Chapter 14

Introduction to Matplotlib

Installation of Matplotlib

- Go to Python's subdirectory with **pip**, eg, …/Python/Scripts,

- With the internet being on, type the following commands in the prompt terminal:

pip install matplotlib

- Documentation: **http://matplotlib.org**

- In Anaconda

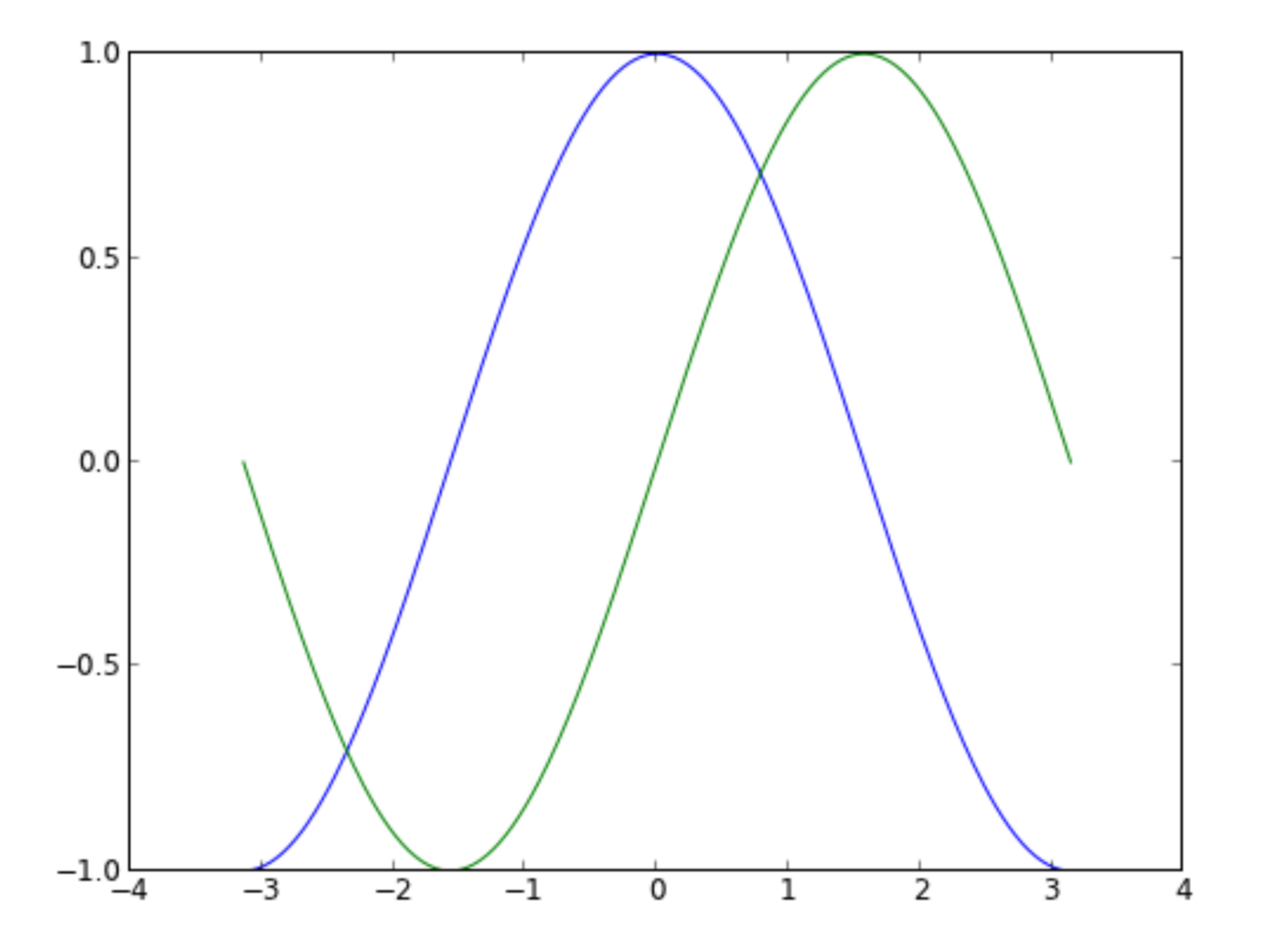
conda install matplotlib

Simple 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()
```

or

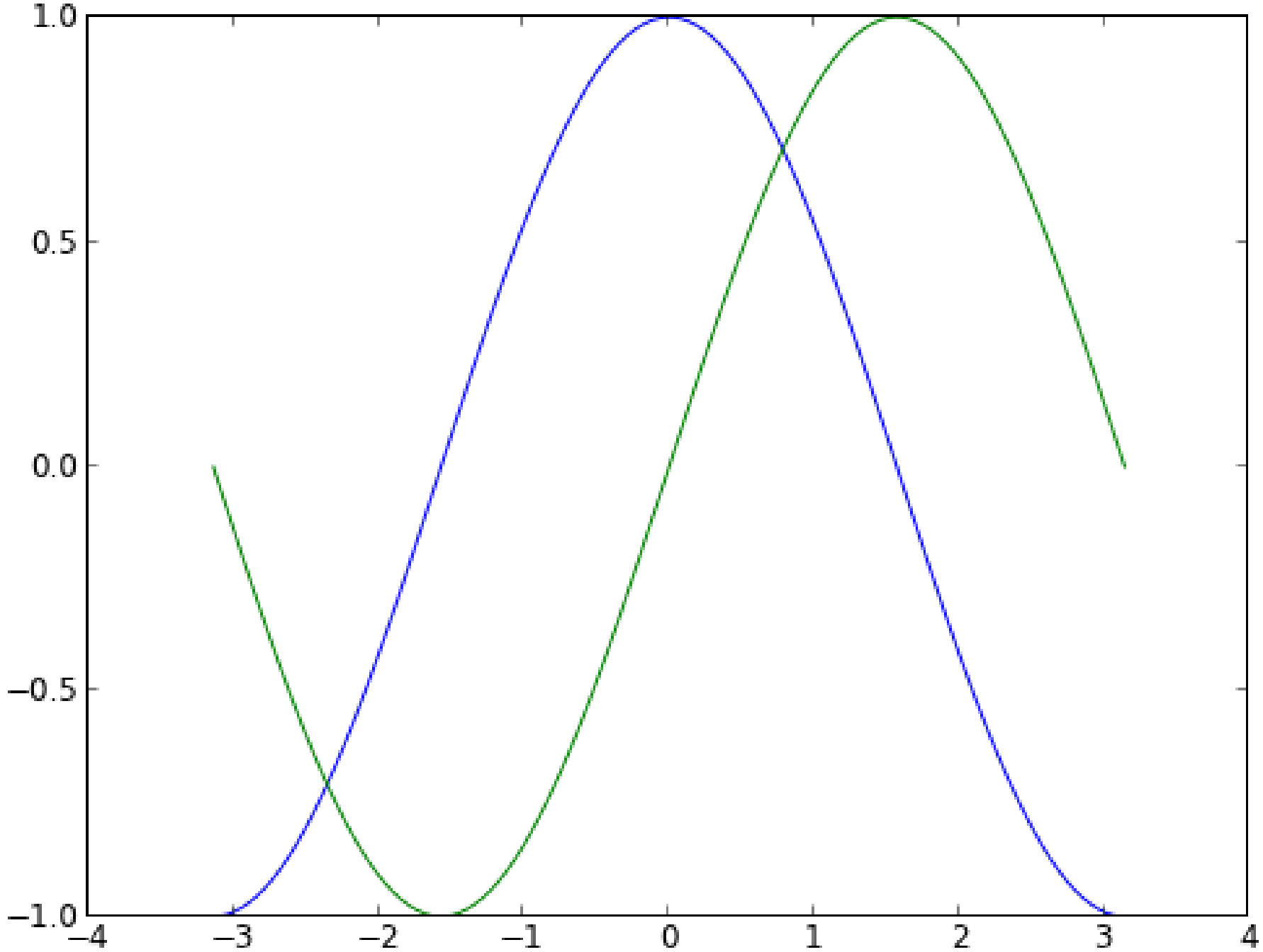
```
Import numpy as np
from pylab import *
X = np.linspace(-np.pi, np.pi, 256,endpoint=True)
C,S = np.cos(X), np.sin(X)
plot(X,C)
plot(X,S)
show()
```


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

Instantiating defaults

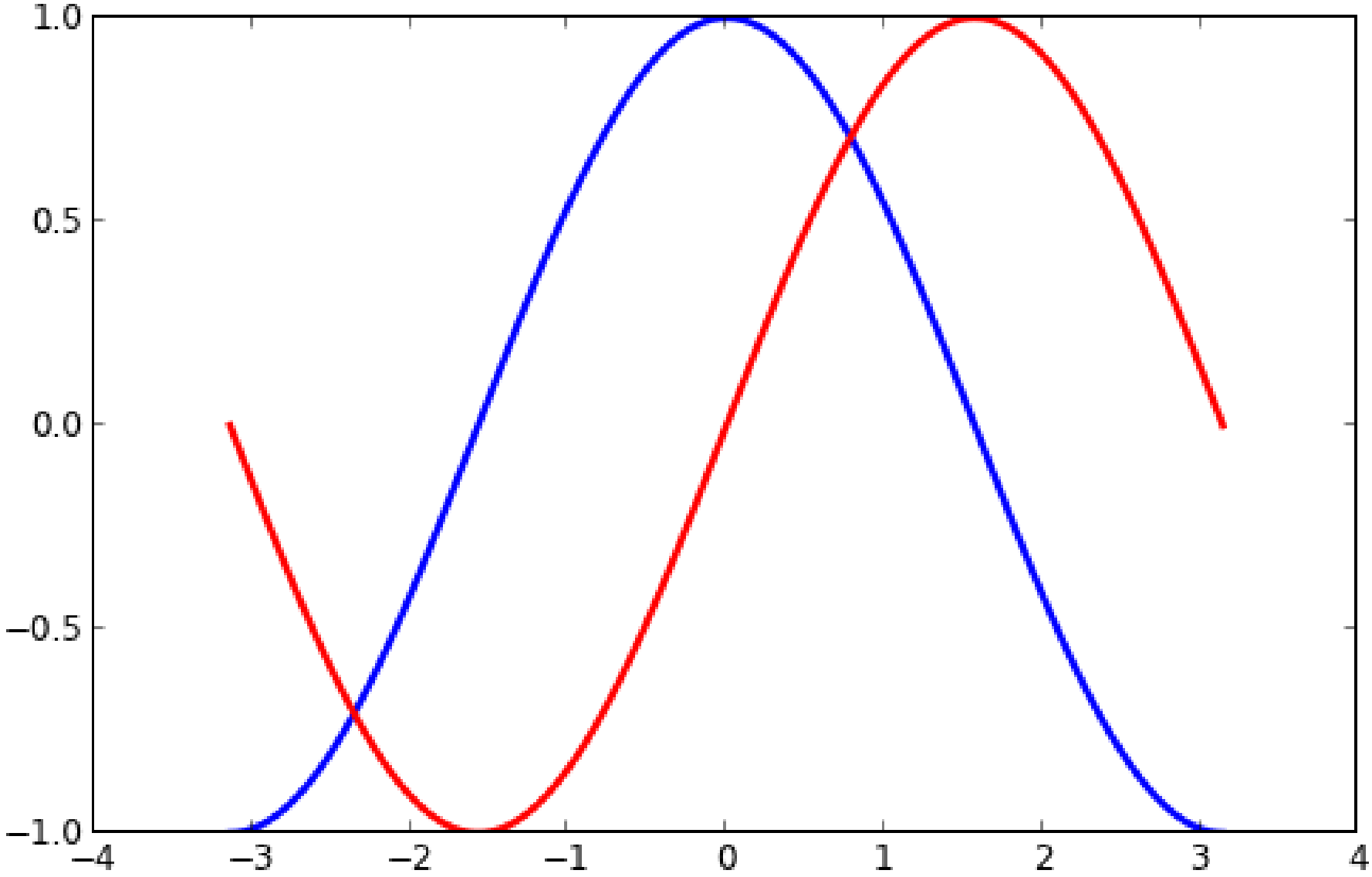
```
import numpy as np
from pylab import *

