Visualization in Python

Courtesy of Prof. John Lui

Computer Science & Engineering Department

The Chinese University of Hong Kong

Data Visualization

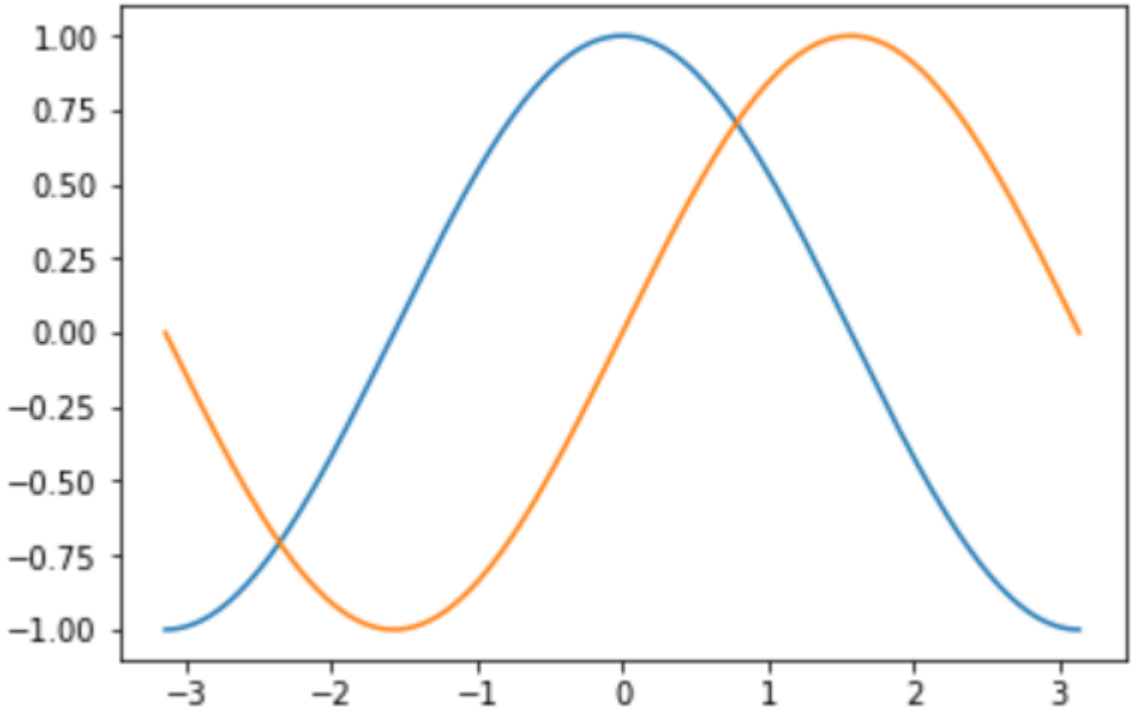
- Data analysts and scientists want to perform data visualization (e.g., plotting graphs, bar chart,..etc)
- Python provides a RICH set of plotting functionalities
- Meet Python library matplotlib

- We can
  - download data,
  - parse the data from columns & rows to list of dictionary,
  - render the data

Matplotlib

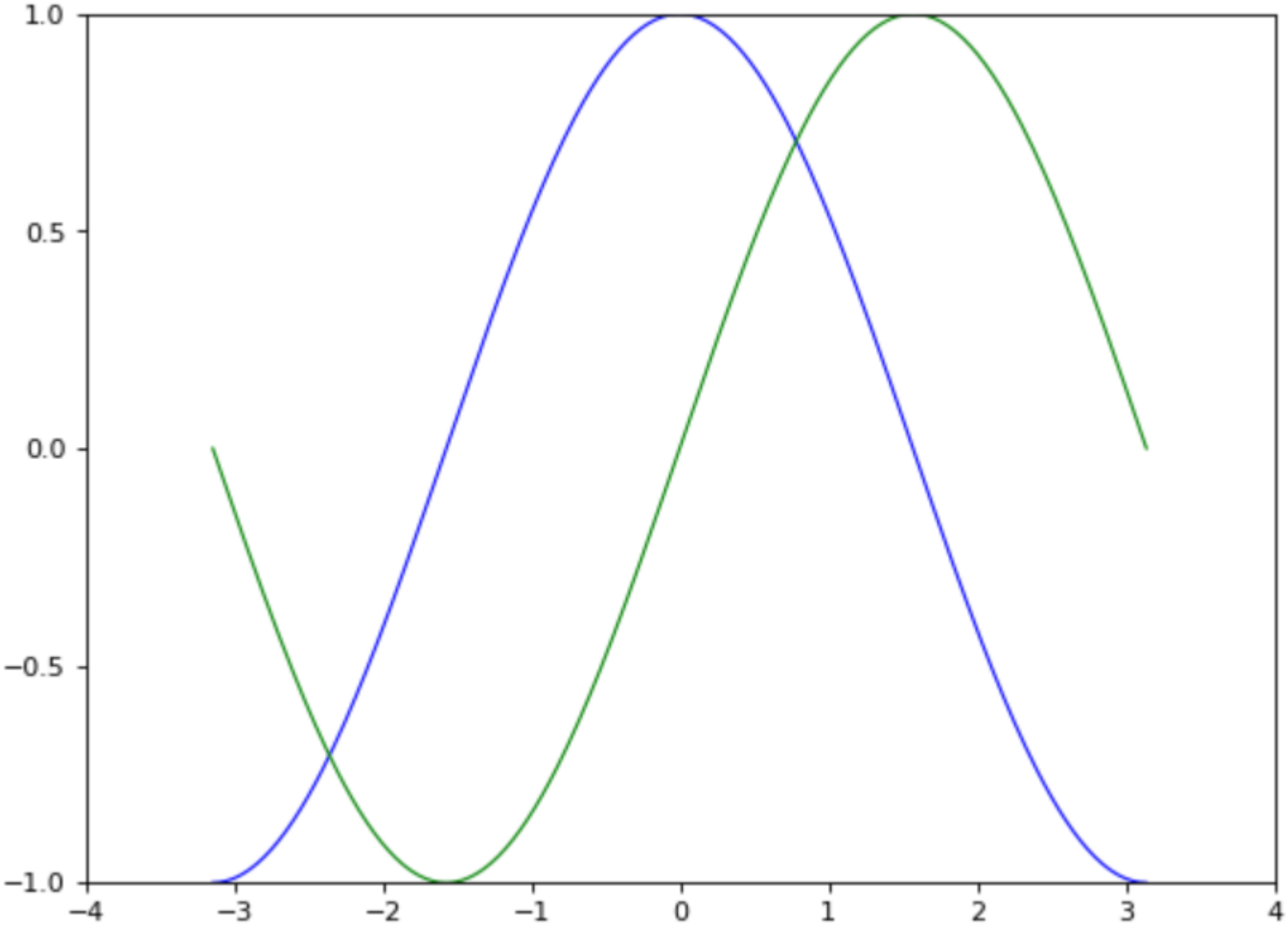
- a popular scientific library that gives the developer tools to produce 2D figures
- need both numpy and matplotlib
- Rich examples: http://matplotlib.org/examples/index.html
- GeoJSON s a derivative of JSON. It's a data format for simple geological feature, including coordinate points.
- GitHub has an awesome feature that allows folks to paste GeoJSON files into Gists, and renders as a map

```
1  Import numpy as np
2  Import matplotlib.pyplot as plt
3  X = np.linspace(-np.pi, np.pi, 256, endpoint=True)
4  C, S = np.cos(X), np.sin(X)
5  plt.plot(X, C)
6  plt.plot(X, S)
7  plt.show()
```



