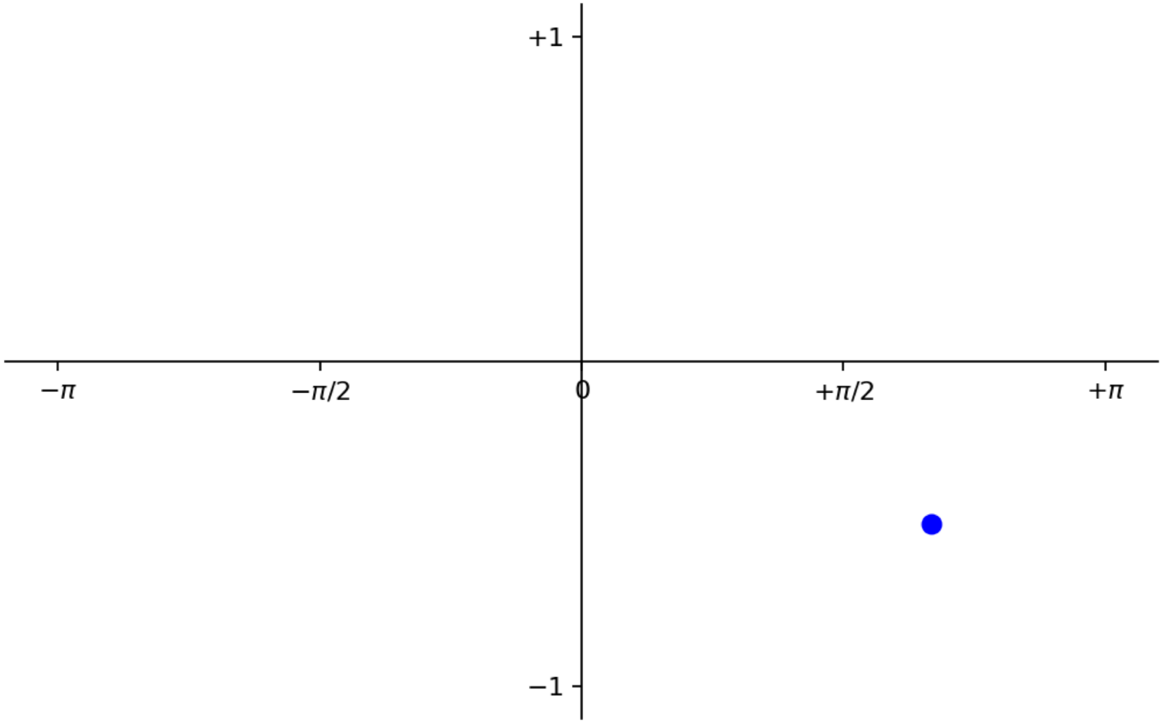
Exercise 9-1

```
# Draw a vertical line from (t, 0) to (t,
cos(t))
t = 2*np.pi/3
plt.plot([t,t],[0,np.cos(t)],
         color ='blue',  linewidth=1.5,
linestyle="--")
```