# Create a new figure of size 8x6 inches, 80 dots/inch
figure(figsize=(8,6), dpi=80)

# Create a new subplot from a grid of 1x1
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)
plot(X, C, color="blue", linewidth=1.0, linestyle="-")

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

```
# Set x limits
xlim(-4.0,4.0)

# Set x ticks
xticks(np.linspace(-4,4,9,endpoint=True))

# Set y limits
ylim(-1.0,1.0)

# Set y ticks
yticks(np.linspace(-1,1,5,endpoint=True))

# Save figure using 72 dots per inch
# savefig("exercice_2.png",dpi=72)

# Show result on screen
show()
```


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

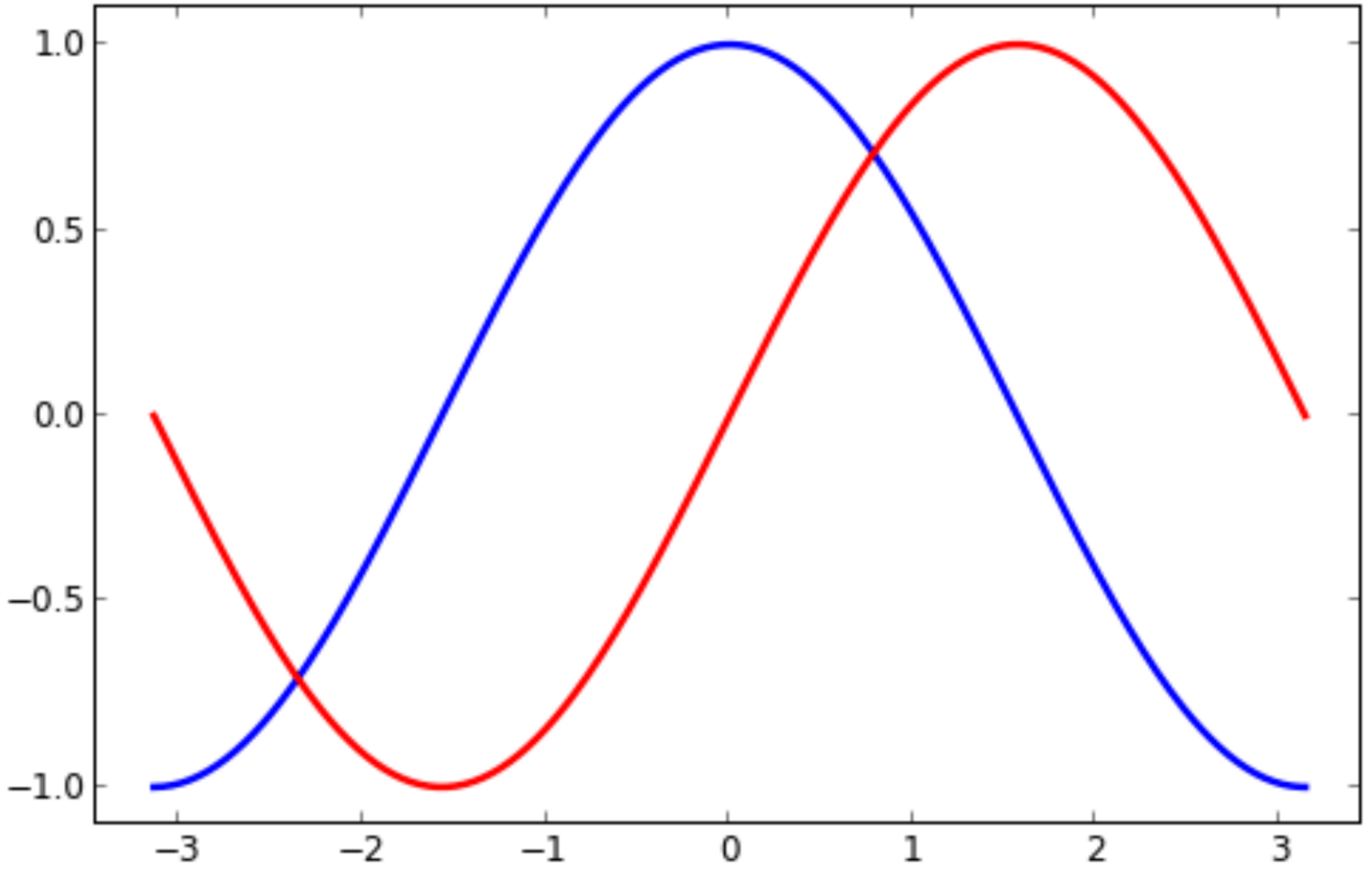
Changing colors and line widths

```
import numpy as np
from pylab import *

figure(figsize=(8,5), dpi=80)
subplot(111)

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

plot(X, C, color="blue", linewidth=2.5, linestyle="-")
plot(X, S, color="red",   linewidth=2.5, linestyle="-")

xlim(-4.0, 4.0)
xticks(np.linspace(-4,4,9,endpoint=True))

ylim(-1.0, 1.0)
yticks(np.linspace(-1,1,5,endpoint=True))

show()
```


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

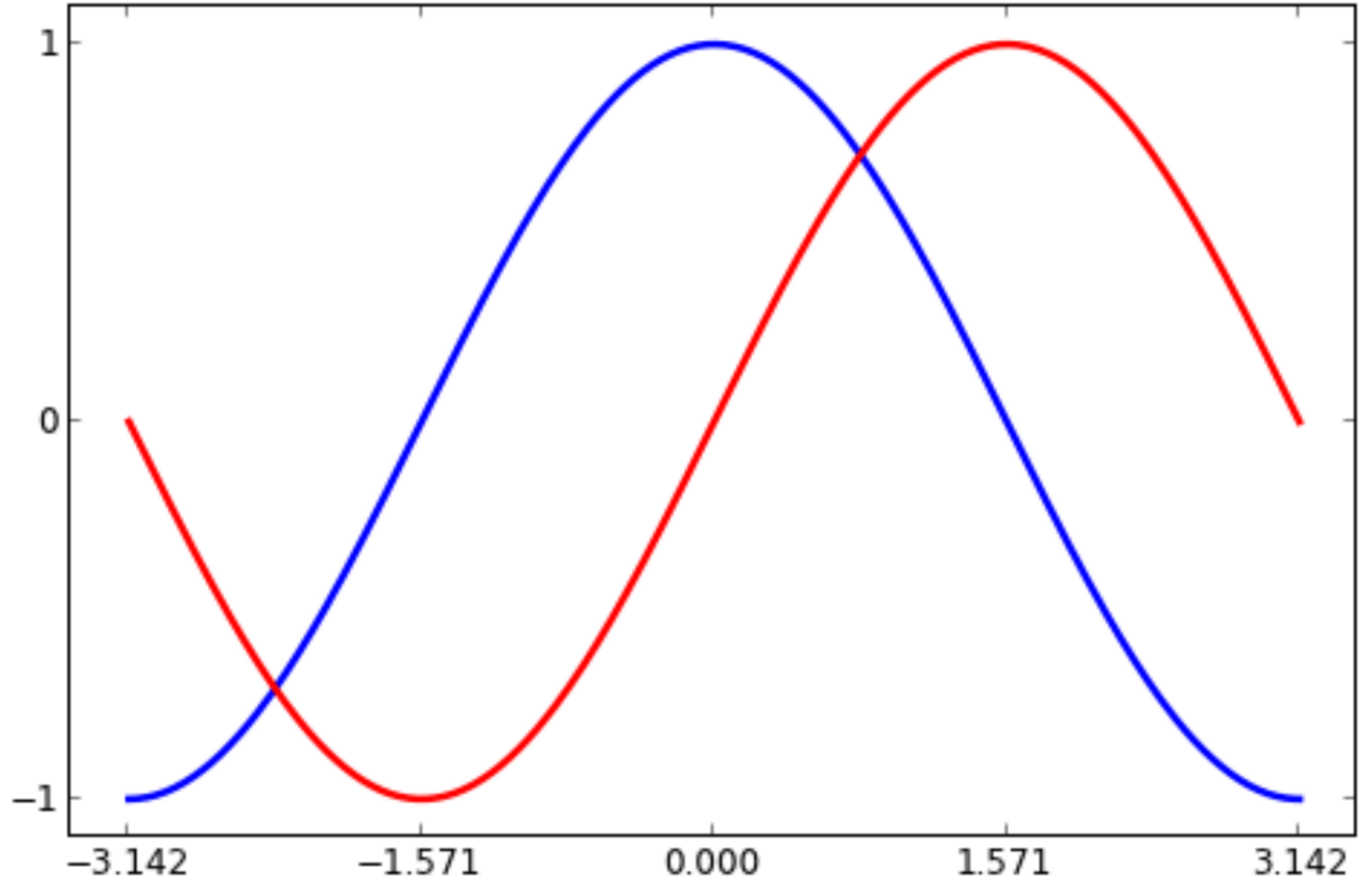
Setting limits

```
import numpy as np
from pylab import *

figure(figsize=(8,5), dpi=80)
subplot(111)

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

plot(X, C, color="blue", linewidth=2.5, linestyle="-")
plot(X, S, color="red",   linewidth=2.5, linestyle="-")

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

show()
```