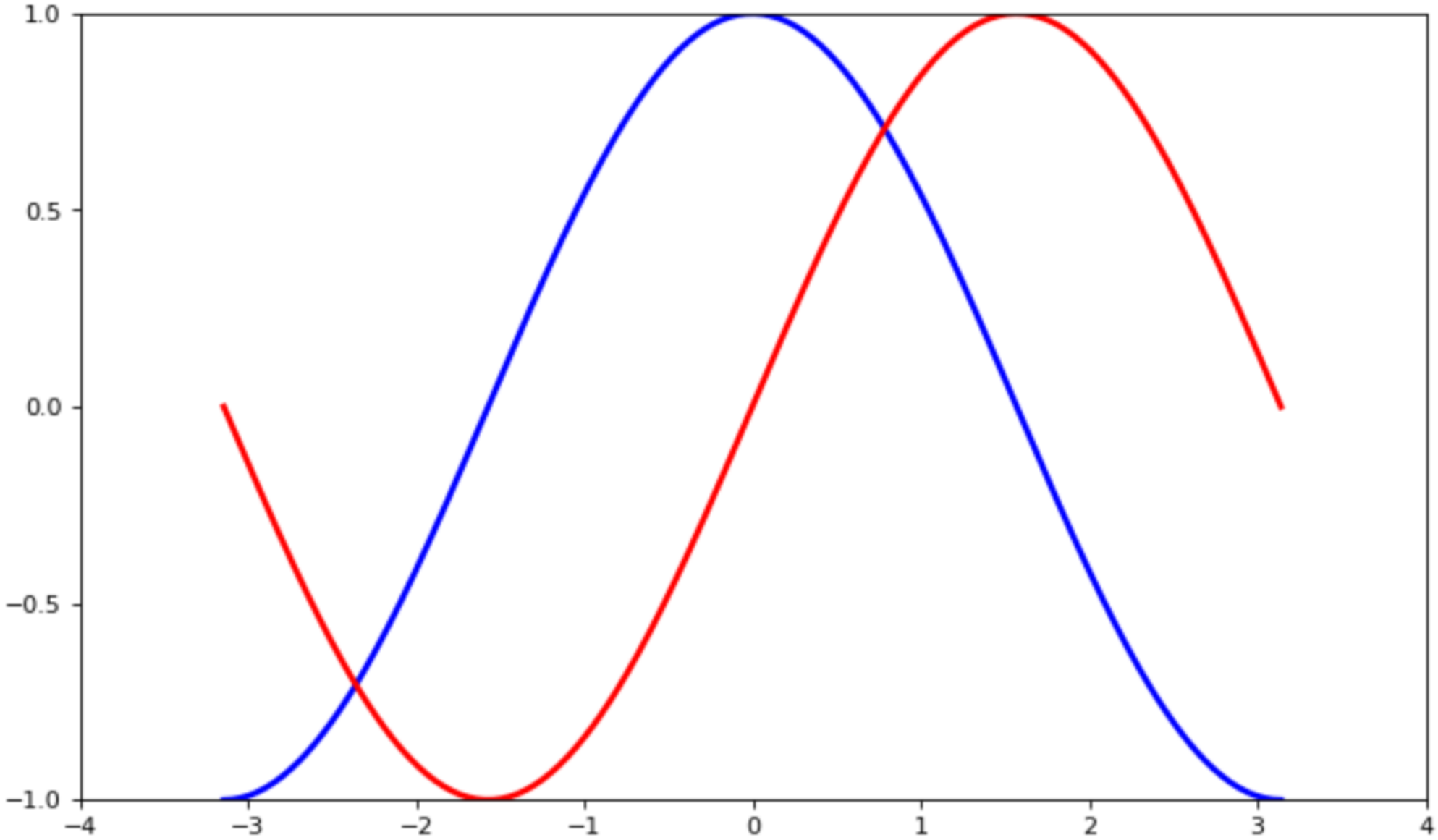
```
1  # plot2.py
2  import numpy as np
3  import matplotlib.pyplot as plt
4
5  # Create a figure of size 8x6 inches, 80 dots per inch
6  plt.figure(figsize=(8, 6), dpi=80)
7
8  # Create a new subplot from a grid of 1x1
9  plt.subplot(1, 1, 1)
10
11 X = np.linspace(-np.pi, np.pi, 256, endpoint=True)
12 C, S = np.cos(X), np.sin(X)
13
14 # Plot cosine with a blue continuous line of width 1 (pixels)
15 plt.plot(X, C, color="blue", linewidth=1.0, linestyle="-")
16
17 # Plot sine with a green continuous line of width 1 (pixels)
18 plt.plot(X, S, color="green", linewidth=1.0, linestyle="-")
```

```
1  # Set x limits
2  plt.xlim(-4.0, 4.0)
3
4  # Set x ticks
5  plt.xticks(np.linspace(-4, 4, 9, endpoint=True))
6
7  # Set y limits
8  plt.ylim(-1.0, 1.0)
9
10 # Set y ticks
11 plt.yticks(np.linspace(-1, 1, 5, endpoint=True))
12
13 # Save figure using 72 dots per inch
14 plt.savefig("exercise_2.png", dpi=72)
15
16 # Show result on screen
17 plt.show()
18
```


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

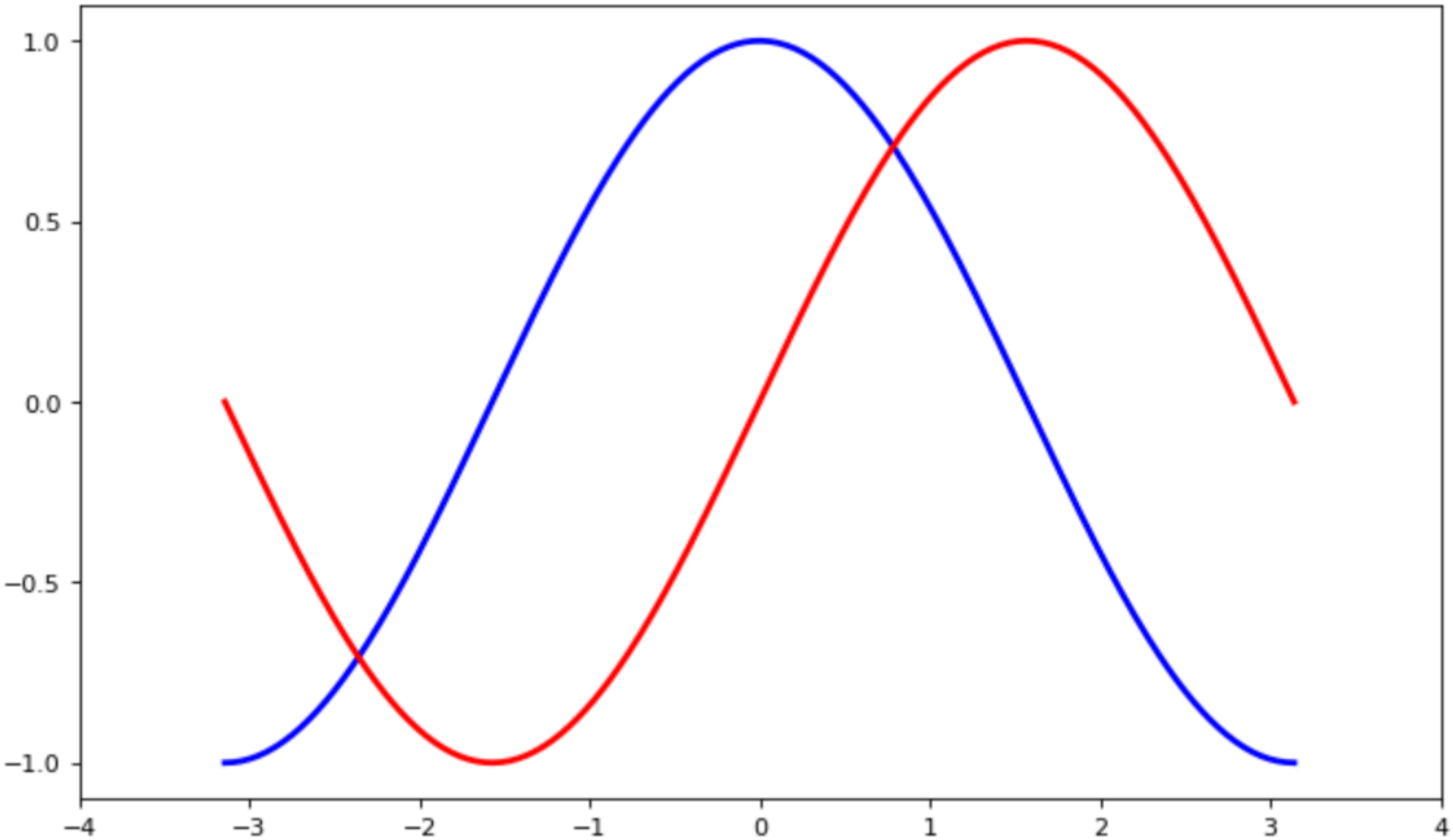
```
1  # ploy3.py
2
3
4  import numpy as np
5  import matplotlib.pyplot as plt
6
7  # Create a figure of size 8x6 inches, 80 dots per inch
8  plt.figure(figsize=(10, 6), dpi=80)
9
10 # Create a new subplot from a grid of 1x1
11 plt.subplot(1, 1, 1)
12
13 X = np.linspace(-np.pi, np.pi, 256, endpoint=True)
14 C, S = np.cos(X), np.sin(X)
15
16 # Plot cosine with a blue continuous line of width 1 (pixels)
17 plt.plot(X, C, color="blue", linewidth=2.5, linestyle="-")
18
```

```
1  # Plot sine with a green continuous line of width 1 (pixels)
2  plt.plot(X, S, color="red", linewidth=2.5, linestyle="-")
3
4  # Set x limits
5  plt.xlim(-4.0, 4.0)
6  # Set x ticks
7  plt.xticks(np.linspace(-4, 4, 9, endpoint=True))
8
9  # Set y limits
10 plt.ylim(-1.0, 1.0)
11 # Set y ticks
12 plt.yticks(np.linspace(-1, 1, 5, endpoint=True))
13
14 # Save figure using 72 dots per inch
15 plt.savefig("exercise_2.png", dpi=72)
16
17 # Show result on screen
18 plt.show()
```