+1
−1
−π
−π/2
0
+π/2
+π

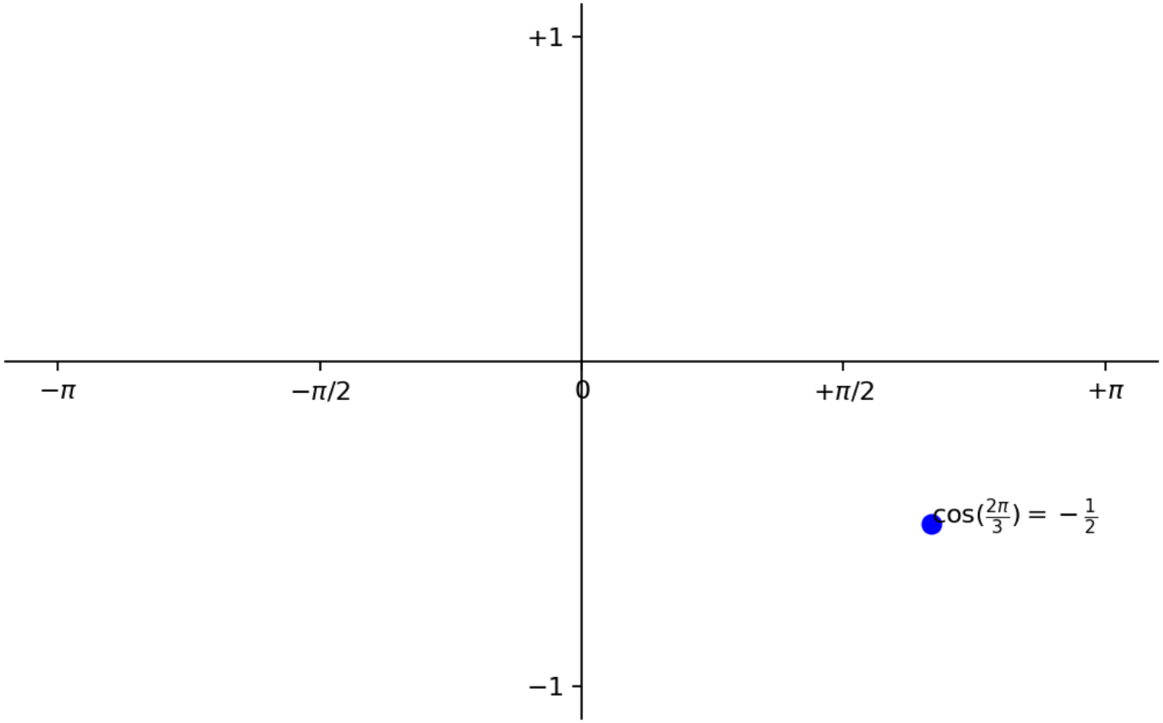
Exercise 9-2

```
# Draw a point at (t, cost(t))
plt.scatter([t,],[np.cos(t),], 50, color='blue')
```


+1
−1
−π
−π/2
0
+π/2
+π

Exercise 9-3

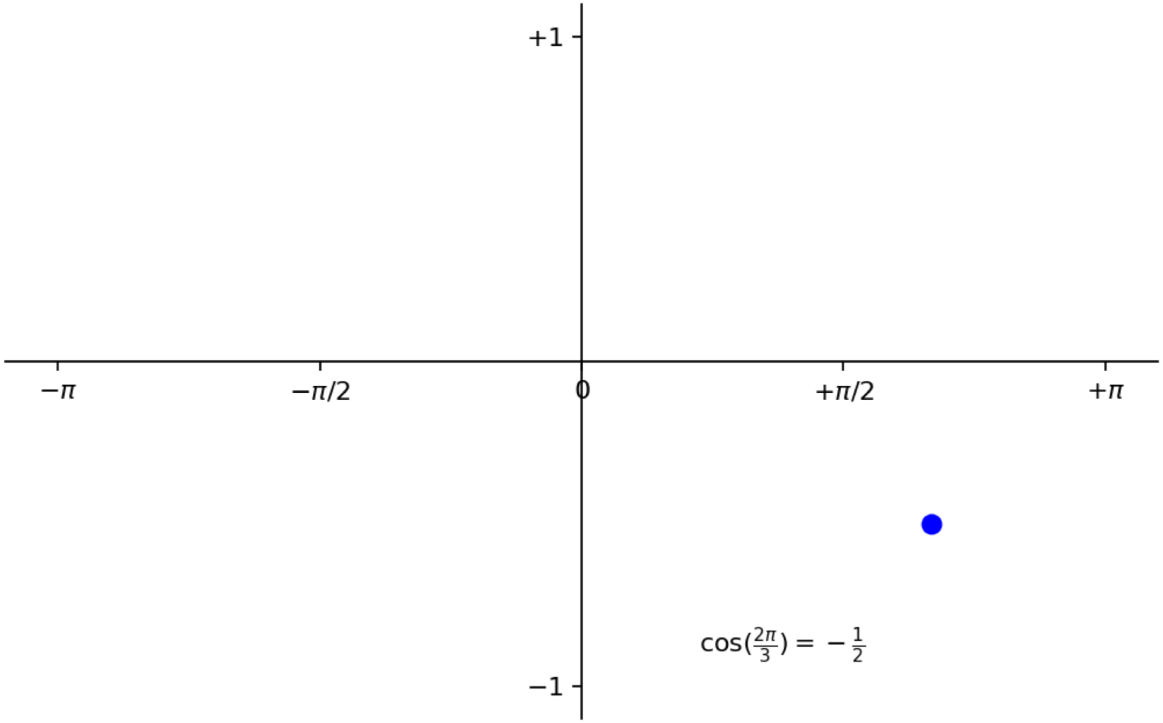
```
# Add a label at (t, cost(t))
plt.annotate(r'$\cos(\frac{2\pi}{3})=-\frac{1}{2}$', xy=(t, np.cos(t)))
```