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

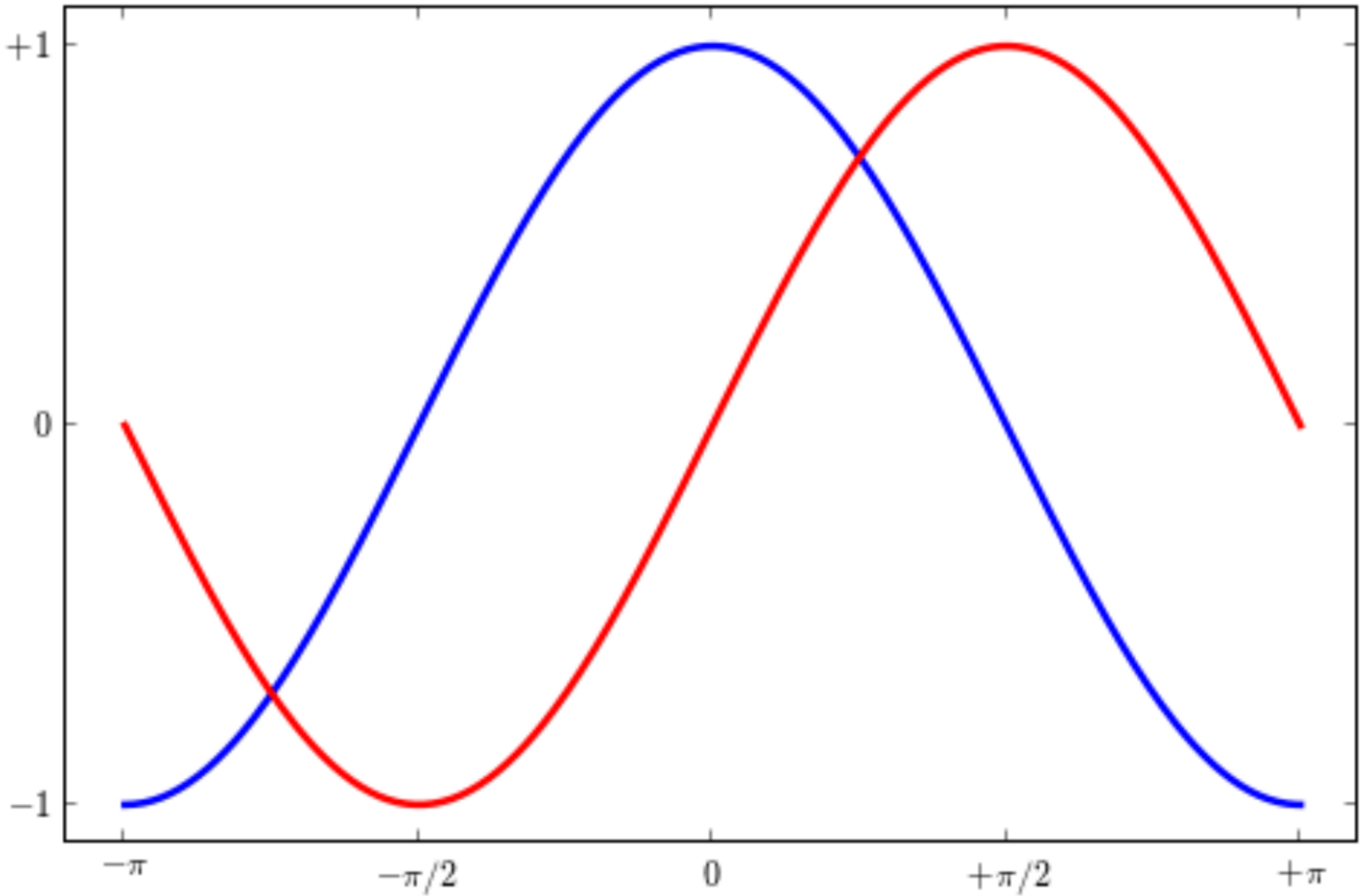
Setting ticks

```
import numpy as np
from pylab import *

figure(figsize=(8,5), dpi=80)
subplot(111)

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

plot(X, C, color="blue", linewidth=2.5, linestyle="-")
plot(X, S, color="red", linewidth=2.5, linestyle="-")

xlim(X.min()*1.1, X.max()*1.1)
xticks( [-np.pi, -np.pi/2, 0, np.pi/2, np.pi])

ylim(C.min()*1.1, C.max()*1.1)
yticks([-1, 0, +1])

show()
```


1
0
−1
−3.142
−1.571
0.000
1.571
3.142

Setting tick labels

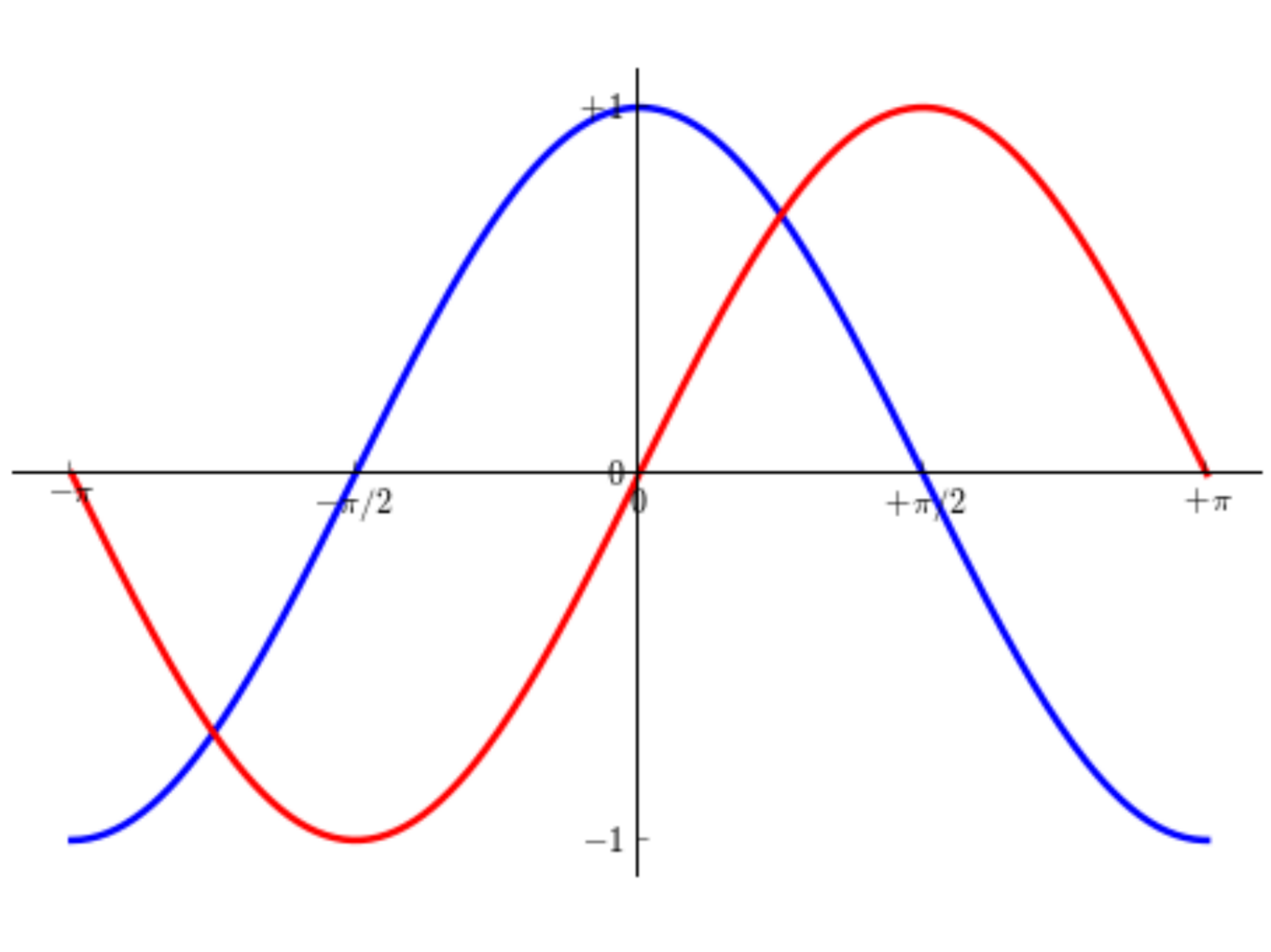
```
import numpy as np
from pylab import *

figure(figsize=(8,5), dpi=80)
subplot(111)

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

plot(X, C, color="blue", linewidth=2.5, linestyle="-")
plot(X, S, color="red", linewidth=2.5, linestyle="-")

xlim(X.min()*1.1, X.max()*1.1)
xticks([-np.pi, -np.pi/2, 0, np.pi/2, np.pi],
       [r'$-\pi$', r'$-\pi/2$', r'$0$', r'$+\pi/2$', r'$+\pi$'])
ylim(C.min()*1.1, C.max()*1.1)
yticks([-1, 0, +1], [r'$-1$', r'$0$', r'$+1$'])

show()
```


+1
0
−1
−π
−π/2
0
+π/2
+π

Moving spines

```
import numpy as np
from pylab import *

figure(figsize=(8,5), dpi=80)
ax = subplot(111)

ax.spines['right'].set_color('none')
ax.spines['top'].set_color('none')
ax.xaxis.set_ticks_position('bottom')
ax.spines['bottom'].set_position(('data',0))
ax.yaxis.set_ticks_position('left')
ax.spines['left'].set_position(('data',0))

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

plot(X, C, color="blue", linewidth=2.5, linestyle="-")
plot(X, S, color="red", linewidth=2.5, linestyle="-")
```