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

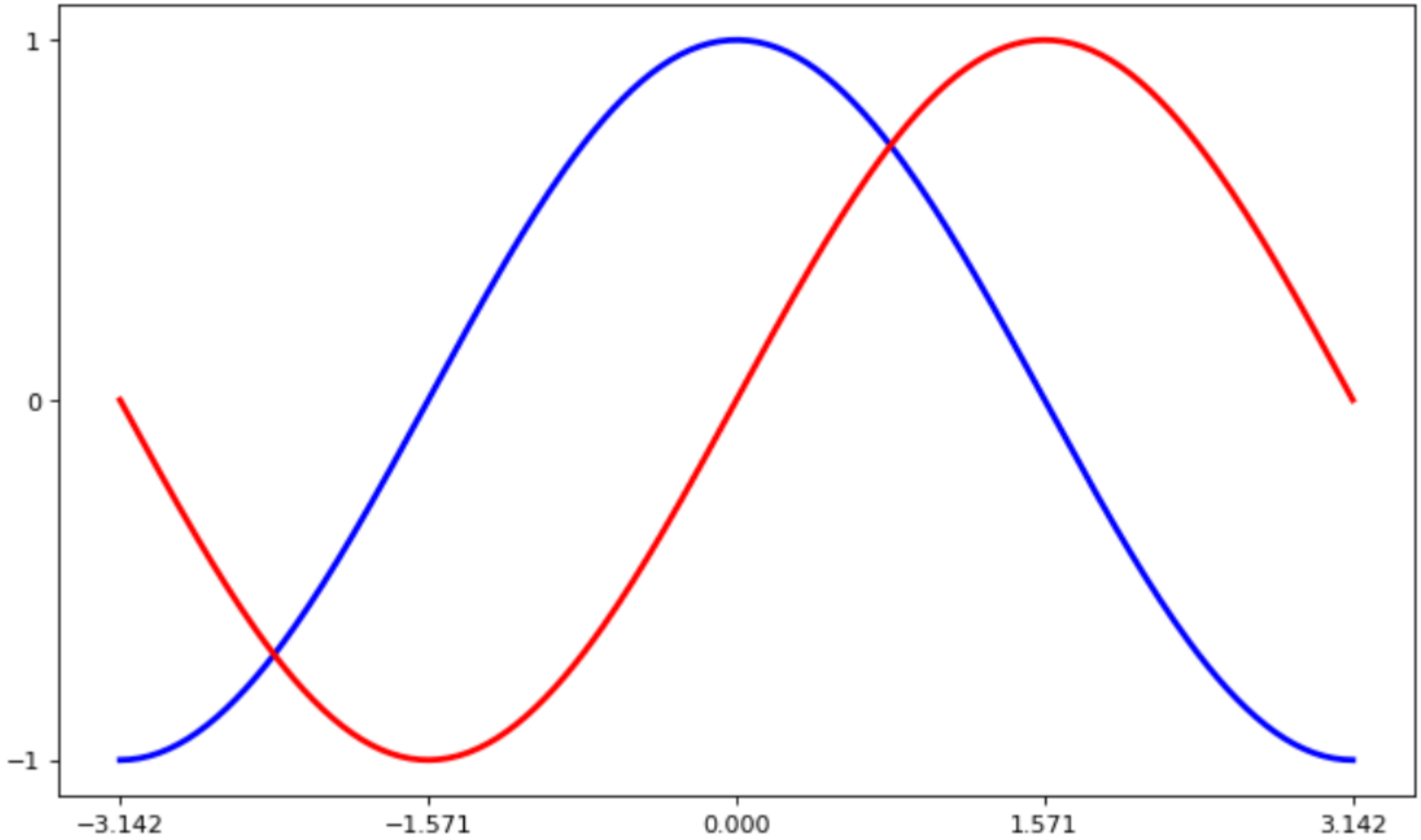
```
1  # plot4.py
2
3
4  import numpy as np
5  import matplotlib.pyplot as plt
6
7  # Create a figure of size 8x6 inches, 80 dots per inch
8  plt.figure(figsize=(10, 6), dpi=80)
9
10 # Create a new subplot from a grid of 1x1
11 plt.subplot(1, 1, 1)
12
13 X = np.linspace(-np.pi, np.pi, 256, endpoint=True)
14 C, S = np.cos(X), np.sin(X)
15
16 # Plot cosine with a blue continuous line of width 1 (pixels)
17 plt.plot(X, C, color="blue", linewidth=2.5, linestyle="-")
18
```

```
1  # Plot sine with a green continuous line of width 1 (pixels)
2  plt.plot(X, S, color="red", linewidth=2.5, linestyle="-")
3
4  # Set x limits
5  plt.xlim(X.min() * 1.1, X.max() * 1.1)
6  # Set x ticks
7  plt.xticks(np.linspace(-4, 4, 9, endpoint=True))
8
9  # Set y limits
10 plt.ylim(C.min() * 1.1, C.max() * 1.1)
11 # Set y ticks
12 plt.yticks(np.linspace(-1, 1, 5, endpoint=True))
13
14 # Save figure using 72 dots per inch
15 plt.savefig("exercise_2.png", dpi=72)
16
17 # Show result on screen
18 plt.show()
```


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

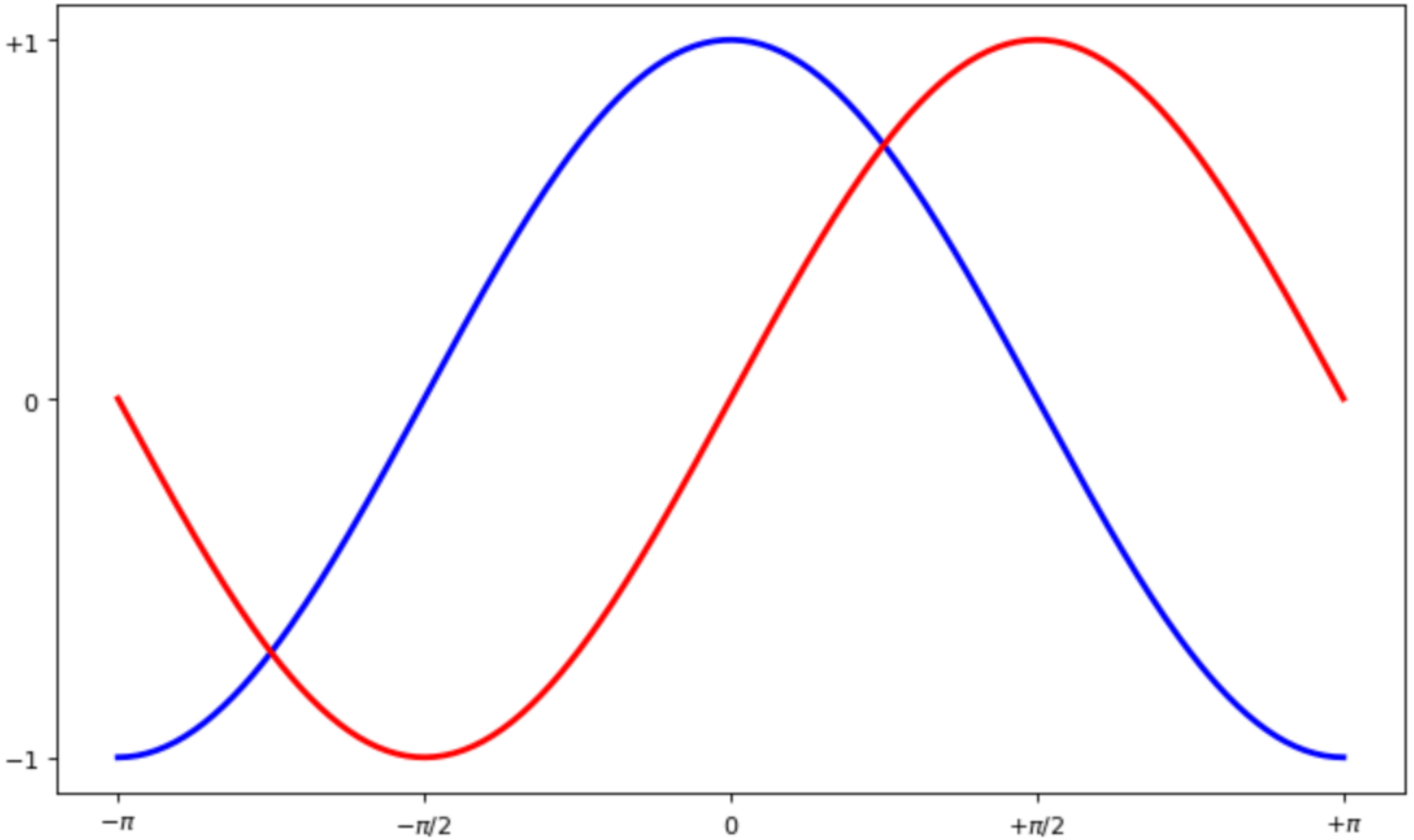
```
1  # plot5.py
2  import numpy as np
3  import matplotlib.pyplot as plt
4
5  # Create a figure of size 8x6 inches, 80 dots per inch
6  plt.figure(figsize=(10, 6), dpi=80)
7
8  # Create a new subplot from a grid of 1x1
9  plt.subplot(1, 1, 1)
10
11 X = np.linspace(-np.pi, np.pi, 256, endpoint=True)
12 C, S = np.cos(X), np.sin(X)
13
14 # Plot cosine with a blue continuous line of width 1 (pixels)
15 plt.plot(X, C, color="blue", linewidth=2.5, linestyle="-")
16
17 # Plot sine with a green continuous line of width 1 (pixels)
18 plt.plot(X, S, color="red", linewidth=2.5, linestyle="-")
```

```
1  # Set x limits
2  plt.xlim(X.min() * 1.1, X.max() * 1.1)
3  # Set x ticks
4  #plt.xticks(np.linspace(-4, 4, 9, endpoint=True))
5  plt.xticks([-np.pi, -np.pi/2, 0, np.pi/2, np.pi])
6
7  # Set y limits
8  plt.ylim(C.min() * 1.1, C.max() * 1.1)
9  # Set y ticks
10 #plt.yticks(np.linspace(-1, 1, 5, endpoint=True))
11 plt.yticks([-1, 0, +1])
12
13 # Save figure using 72 dots per inch
14 plt.savefig("exercice_2.png", dpi=72)
15
16 # Show result on screen
17 plt.show()
18
```