The label is located at (t, cos(t))

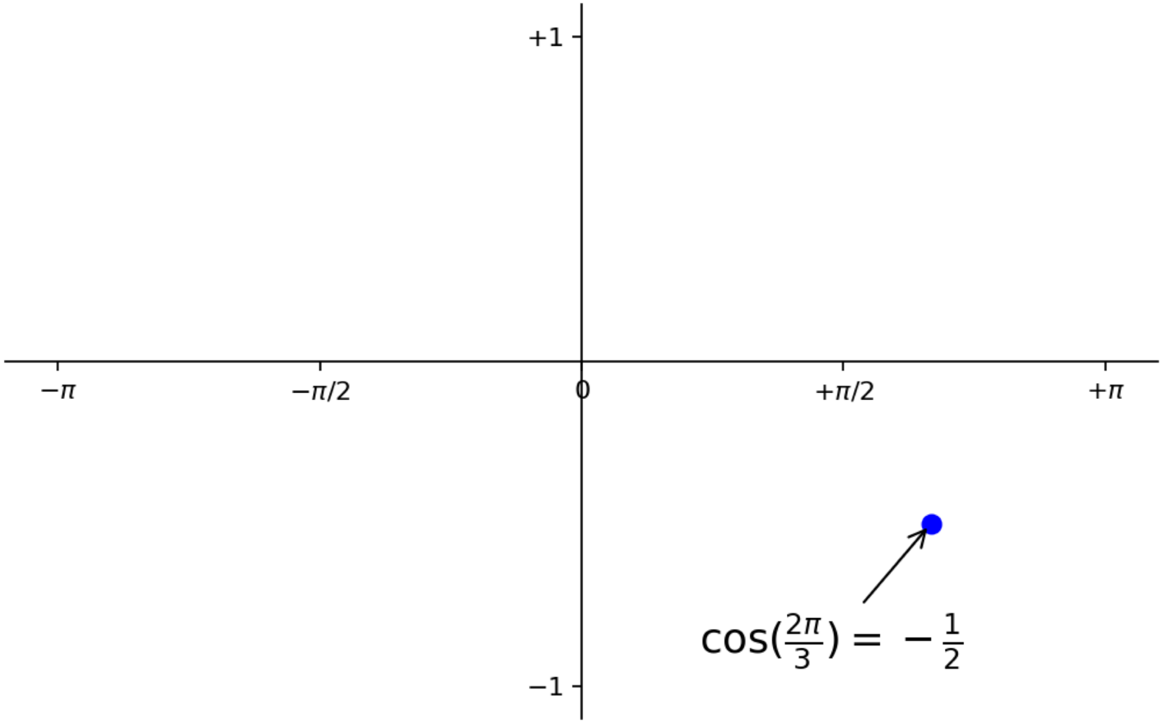
Exercise 9-4

```
# Make the label at (t, cost(t))+(-90, -50)
plt.annotate(r'$\cos(\frac{2\pi}{3})=-
\frac{1}{2}$',
             xy=(t, np.cos(t)),
             xytext=(-90, -50),
textcoords='offset points')
```