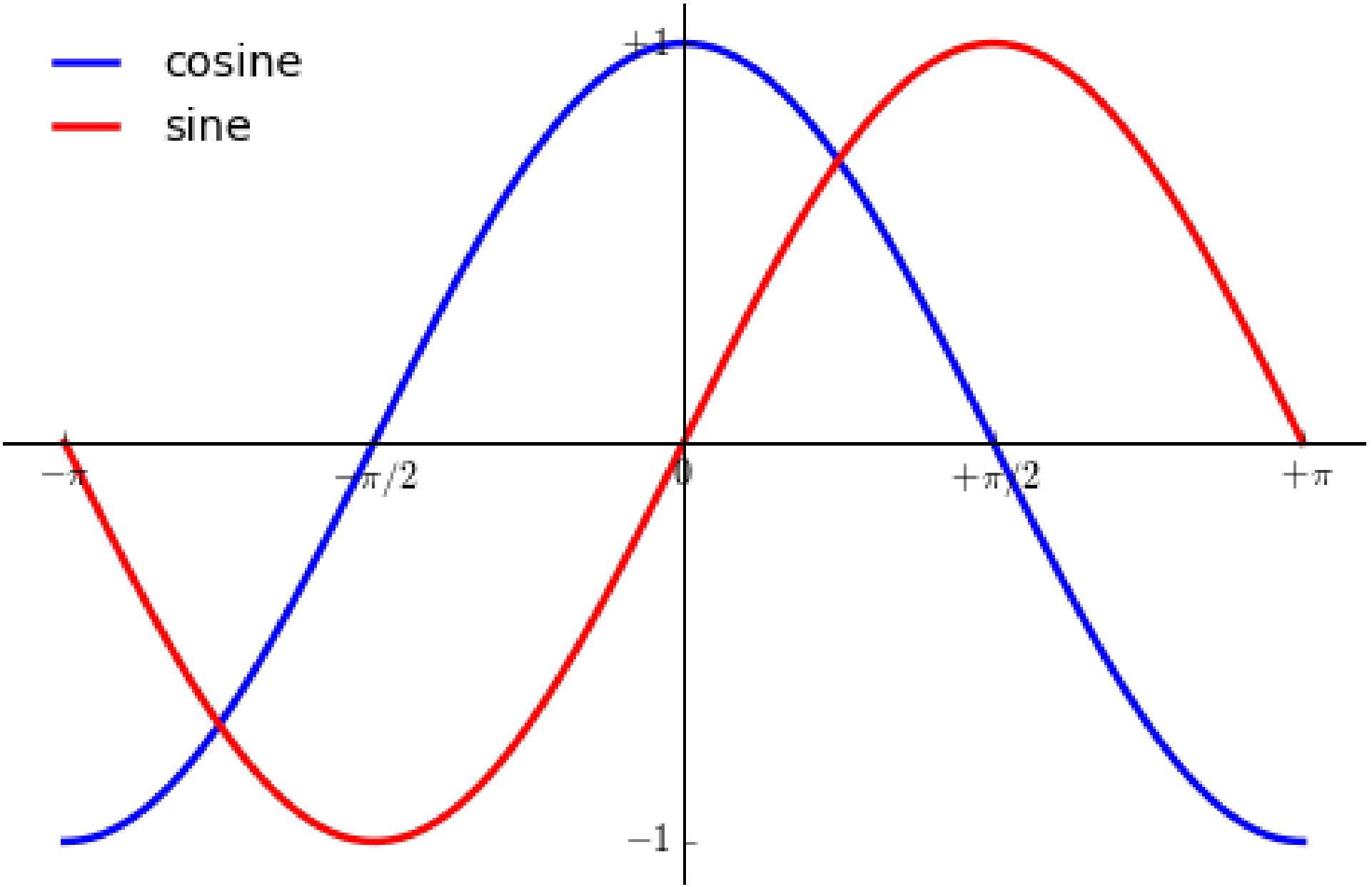
```
xlim(X.min()*1.1, X.max()*1.1)
xticks([-np.pi, -np.pi/2, 0, np.pi/2, np.pi],
       [r'$-\pi$', r'$-\pi/2$', r'$0$', r'$+\pi/2$', r'$+\pi$'])
ylim(C.min()*1.1, C.max()*1.1)
yticks([-1, 0, +1], [r'$-1$', r'$0$', r'$+1$'])

show()
```


+1
0
−1
−π
−π/2
0
+π/2
+π

Adding a legend

```
import numpy as np
from pylab import *

figure(figsize=(8,5), dpi=80)
ax = subplot(111)

ax.spines['right'].set_color('none')
ax.spines['top'].set_color('none')
ax.xaxis.set_ticks_position('bottom')
ax.spines['bottom'].set_position(('data',0))
ax.yaxis.set_ticks_position('left')
ax.spines['left'].set_position(('data',0))

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

```
plot(X, C, color="blue", linewidth=2.5, linestyle="-",
                                    label="cosine")
plot(X, S, color="red",   linewidth=2.5, linestyle="-",
                                    label="sine")

xlim(X.min()*1.1, X.max()*1.1)
xticks([-np.pi, -np.pi/2, 0, np.pi/2, np.pi],
       [r'$-\pi$', r'$-\pi/2$', r'$0$', r'$+\pi/2$', r'$+\pi$'])
ylim(C.min()*1.1, C.max()*1.1)
yticks([-1, 0, +1], [r'$-1$', r'$0$', r'$+1$'])

legend(loc='upper left')
show()
```

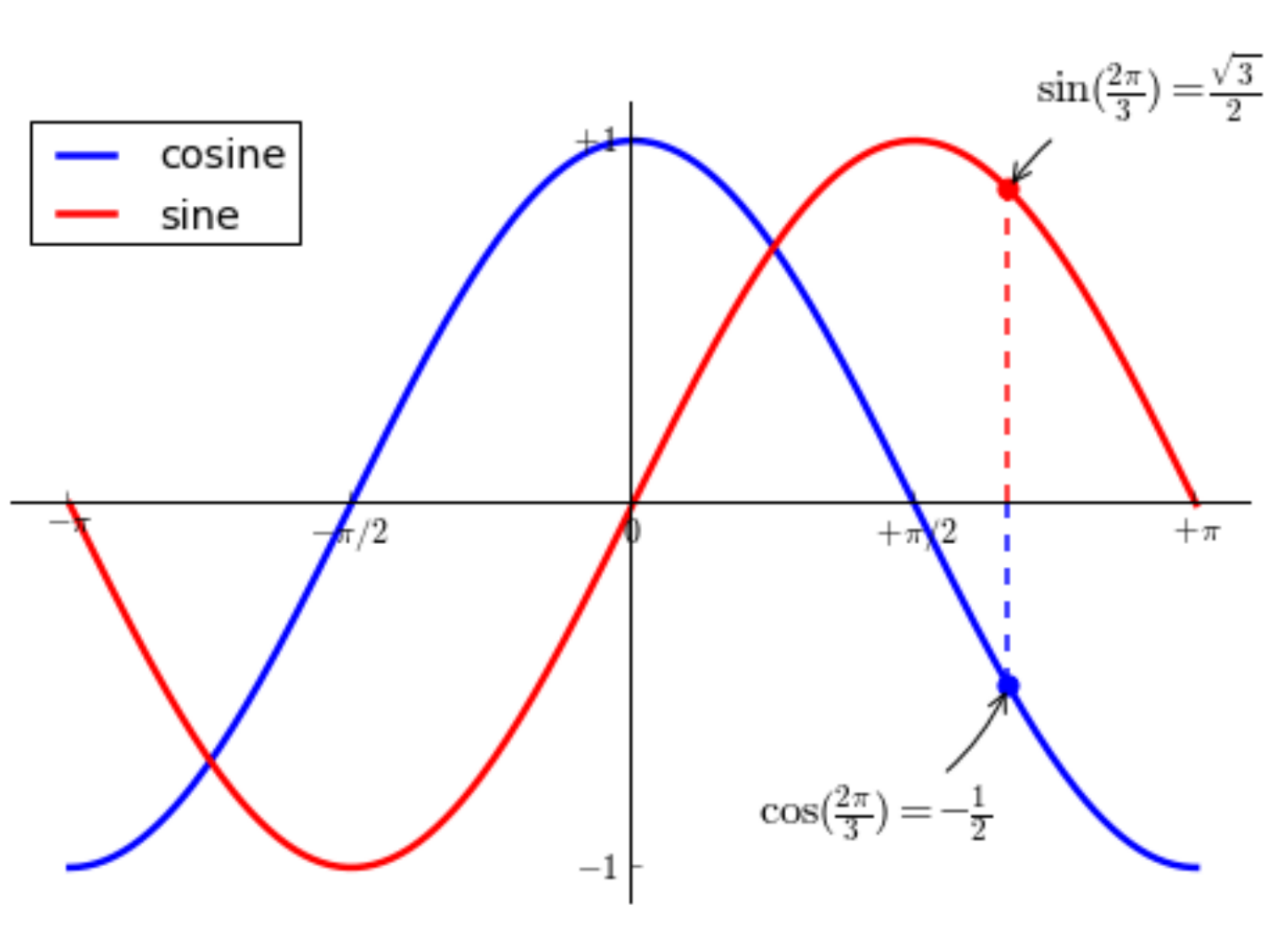

cosine
sine
+1
−1
−π
−π/2
0
+π/2
+π

Annotate some points

```
import numpy as np
from pylab import *

figure(figsize=(8,5), dpi=80)
ax = subplot(111)

ax.spines['right'].set_color('none')
ax.spines['top'].set_color('none')
ax.xaxis.set_ticks_position('bottom')
ax.spines['bottom'].set_position(('data',0))
ax.yaxis.set_ticks_position('left')
ax.spines['left'].set_position(('data',0))

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

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

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

xlim(X.min()*1.1, X.max()*1.1)
xticks([-np.pi, -np.pi/2, 0, np.pi/2, np.pi],
       [r'$-\pi$', r'$-\pi/2$', r'$0$', r'$+\pi/2$', r'$+\pi$'])
ylim(C.min()*1.1, C.max()*1.1)
yticks([-1, 0, +1], [r'$-1$', r'$0$', r'$+1$'])