1
0
−1
−3.142
−1.571
0.000
1.571
3.142

```
1  # plot6.py
2  import numpy as np
3  import matplotlib.pyplot as plt
4
5  # Create a figure of size 8x6 inches, 80 dots per inch
6  plt.figure(figsize=(10, 6), dpi=80)
7
8  # Create a new subplot from a grid of 1x1
9  plt.subplot(1, 1, 1)
10
11 X = np.linspace(-np.pi, np.pi, 256, endpoint=True)
12 C, S = np.cos(X), np.sin(X)
13
14 # Plot cosine with a blue continuous line of width 1 (pixels)
15 plt.plot(X, C, color="blue", linewidth=2.5, linestyle="-")
16
17 # Plot sine with a green continuous line of width 1 (pixels)
18 plt.plot(X, S, color="red", linewidth=2.5, linestyle="-")
```

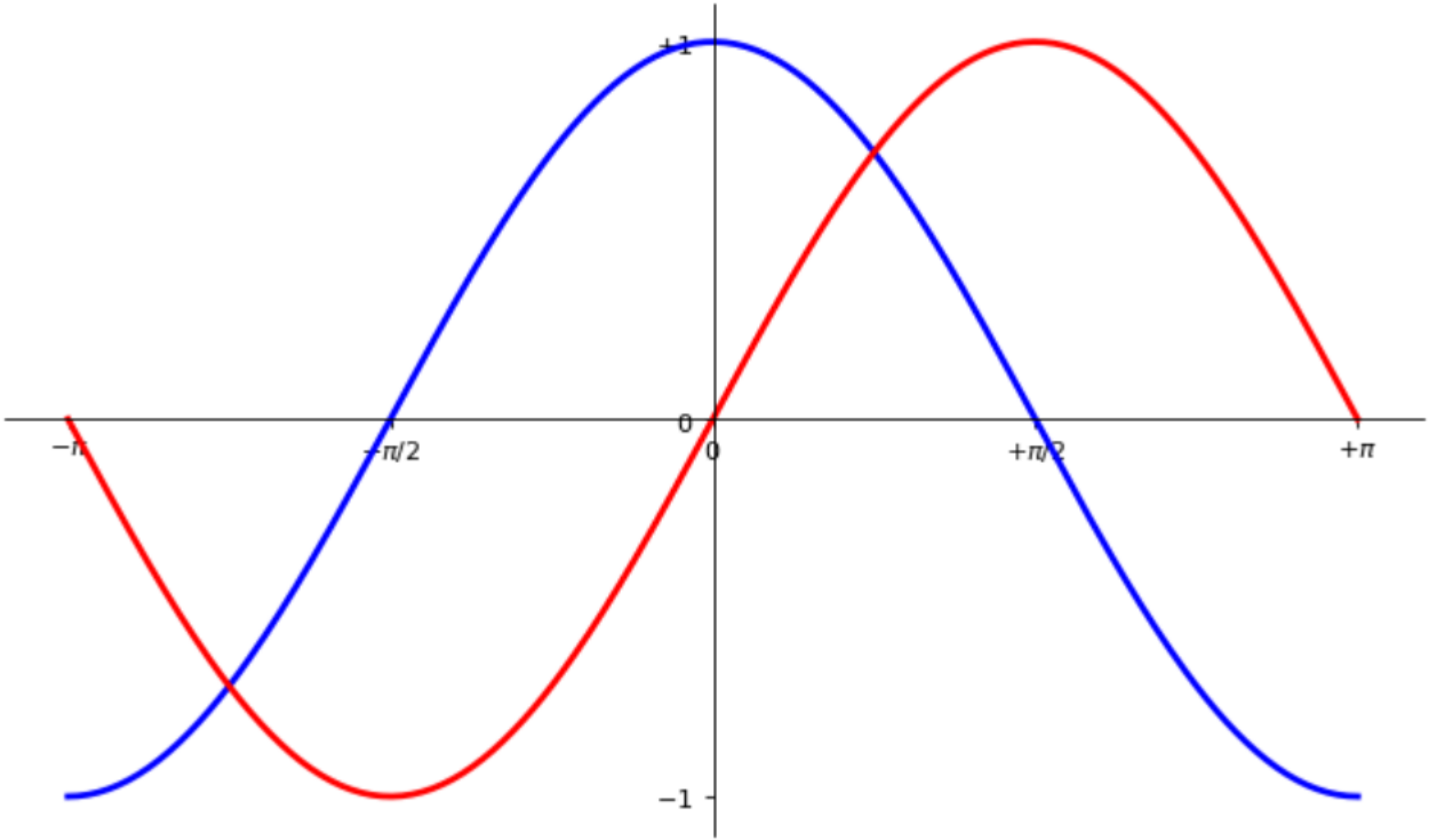
```
1  # Set x limits
2  plt.xlim(X.min() * 1.1, X.max() * 1.1)
3  # Set x ticks
4  #plt.xticks(np.linspace(-4, 4, 9, endpoint=True))
5  #plt.xticks([-np.pi, -np.pi/2, 0, np.pi/2, np.pi])
6  plt.xticks([-np.pi, -np.pi/2, 0, np.pi/2, np.pi],
7          [r'$-\pi$', r'$-\pi/2$', r'$0$', r'$+\pi/2$', r'$+\pi$'])
8  # Set y limits
9  plt.ylim(C.min() * 1.1, C.max() * 1.1)
10 # Set y ticks
11 #plt.yticks(np.linspace(-1, 1, 5, endpoint=True))
12 #plt.yticks([-1, 0, +1])
13 plt.yticks([-1, 0, +1],
14         [r'$-1$', r'$0$', r'$+1$'])
15 # Save figure using 72 dots per inch
16 plt.savefig("exercice_2.png", dpi=72)
17 # Show result on screen
18 plt.show()
```


+1
0
−1
−π
−π/2
0
+π/2
+π

```
1  # plot7.py
2  import numpy as np
3  import matplotlib.pyplot as plt
4  # Create a figure of size 8x6 inches, 80 dots per inch
5  plt.figure(figsize=(10, 6), dpi=80)
6  # Create a new subplot from a grid of 1x1
7  plt.subplot(1, 1, 1)
8  X = np.linspace(-np.pi, np.pi, 256, endpoint=True)
9  C, S = np.cos(X), np.sin(X)
10 # Plot cosine with a blue continuous line of width 1 (pixels)
11 plt.plot(X, C, color="blue", linewidth=2.5, linestyle="-")
12 # Plot sine with a green continuous line of width 1 (pixels)
13 plt.plot(X, S, color="red", linewidth=2.5, linestyle="-")
14
15 # Set x limits
16 plt.xlim(X.min() * 1.1, X.max() * 1.1)
17
18
```

```
1   # Set x ticks
2   #plt.xticks(np.linspace(-4, 4, 9, endpoint=True))
3   #plt.xticks([-np.pi, -np.pi/2, 0, np.pi/2, np.pi])
4   plt.xticks([-np.pi, -np.pi/2, 0, np.pi/2, np.pi],
5           [r'$-\pi$', r'$-\pi/2$', r'$0$', r'$+\pi/2$', r'$+\pi$'])
6
7   # Set y limits
8   plt.ylim(C.min() * 1.1, C.max() * 1.1)
9
10  # Set y ticks
11  #plt.yticks(np.linspace(-1, 1, 5, endpoint=True))
12  #plt.yticks([-1, 0, +1])
13  plt.yticks([-1, 0, +1],
14          [r'$-1$', r'$0$', r'$+1$'])
15
16
17
18
```