+1
−1
−π
−π/2
0
+π/2
+π
$\cos(\frac{2\pi}{3}) = -\frac{1}{2}$

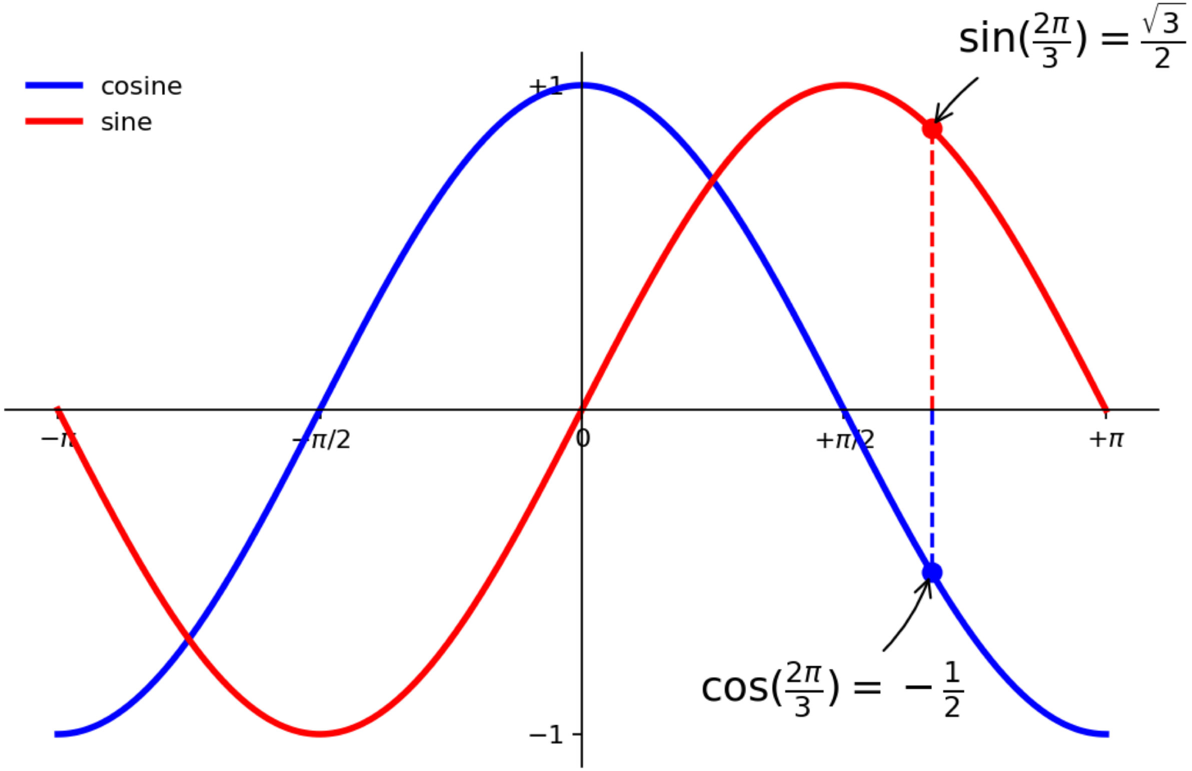
Exercise 9-5

```
# Add an arrow to the labeled data
plt.annotate(r'$\cos(\frac{2\pi}{3})=-
\frac{1}{2}$',
             xy=(t, np.cos(t)),
             xytext=(-90, -50),
textcoords='offset points', fontsize=16,
             arrowprops=dict(arrowstyle="->"))
```


+1
−1
−π
−π/2
0
+π/2
+π
$\cos(\frac{2\pi}{3}) = -\frac{1}{2}$

Exercise 9-6

```
# Add an arc arrow to the labeled data
plt.annotate(r'$\cos(\frac{2\pi}{3})=-\frac{1}{2}$',
             xy=(t, np.cos(t)), xycoords='data',
             xytext=(-90, -50), textcoords='offset points', fontsize=16,
             arrowprops=dict(arrowstyle="->", connectionstyle="arc3,rad=.2"))
```



cosine
sine
$\sin(\frac{2\pi}{3}) = \frac{\sqrt{3}}{2}$
$\cos(\frac{2\pi}{3}) = -\frac{1}{2}$
+1
−1
−π
−π/2
0
+π/2
+π

Notes on Annotate

- More info on "annotate()" can be found at https://matplotlib.org/tutorials/text/annotations.html.

More Plots

- Please read the rest of the tutorial:

https://github.com/rougier/matplotlib-tutorial