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

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

legend(loc='upper left')
Show()
```

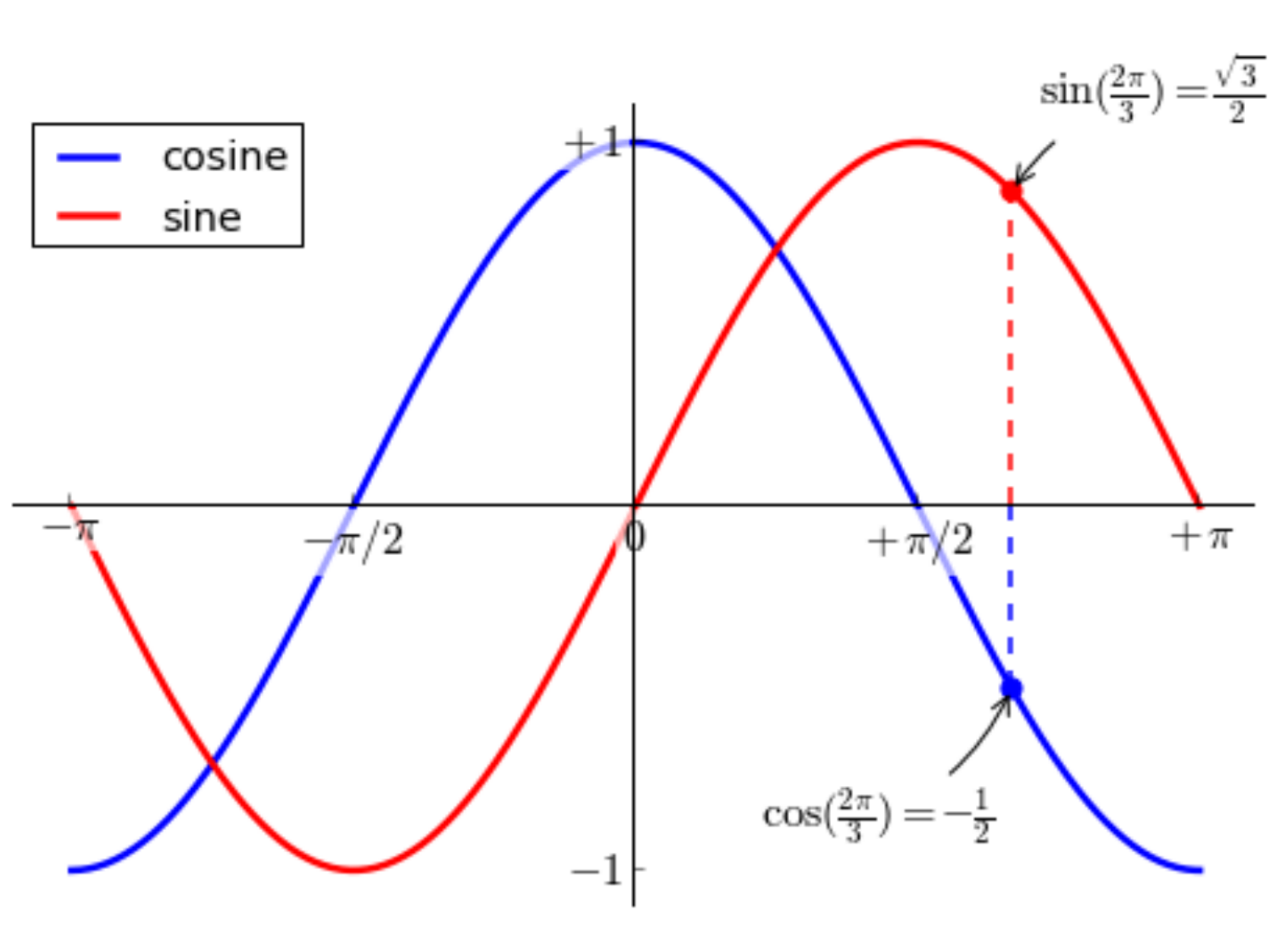

cosine
sine
$\sin(\frac{2\pi}{3}) = \frac{\sqrt{3}}{2}$
$\cos(\frac{2\pi}{3}) = -\frac{1}{2}$
+1
−1
$-\pi$
$-\pi/2$
0
$+\pi/2$
$+\pi$

Devil is in the details

```
import numpy as np
from pylab import *

figure(figsize=(8,5), dpi=80)
ax = subplot(111)

ax.spines['right'].set_color('none')
ax.spines['top'].set_color('none')
ax.xaxis.set_ticks_position('bottom')
ax.spines['bottom'].set_position(('data',0))
ax.yaxis.set_ticks_position('left')
ax.spines['left'].set_position(('data',0))

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

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

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

xlim(X.min()*1.1, X.max()*1.1)
xticks([-np.pi, -np.pi/2, 0, np.pi/2, np.pi],
       [r'$-\pi$', r'$-\pi/2$', r'$0$', r'$+\pi/2$', r'$+\pi$'])
ylim(C.min()*1.1, C.max()*1.1)
yticks([-1, 0, +1], [r'$-1$', r'$0$', r'$+1$'])

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

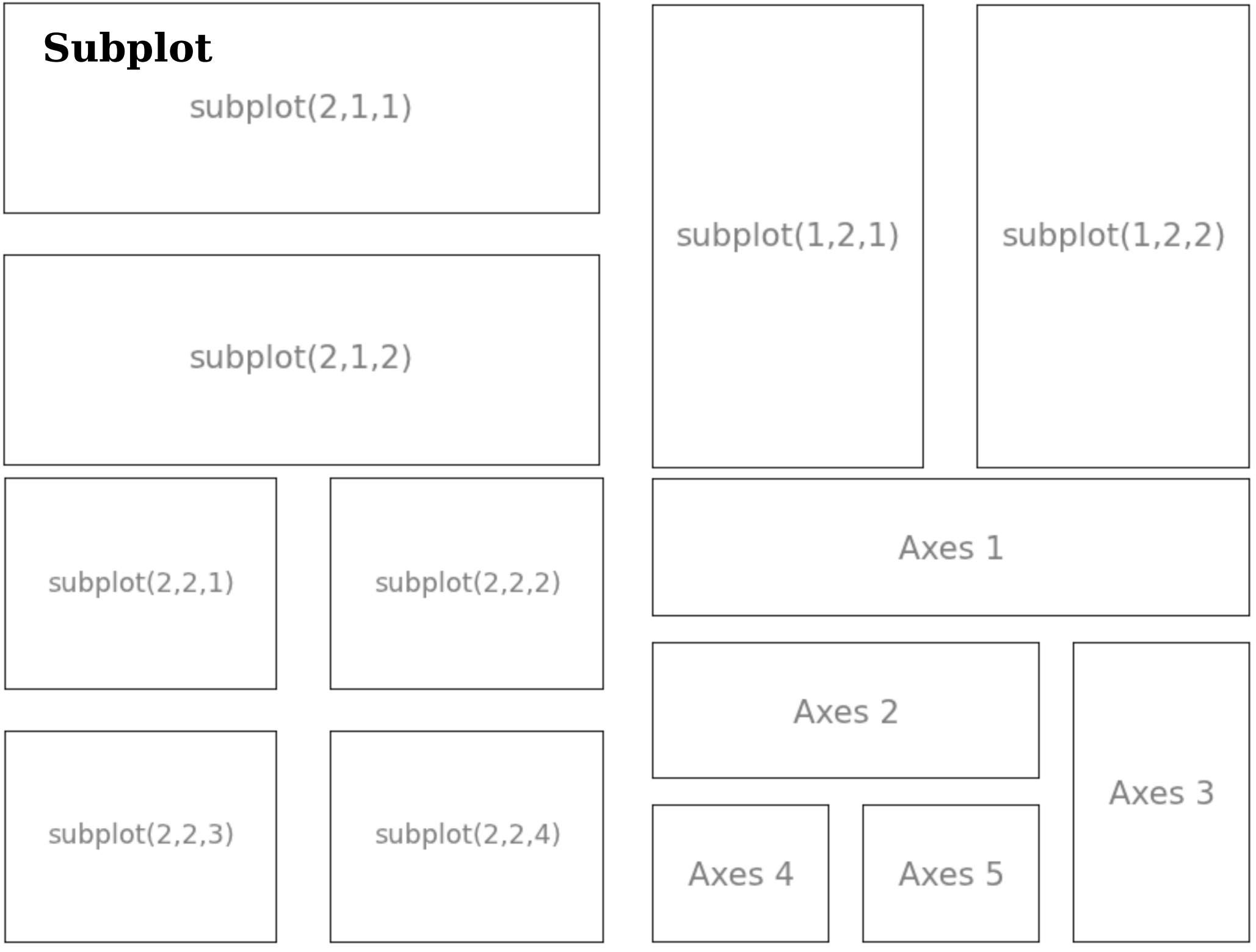
```
plot([t,t],[0,np.sin(t)], color ='red', linewidth=2.5,
                                        linestyle="--")
scatter([t,],[np.sin(t),], 50, color ='red')
annotate(r'$\cos(\frac{2\pi}{3})=-\frac{1}{2}$',
            xy=(t, np.cos(t)), xycoords='data',
            xytext=(-90, -50), textcoords='offset points',
            fontsize=16,arrowprops=dict(arrowstyle="->",
            connectionstyle="arc3, rad=.2"))

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

legend(loc='upper left')
Show()
```

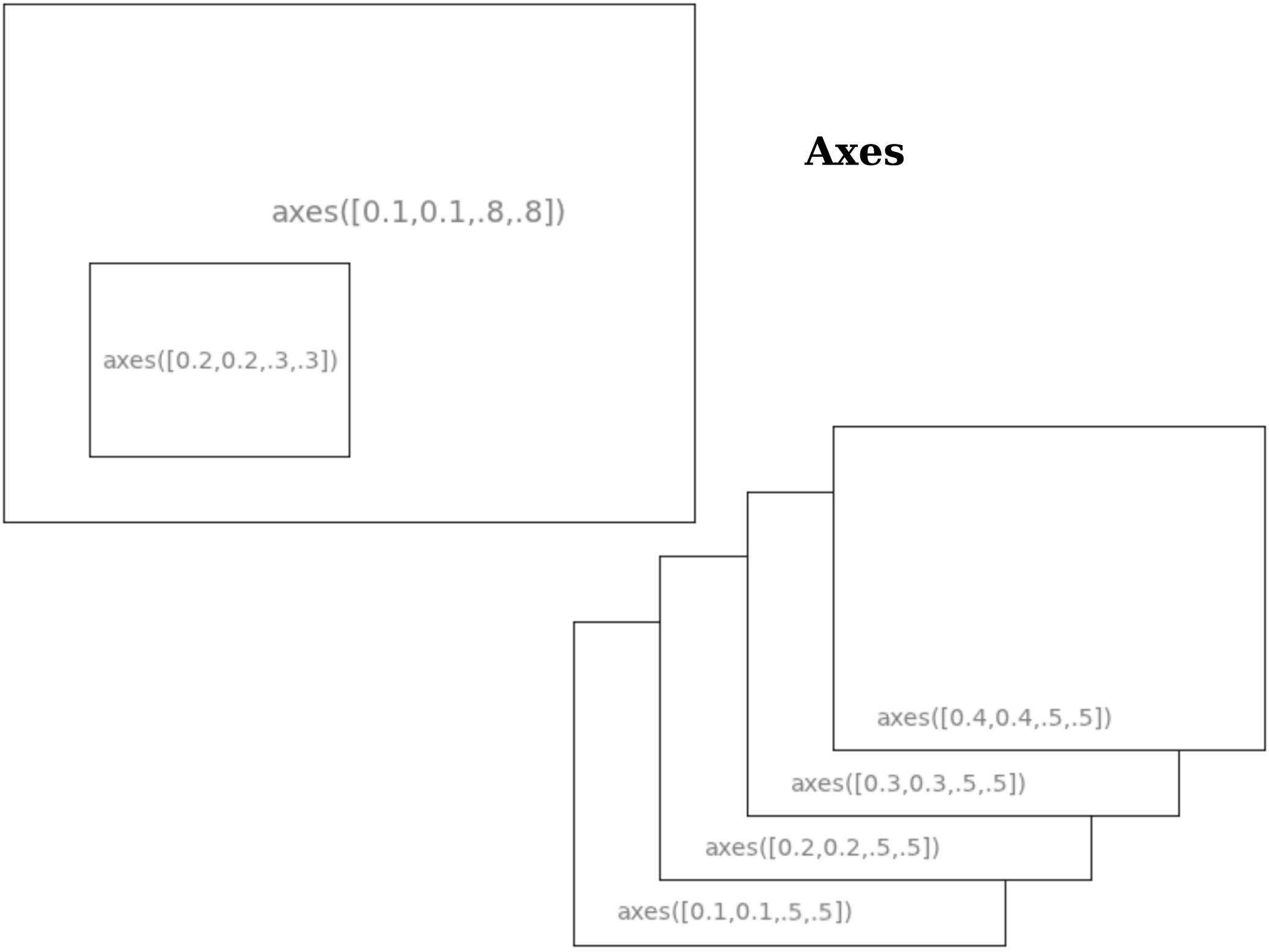

cosine
sine
$\sin(\frac{2\pi}{3}) = \frac{\sqrt{3}}{2}$
$+1$
-1
$-\pi$
$-\pi/2$
0
$+\pi/2$
$+\pi$
$\cos(\frac{2\pi}{3}) = -\frac{1}{2}$