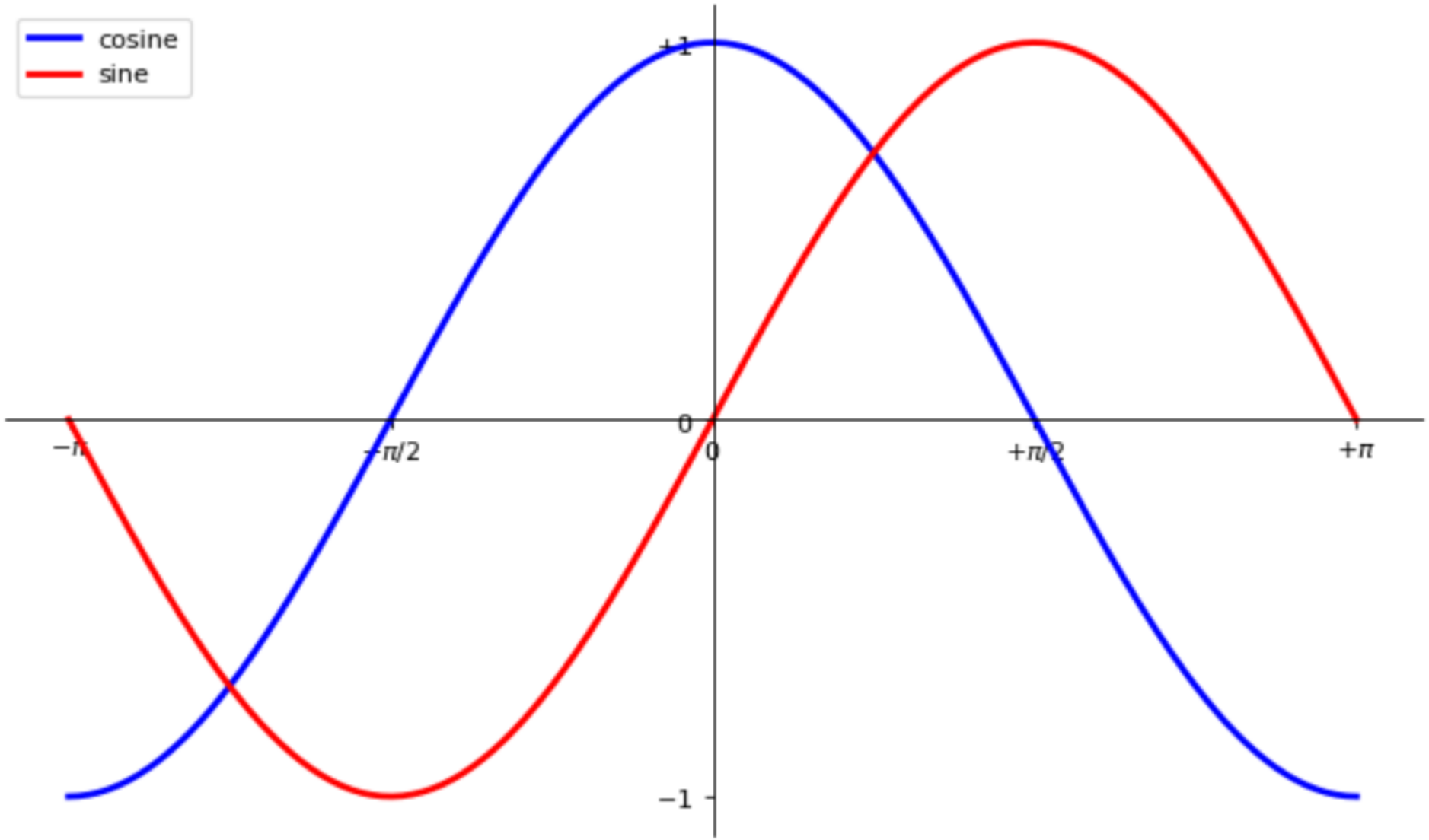
```
1  # Save figure using 72 dots per inch
2  #plt.savefig("exercice_2.png", dpi=72)
3  ax = plt.gca()  # gca stands for 'get current axis'
4  ax.spines['right'].set_color('none')
5  ax.spines['top'].set_color('none')
6  ax.xaxis.set_ticks_position('bottom')
7  ax.spines['bottom'].set_position(('data',0))
8  ax.yaxis.set_ticks_position('left')
9  ax.spines['left'].set_position(('data',0))
10
11
12 # Show result on screen
13 plt.show()
14
15
16
17
18
```


+1
0
0
−1
−π
−π/2
+π/2
+π

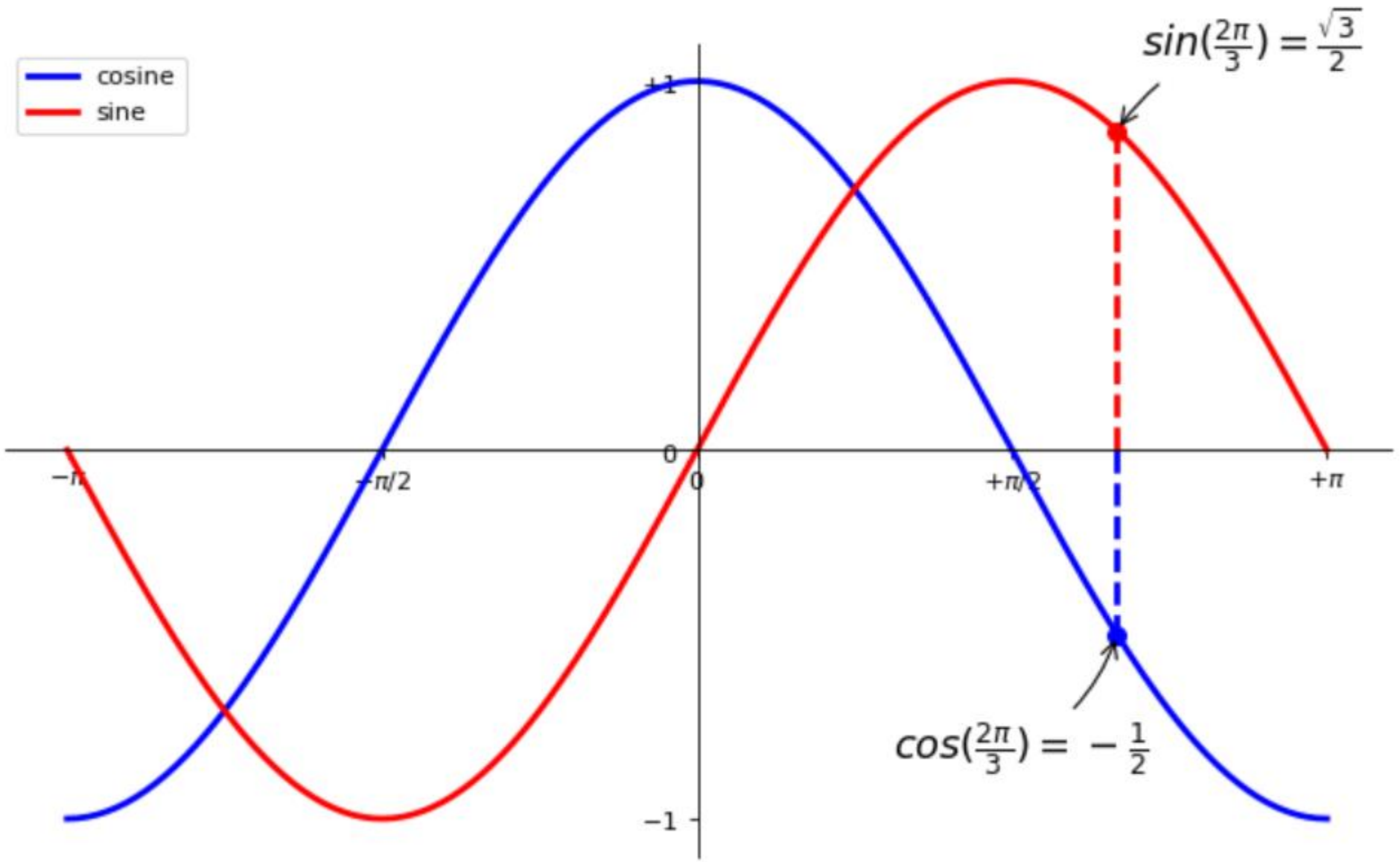
```
1  # plot8.py
2  import numpy as np
3  import matplotlib.pyplot as plt
4  # Create a figure of size 8x6 inches, 80 dots per inch
5  plt.figure(figsize=(10, 6), dpi=80)
6  # Create a new subplot from a grid of 1x1
7  plt.subplot(1, 1, 1)
8
9  X = np.linspace(-np.pi, np.pi, 256, endpoint=True)
10 C, S = np.cos(X), np.sin(X)
11
12 # Plot cosine with a blue continuous line of width 1 (pixels)
13 #plt.plot(X, C, color="blue", linewidth=2.5, linestyle="-")
14 plt.plot(X, C, color="blue", linewidth=2.5, linestyle="-", label="cosine")
15
16 # Plot sine with a green continuous line of width 1 (pixels)
17 #plt.plot(X, S, color="red", linewidth=2.5, linestyle="-")
18 plt.plot(X, S, color="red",  linewidth=2.5, linestyle="-", label="sine")
```

```
1   # Set x limits
2   plt.xlim(X.min() * 1.1, X.max() * 1.1)
3
4   # Set x ticks
5   #plt.xticks(np.linspace(-4, 4, 9, endpoint=True))
6   #plt.xticks([-np.pi, -np.pi/2, 0, np.pi/2, np.pi])
7   plt.xticks([-np.pi, -np.pi/2, 0, np.pi/2, np.pi],
8           [r'$-\pi$', r'$-\pi/2$', r'$0$', r'$+\pi/2$', r'$+\pi$'])
9
10  # Set y limits
11  plt.ylim(C.min() * 1.1, C.max() * 1.1)
12
13  # Set y ticks
14  #plt.yticks(np.linspace(-1, 1, 5, endpoint=True))
15  #plt.yticks([-1, 0, +1])
16  plt.yticks([-1, 0, +1],
17          [r'$-1$', r'$0$', r'$+1$'])
18
```