Figures, Subplots, Axes and Ticks

Figures

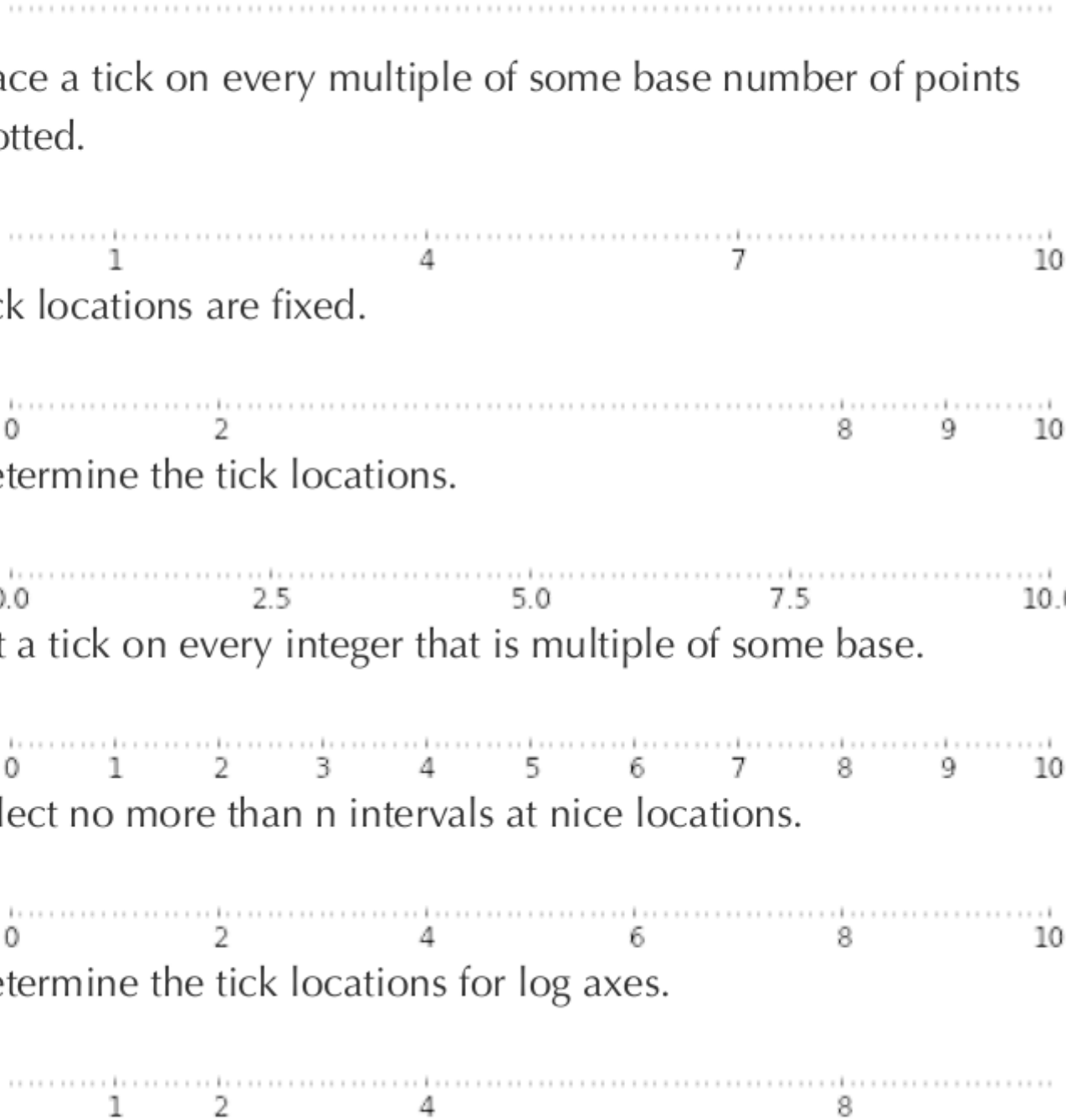
Argument	Default	Description
num	1	number of figure
figsize	figure.figsize	figure size in in inches (width, height)
dpi	figure.dpi	resolution in dots per inch
facecolor	figure.facecolor	color of the drawing background
edgecolor	figure.edgecolor	color of edge around the drawing background
frameon	True	draw figure frame or not


Subplot
subplot(2,1,1)
subplot(2,1,2)
subplot(2,2,1)
subplot(2,2,2)
subplot(2,2,3)
subplot(2,2,4)
subplot(1,2,1)
subplot(1,2,2)
Axes 1
Axes 2
Axes 3
Axes 4
Axes 5


Axes
axes([0.1,0.1,.8,.8])
axes([0.2,0.2,.3,.3])
axes([0.4,0.4,.5,.5])
axes([0.3,0.3,.5,.5])
axes([0.2,0.2,.5,.5])
axes([0.1,0.1,.5,.5])

Tick Locators

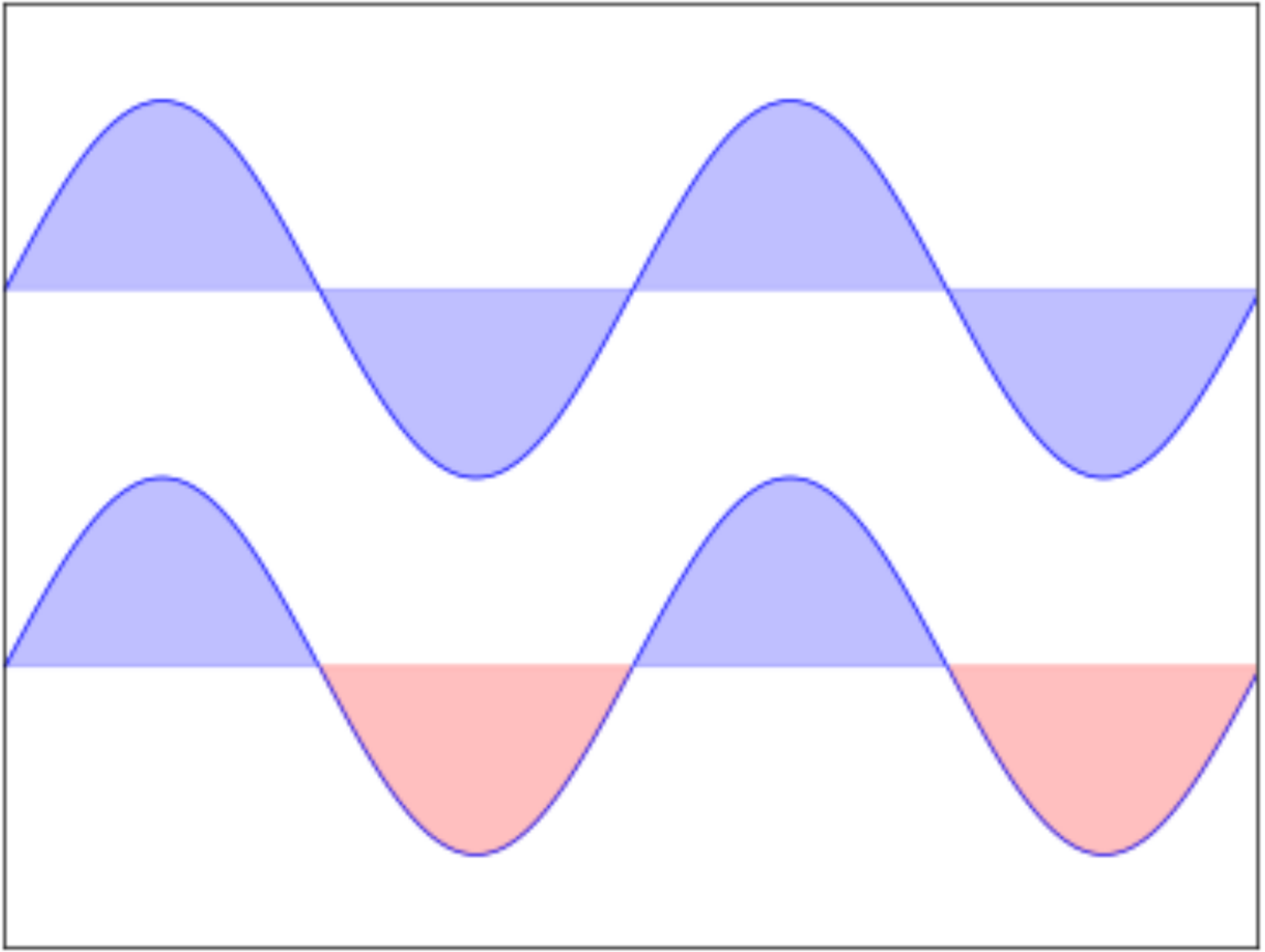
Class	Description
`NullLocator`	No ticks.
`IndexLocator`	Place a tick on every multiple of some base number of points plotted.
`FixedLocator`	Tick locations are fixed.
`LinearLocator`	Determine the tick locations.
`MultipleLocator`	Set a tick on every integer that is multiple of some base.
`AutoLocator`	Select no more than n intervals at nice locations.
`LogLocator`	Determine the tick locations for log axes.