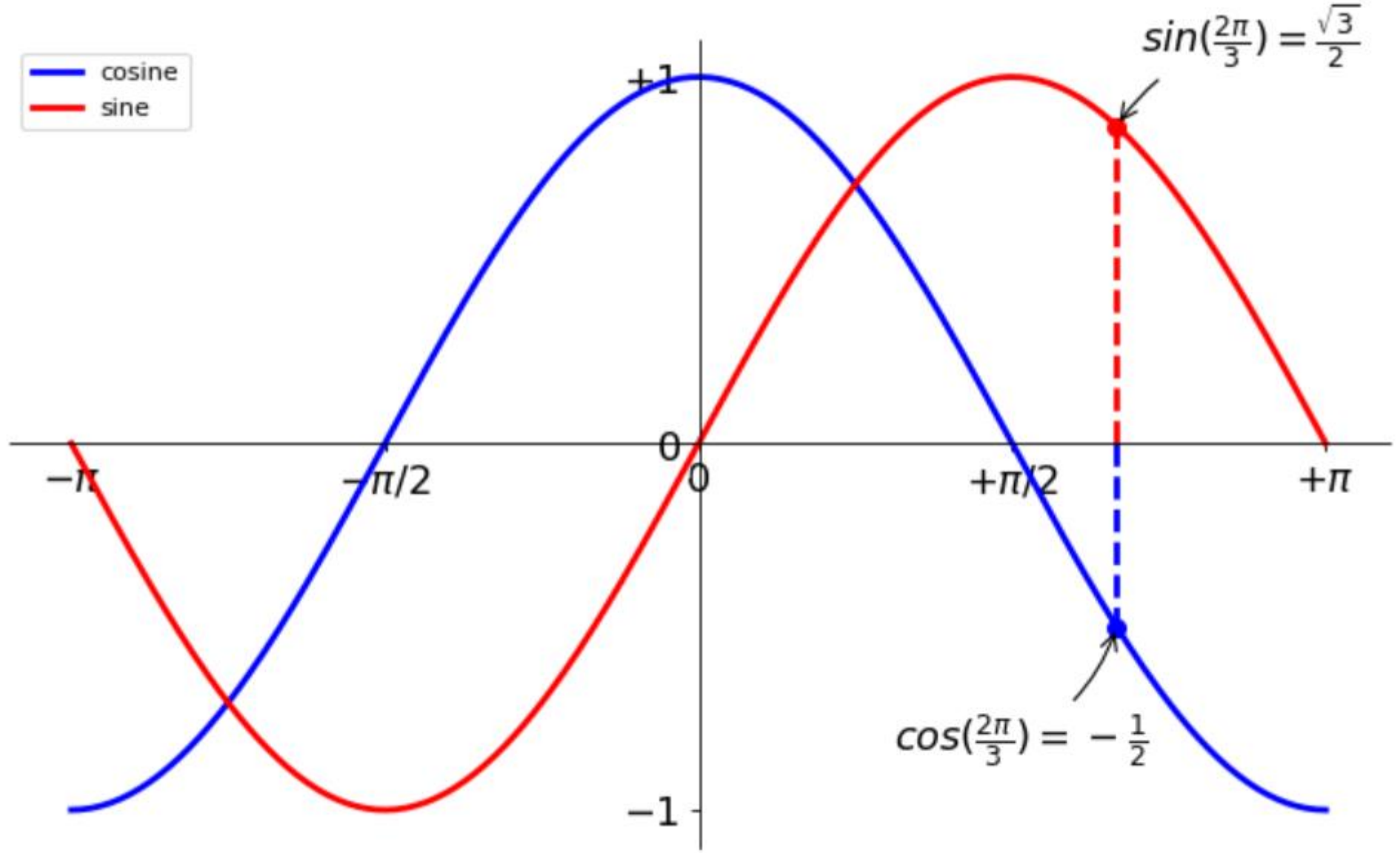
```
1  # Save figure using 72 dots per inch
2  #plt.savefig("exercice_2.png", dpi=72)
3
4  ax = plt.gca()  # gca stands for 'get current axis'
5  ax.spines['right'].set_color('none')
6  ax.spines['top'].set_color('none')
7  ax.xaxis.set_ticks_position('bottom')
8  ax.spines['bottom'].set_position(('data',0))
9  ax.yaxis.set_ticks_position('left')
10 ax.spines['left'].set_position(('data',0))
11
12 # Print legend
13 plt.legend(loc='upper left')
14
15 # Show result on screen
16 plt.show()
17
18
```


cosine
sine
+1
0
−1
−π
−π/2
0
+π/2
+π

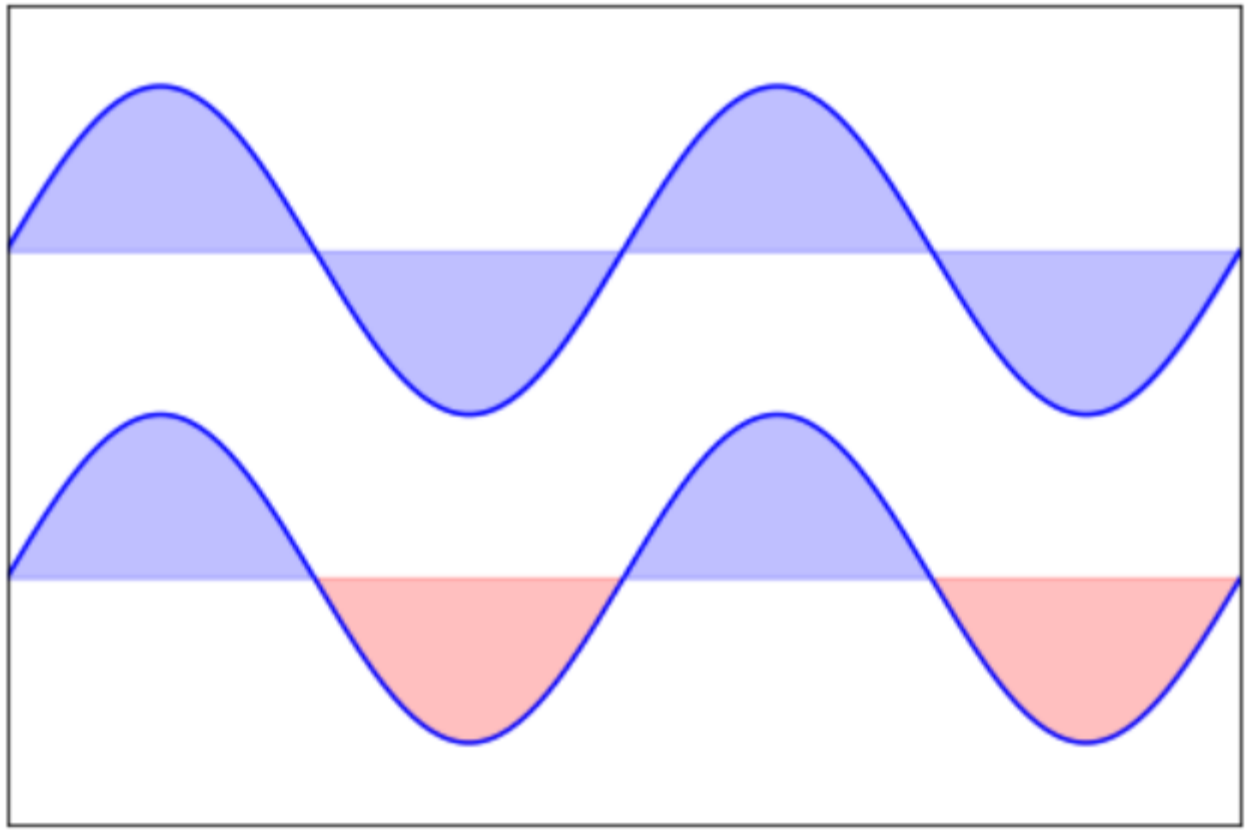
```
# annotate some points
t = 2 * np.pi / 3
plt.plot([t, t], [0, np.cos(t)], color='blue', linewidth=2.5, linestyle="--")
plt.scatter([t, ], [np.cos(t), ], 50, color='blue')

plt.annotate(r'$sin(\frac{2\pi}{3})=\frac{\sqrt{3}}{2}$',
             xy=(t, np.sin(t)), xycoords='data',
             xytext=(+10, +30), textcoords='offset points', fontsize=16,
             arrowprops=dict(arrowstyle="->", connectionstyle="arc3,rad=.2"))

plt.plot([t, t],[0, np.sin(t)], color='red', linewidth=2.5, linestyle="--")
plt.scatter([t, ],[np.sin(t), ], 50, color='red')

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
0
−1
−π
−π/2
0
+π/2
+π

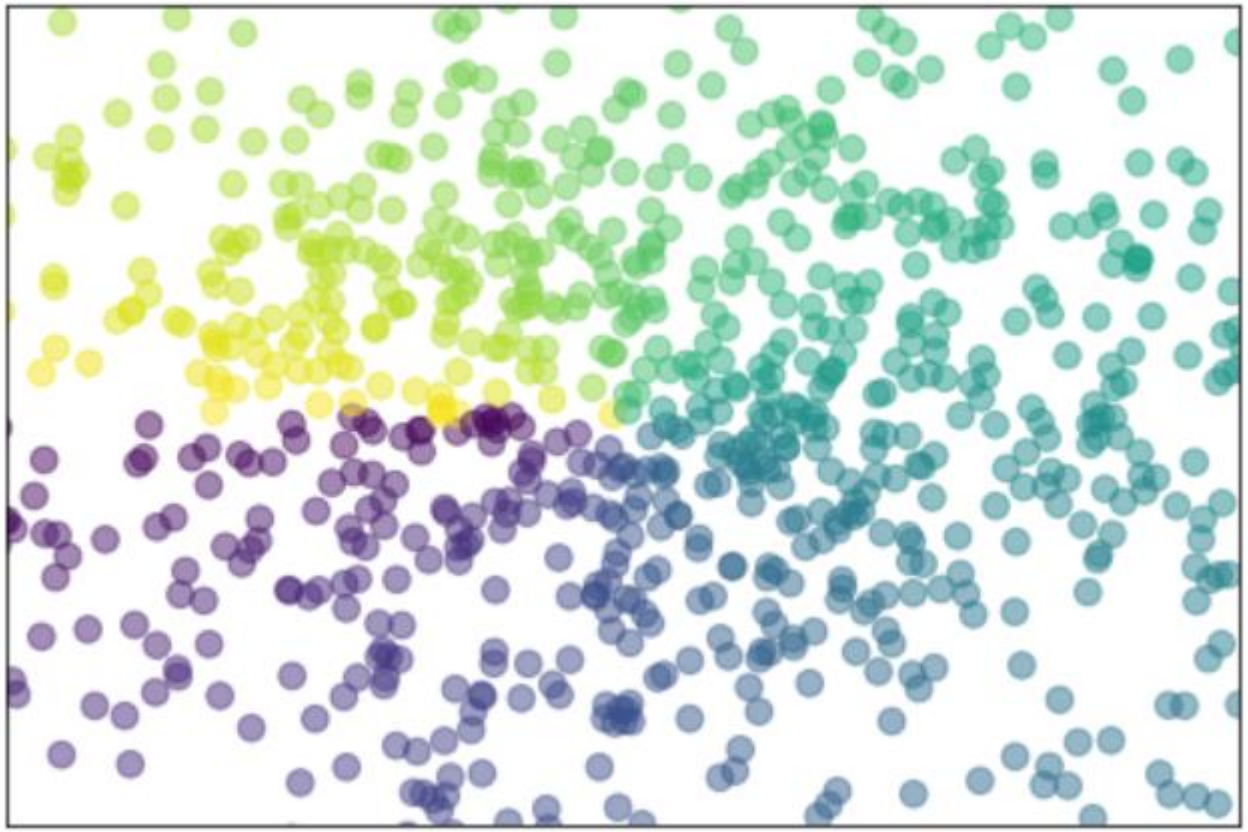
```
# reset x-labels
for label in ax.get_xticklabels() + ax.get_yticklabels():
    label.set_fontsize(16)
    label.set_bbox(dict(facecolor='white', edgecolor='None', alpha=0.65))
```


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