Other Types of Plots

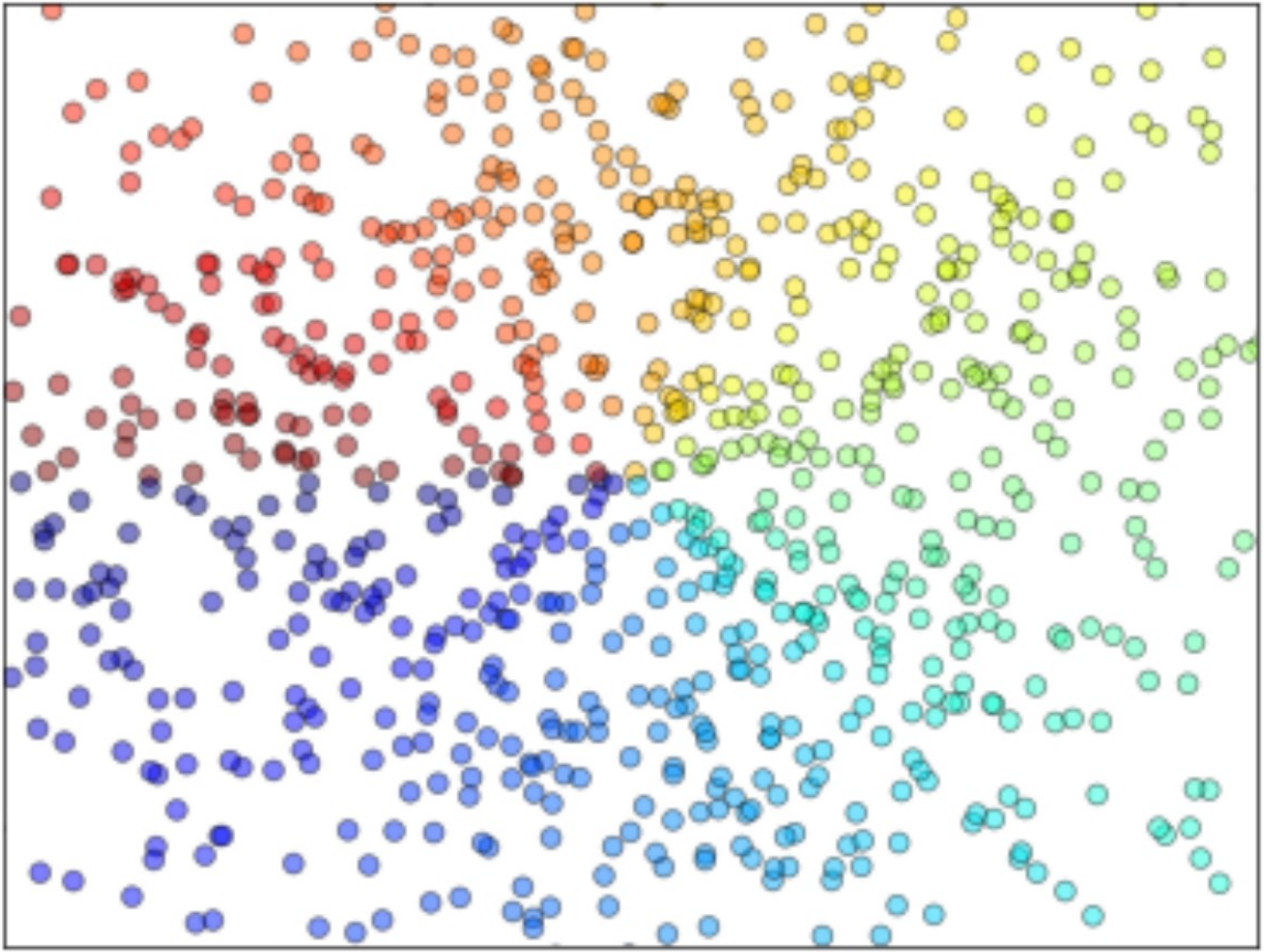
Regular Plots

```
import numpy as np
from pylab import *

X = np.linspace(-np.pi,np.pi,256,endpoint=True)
Y = np.sin(2*X)
axes([0.025,0.025,0.95,0.95])

plot(X, Y+1, color='blue', alpha=1.00)
fill_between(X, 1, Y+1, color='blue', alpha=.25)

plot(X, Y-1, color='blue', alpha=1.00)
fill_between(X,-1,Y-1,(Y-1)>-1, color='blue', alpha=.25)
fill_between(X,-1,Y-1,(Y-1)<-1, color='red',   alpha=.25)

xlim(-np.pi,np.pi), xticks([])
ylim(-2.5    ,2.5    ), yticks([])
show()
```



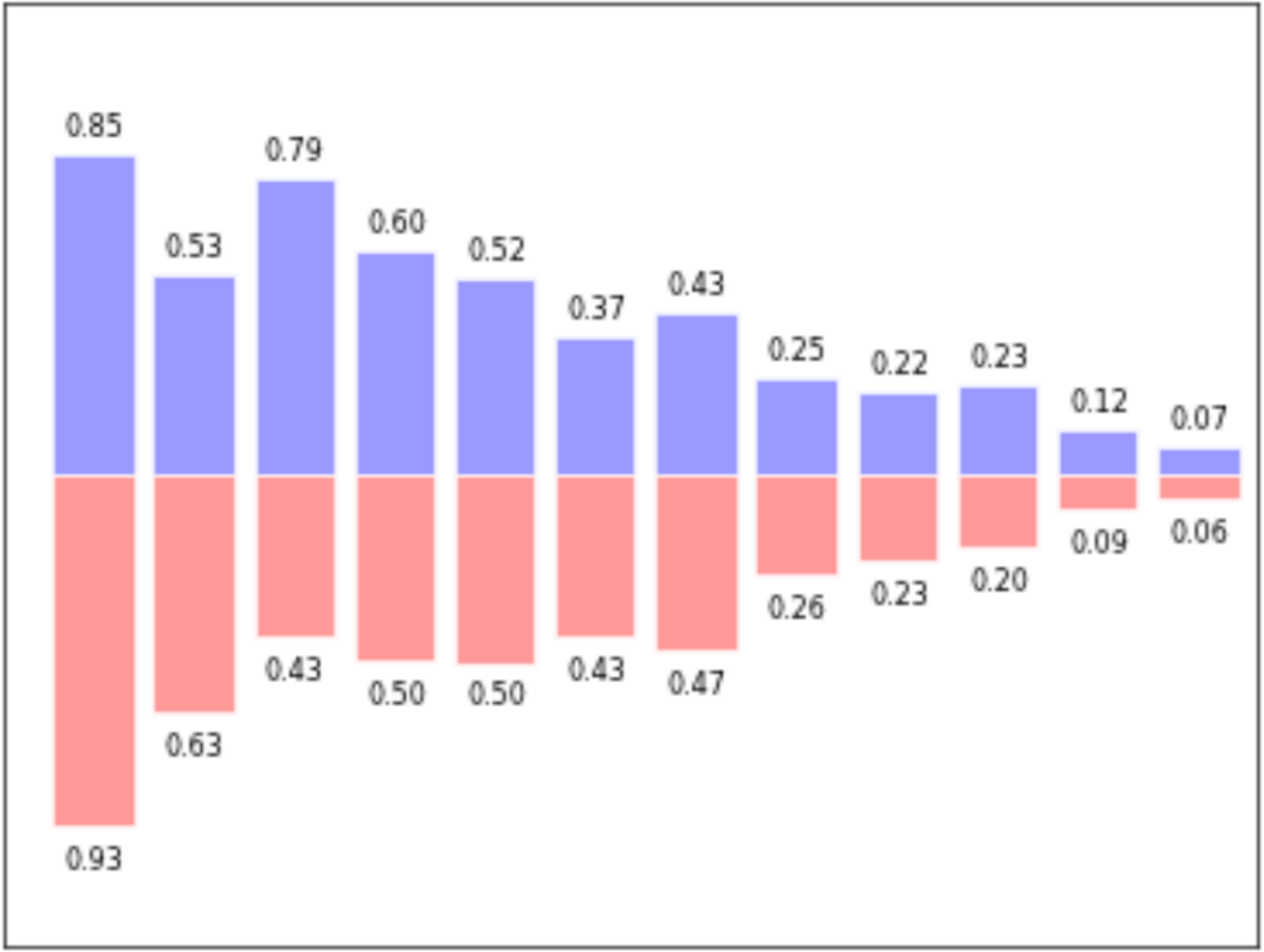
Scatter Plots

```
import numpy as np
from pylab import *

X = np.random.normal(0,1,1024)
Y = np.random.normal(0,1,1024)
T = np.arctan2(Y,X)

axes([0.025,0.025,0.95,0.95])
scatter(X,Y, s=75, c=T, alpha=.5)

xlim(-1.5,1.5), plt.xticks([])
ylim(-1.5,1.5), plt.yticks([])
show()
```



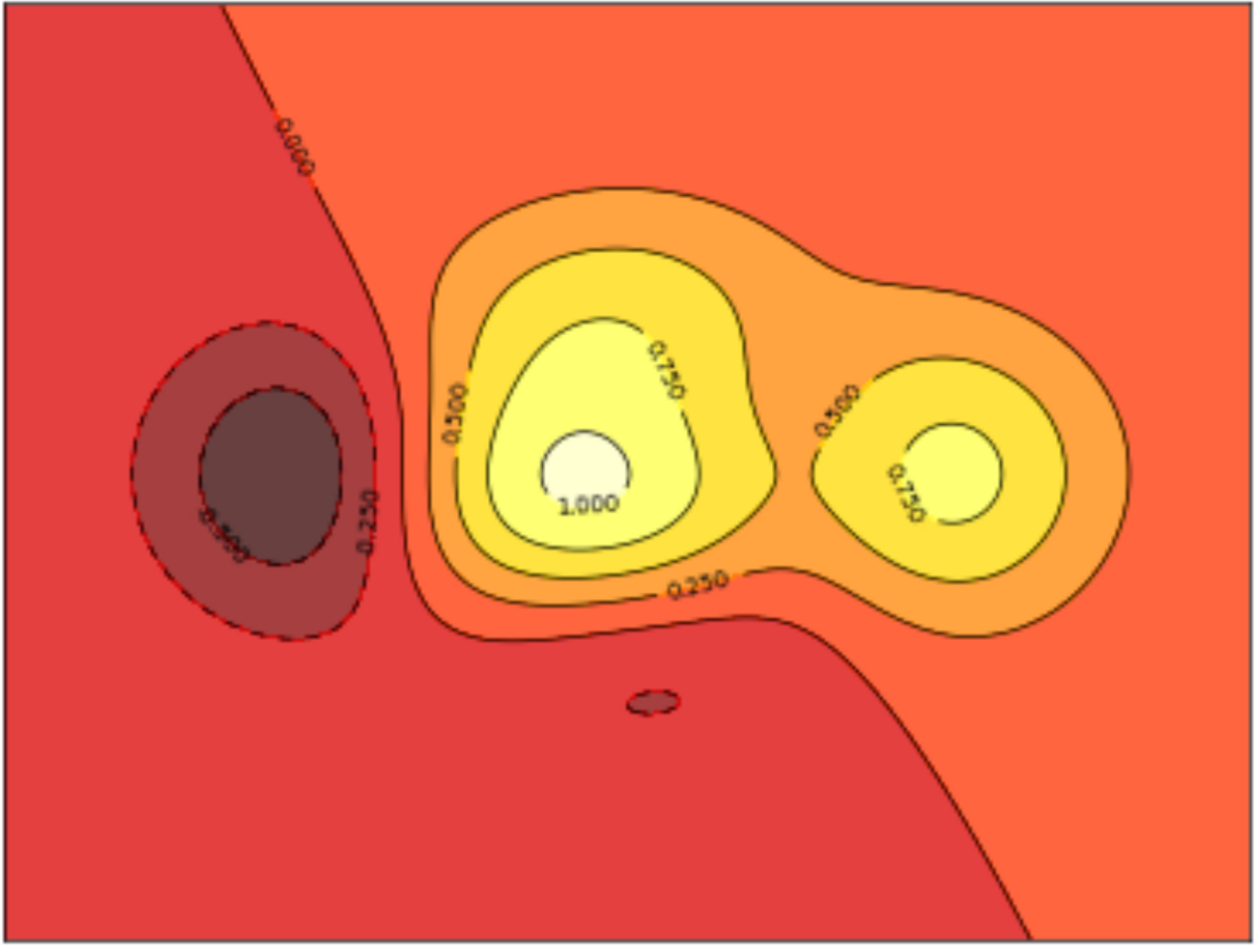
Bar Plots

```
import numpy as np
from pylab import *

X = np.arange(12)
Y1 = (1-X/float(12)) * np.random.uniform(0.5,1.0,12)
Y2 = (1-X/float(12)) * np.random.uniform(0.5,1.0,12)

axes([0.025,0.025,0.95,0.95])
bar(X, +Y1, facecolor='#9999ff', edgecolor='white')
bar(X,  -Y2, facecolor='#ff9999', edgecolor='white')

for x,y in zip(X,Y1):
    text(x+0.4,y+0.05,'%.2f' %y, ha='center',va='bottom')

for x,y in zip(X,Y2):
    text(x+0.4, -y-0.05, '%.2f' %y, ha='center', va= 'top')

xlim(-.5,12), xticks([])
ylim(-1.25,+1.25), yticks([])
show()
```