```
1  # plot11.py
2  import numpy as np
3  import matplotlib.pyplot as plt
4  n = 256
5  X = np.linspace(-np.pi, np.pi, n, endpoint=True)
6  Y = np.sin(2 * X)
7  plt.axes([0.025, 0.025, 0.95, 0.95])
8  plt.plot(X, Y + 1, color='blue', alpha=1.00)
9  plt.fill_between(X, 1, Y + 1, color='blue', alpha=.25)
10
11 plt.plot(X, Y - 1, color='blue', alpha=1.00)
12 plt.fill_between(X, -1, Y - 1, (Y - 1) > -1, color='blue', alpha=.25)
13 plt.fill_between(X, -1, Y - 1, (Y - 1) < -1, color='red',  alpha=.25)
14
15 plt.xlim(-np.pi, np.pi)
16 plt.xticks(())
17 plt.ylim(-2.5, 2.5)
18 plt.yticks(())
   plt.show()
```

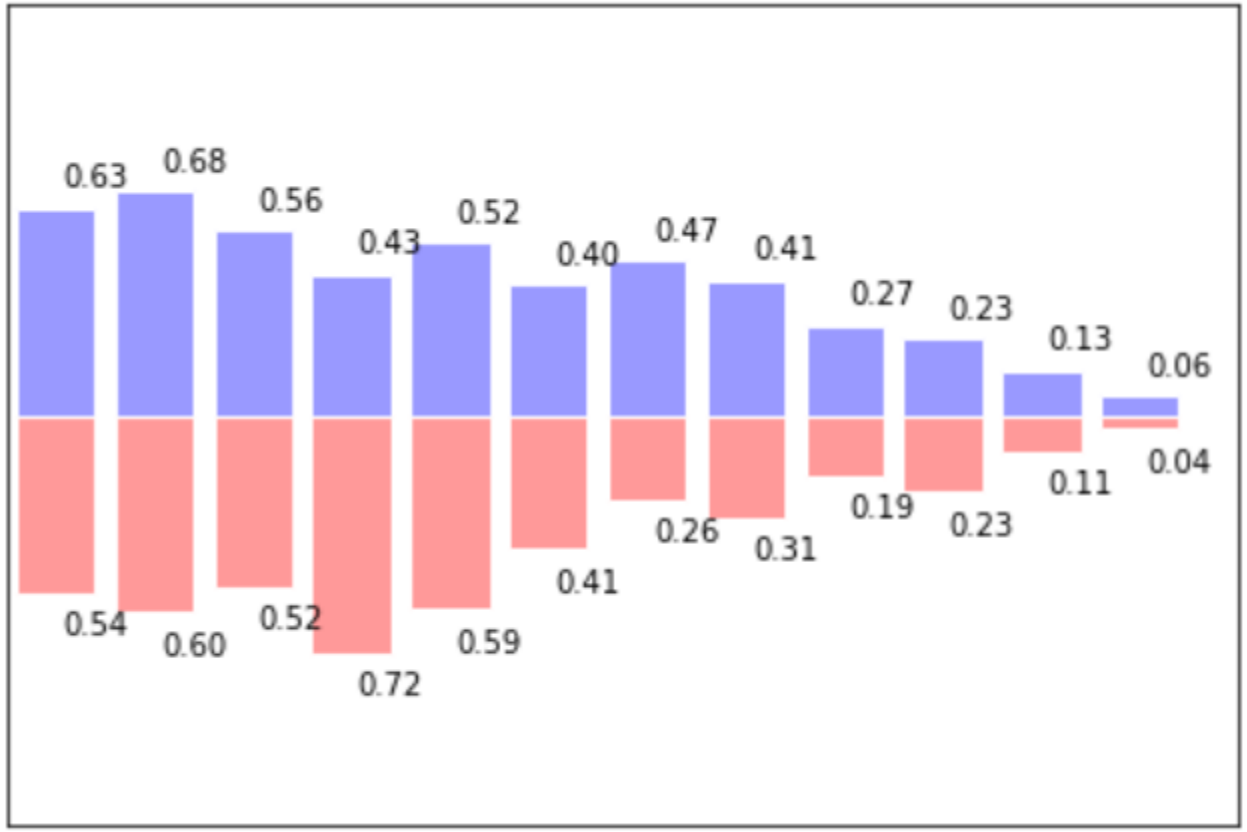


```
1  # plot12.py
2  import numpy as np
3  import matplotlib.pyplot as plt
4
5  n = 1024
6  X = np.random.normal(0, 1, n)
7  Y = np.random.normal(0, 1, n)
8  T = np.arctan2(Y, X)
9
10 plt.axes([0.025, 0.025, 0.95, 0.95])
11 plt.scatter(X, Y, s=75, c=T, alpha=.5)
12
13 plt.xlim(-1.5, 1.5)
14 plt.xticks(())
15 plt.ylim(-1.5, 1.5)
16 plt.yticks(())
17
18 plt.show()
```



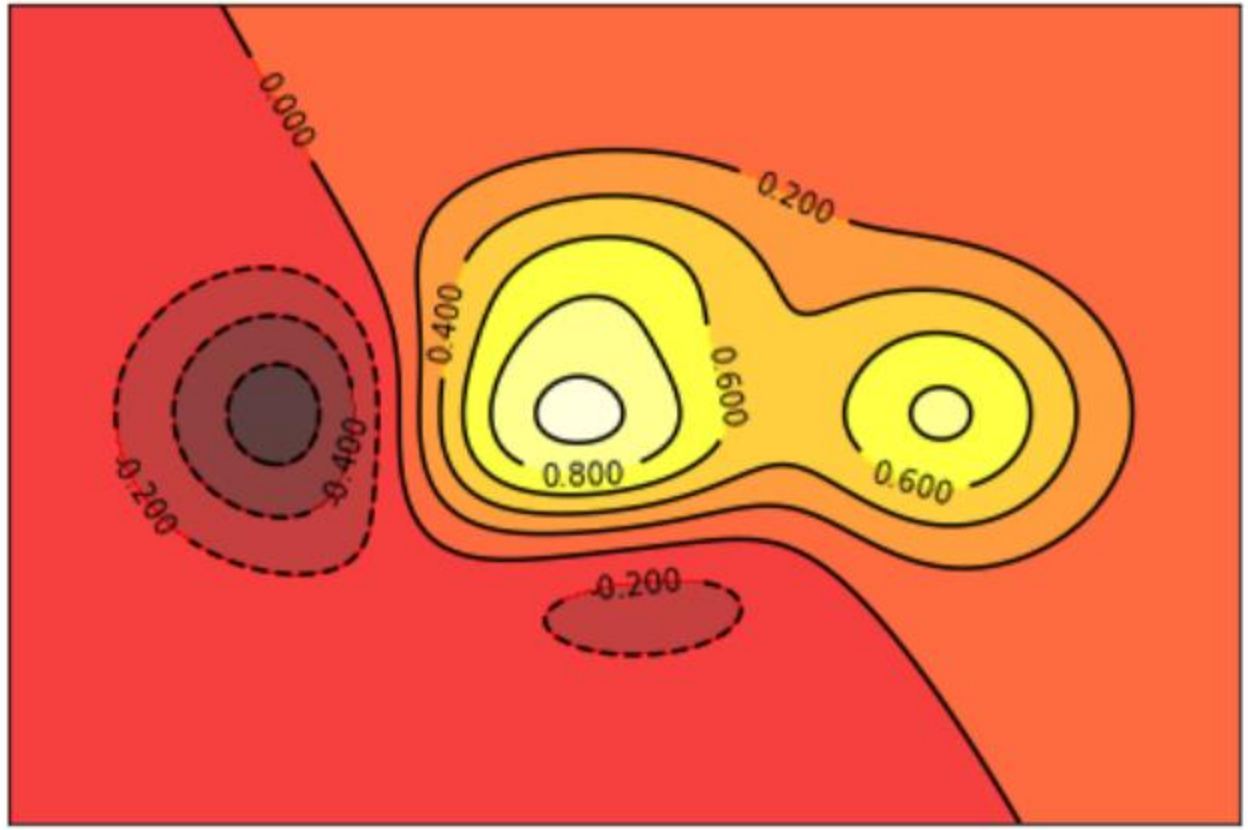
```
1  # plot13.py
2  import numpy as np
3  import matplotlib.pyplot as plt
4
5  n = 12
6  X = np.arange(n)
7  Y1 = (1 - X / float(n)) * np.random.uniform(0.5, 1.0, n)
8  Y2 = (1 - X / float(n)) * np.random.uniform(0.5, 1.0, n)
9
10 plt.axes([0.025, 0.025, 0.95, 0.95])
11 plt.bar(X, +Y1, facecolor='#9999ff', edgecolor='white')
12 plt.bar(X, -Y2, facecolor='#ff9999', edgecolor='white')
13
14
15
16
17
18
```

```
1  for x, y in zip(X, Y1):
2      plt.text(x + 0.4, y + 0.05, '%.2f' % y, ha='center', va= 'bottom')
3
4  for x, y in zip(X, Y2):
5      plt.text(x + 0.4, -y - 0.05, '%.2f' % y, ha='center', va= 'top')
6
7  plt.xlim(-.5, n)
8  plt.xticks(())
9  plt.ylim(-1.25, 1.25)
10 plt.yticks(())
11
12 plt.show()
13
14
15
16
17
18
```


0.63
0.68
0.56
0.43
0.52
0.40
0.47
0.41
0.27
0.23
0.13
0.06
0.54
0.60
0.52
0.72
0.59
0.41
0.26
0.31
0.19
0.23
0.11
0.04

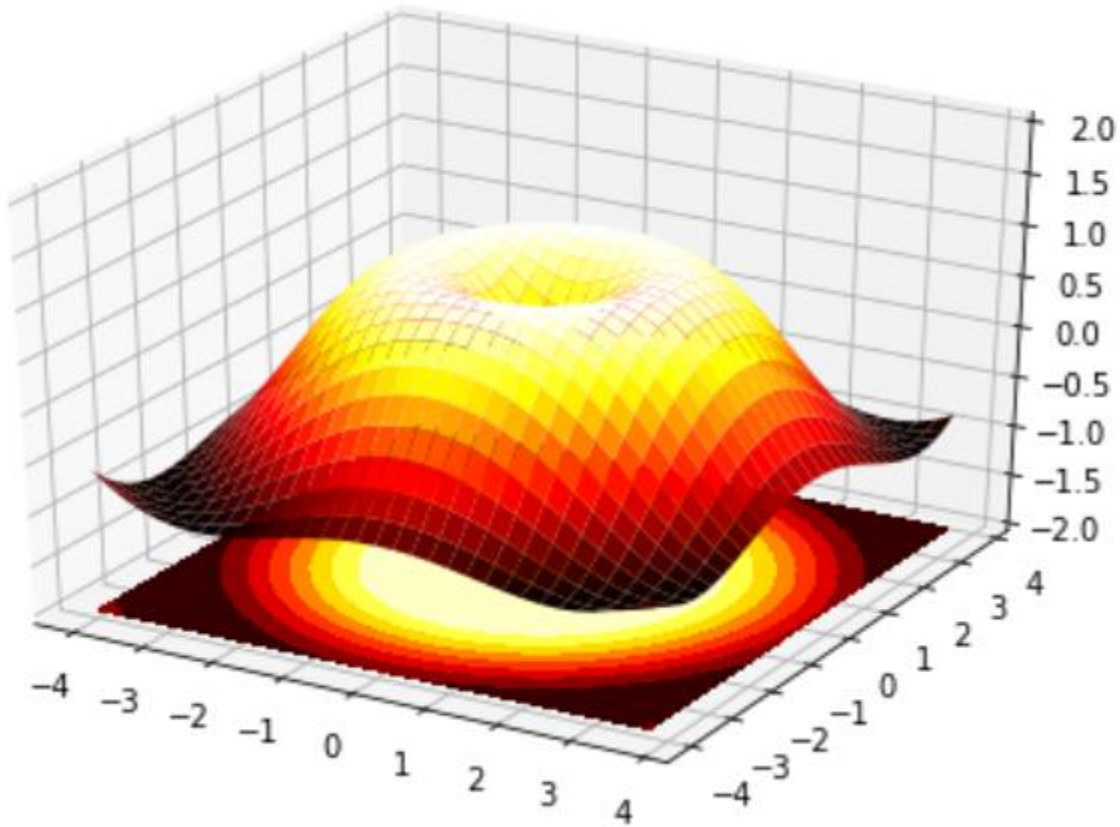
```
1  # plot14.py
2  import numpy as np
3  import matplotlib.pyplot as plt
4
5  def f(x,y):
6      return (1 - x / 2 + x**5 + y**3) * np.exp(-x**2 -y**2)
7
8  n = 256
9  x = np.linspace(-3, 3, n)
10 y = np.linspace(-3, 3, n)
11 X,Y = np.meshgrid(x, y)
12
13 plt.axes([0.025, 0.025, 0.95, 0.95])
14
15
16
17
18
```

```
1  plt.contourf(X, Y, f(X, Y), 8, alpha=.75, cmap=plt.cm.hot)
2  C = plt.contour(X, Y, f(X, Y), 8, colors='black', linewidth=0.5)
3  plt.clabel(C, inline=1, fontsize=10)
4
5  plt.xticks(())
6  plt.yticks(())
7  plt.show()
8
9
10
11
12
13
14
15
16
17
18
```


0.000
0.200
0.400
0.600
0.800
0.600
-0.400
-0.200
-0.200

```
1  # plot15.py
2  import numpy as np
3  import matplotlib.pyplot as plt
4
5  n = 20
6  Z = np.ones(n)
7  Z[-1] *= 2
8
9  plt.axes([0.025, 0.025, 0.95, 0.95])
10
11 plt.pie(Z, explode=Z*.05, colors = ['%f' % (i/float(n)) for i in range(n)])
12 plt.axis('equal')
13 plt.xticks(())
14 plt.yticks()
15
16 plt.show()
17
18
```

```
1  # plot16.py
2  import numpy as np
3  import matplotlib.pyplot as plt
4  from mpl_toolkits.mplot3d import Axes3D
5
6  fig = plt.figure()
7  ax = plt.axes(projection='3d')
8  X = np.arange(-4, 4, 0.25)
9  Y = np.arange(-4, 4, 0.25)
10 X, Y = np.meshgrid(X, Y)
11 R = np.sqrt(X ** 2 + Y ** 2)
12 Z = np.sin(R)
13
14 ax.plot_surface(X, Y, Z, rstride=1, cstride=1, cmap=plt.cm.hot)
15 ax.contourf(X, Y, Z, zdir='z', offset=-2, cmap=plt.cm.hot)
16 ax.set_zlim(-2, 2)
17
18 plt.show()
```


2.0
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−2.0
−4 −3 −2 −1 0 1 2 3 4
−4 −3 −2 −1 0 1 2 3 4