0.85
0.53
0.79
0.60
0.52
0.37
0.43
0.25
0.22
0.23
0.12
0.07
0.93
0.63
0.43
0.50
0.50
0.43
0.47
0.26
0.23
0.20
0.09
0.06

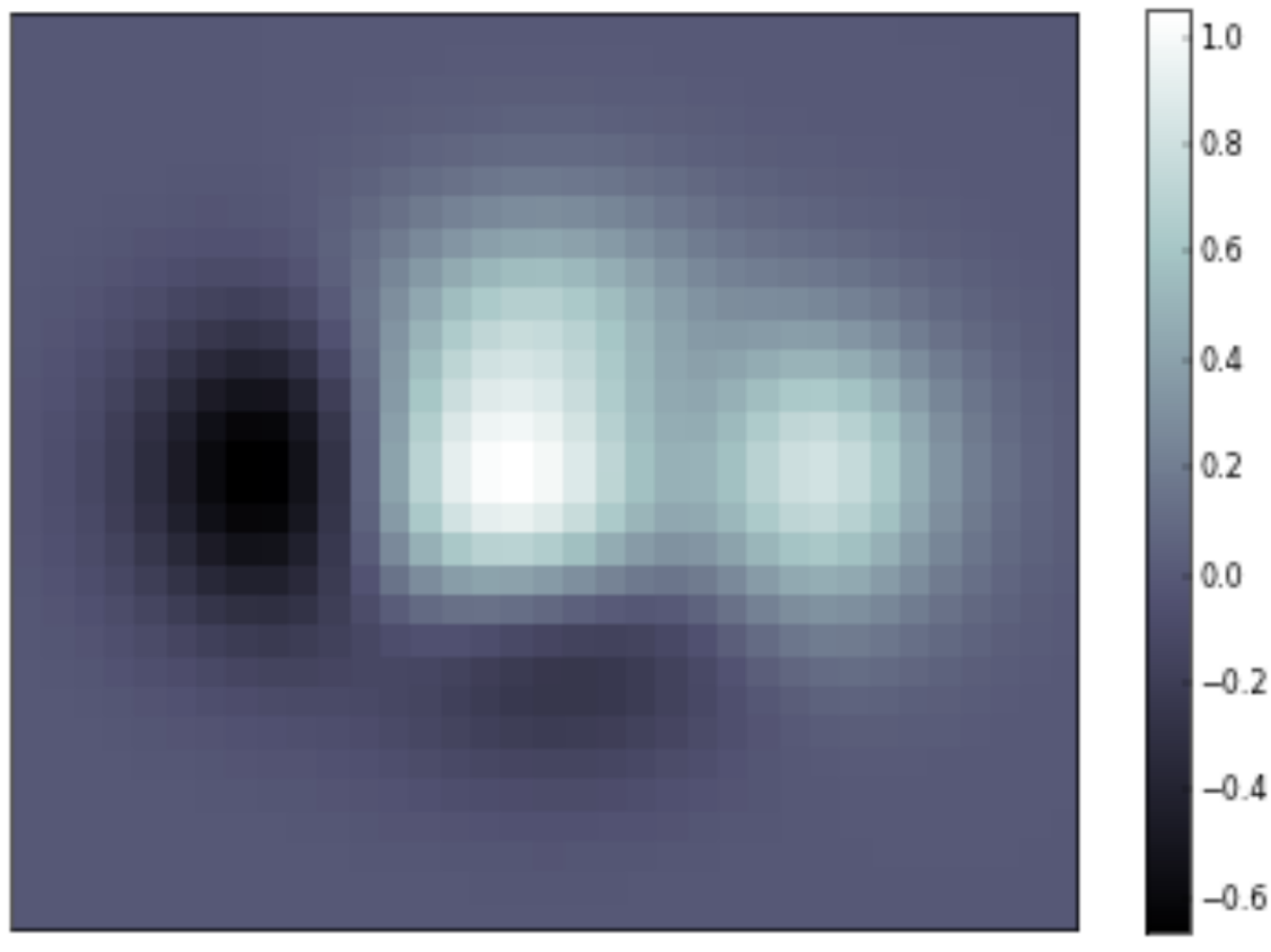
Contour Plots

```
import numpy as np
from pylab import *

def f(x,y):
    return (1-x/2+x**5+y**3)*np.exp(-x**2-y**2)

x = np.linspace(-3,3,256)
y = np.linspace(-3,3,256)
X,Y = np.meshgrid(x,y)

axes([0.025,0.025,0.95,0.95])

contourf(X, Y, f(X,Y), 8, alpha=.75, cmap=cm.hot)
C = contour(X, Y, f(X,Y), 8,colors='black', linewidth=.5)
clabel(C, inline=1, fontsize=10)

xticks([]), yticks([])
show()
```


0.000
0.500
0.750
1.000
0.500
0.750
0.250
0.250
0.500

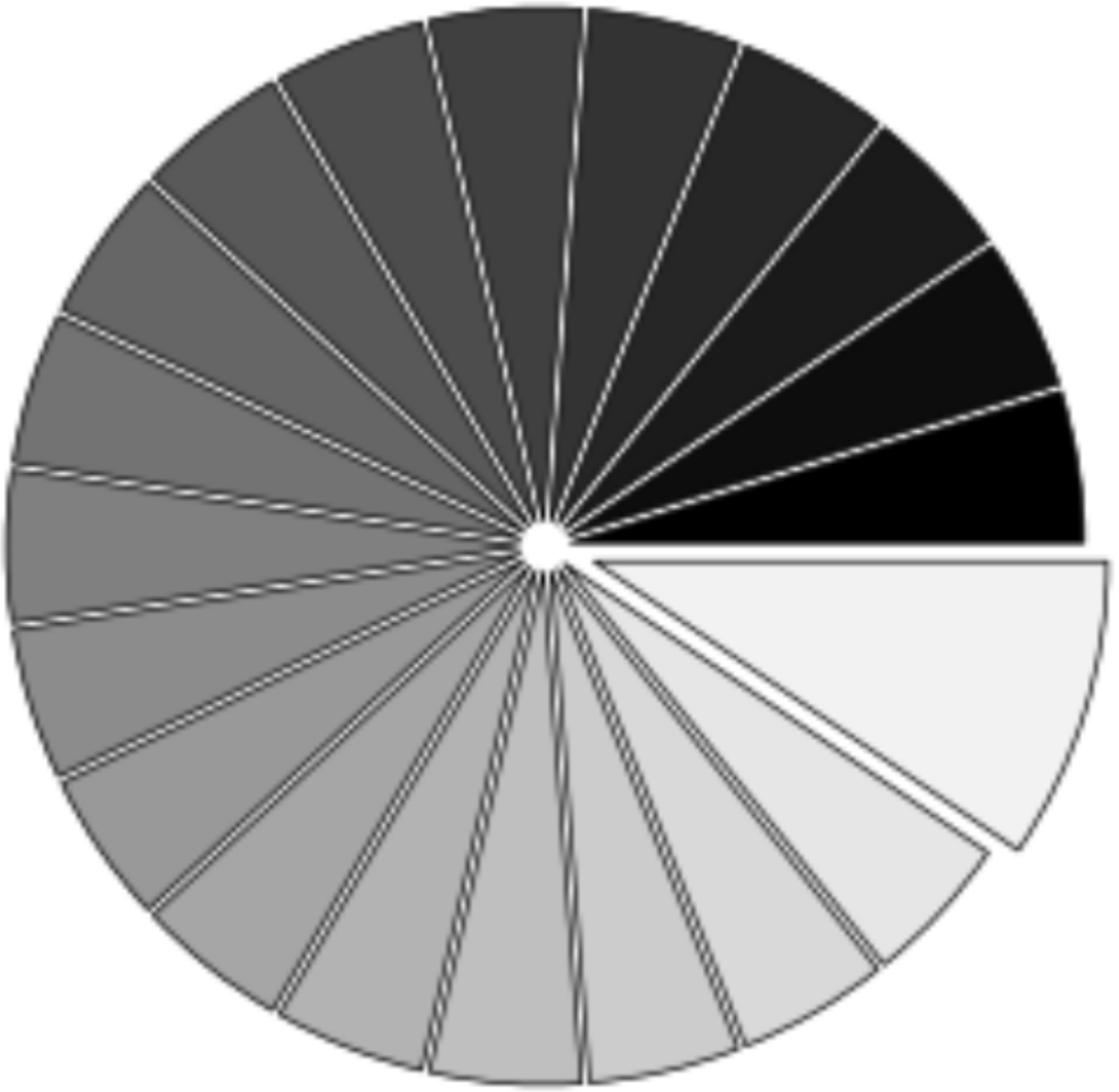
Imshow

```
import numpy as np
from pylab import *

def f(x,y):
    return (1-x/2+x**5+y**3)*np.exp(-x**2-y**2)

x = np.linspace(-3,3,3.5*10)
y = np.linspace(-3,3,3.0*10)
X,Y = np.meshgrid(x,y)
Z = f(X,Y)

axes([0.025,0.025,0.95,0.95])
imshow(Z,interpolation='nearest', cmap='bone',
            origin='lower')
colorbar(shrink=.92)

xticks([]), yticks([])
show()
```


1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4
−0.6

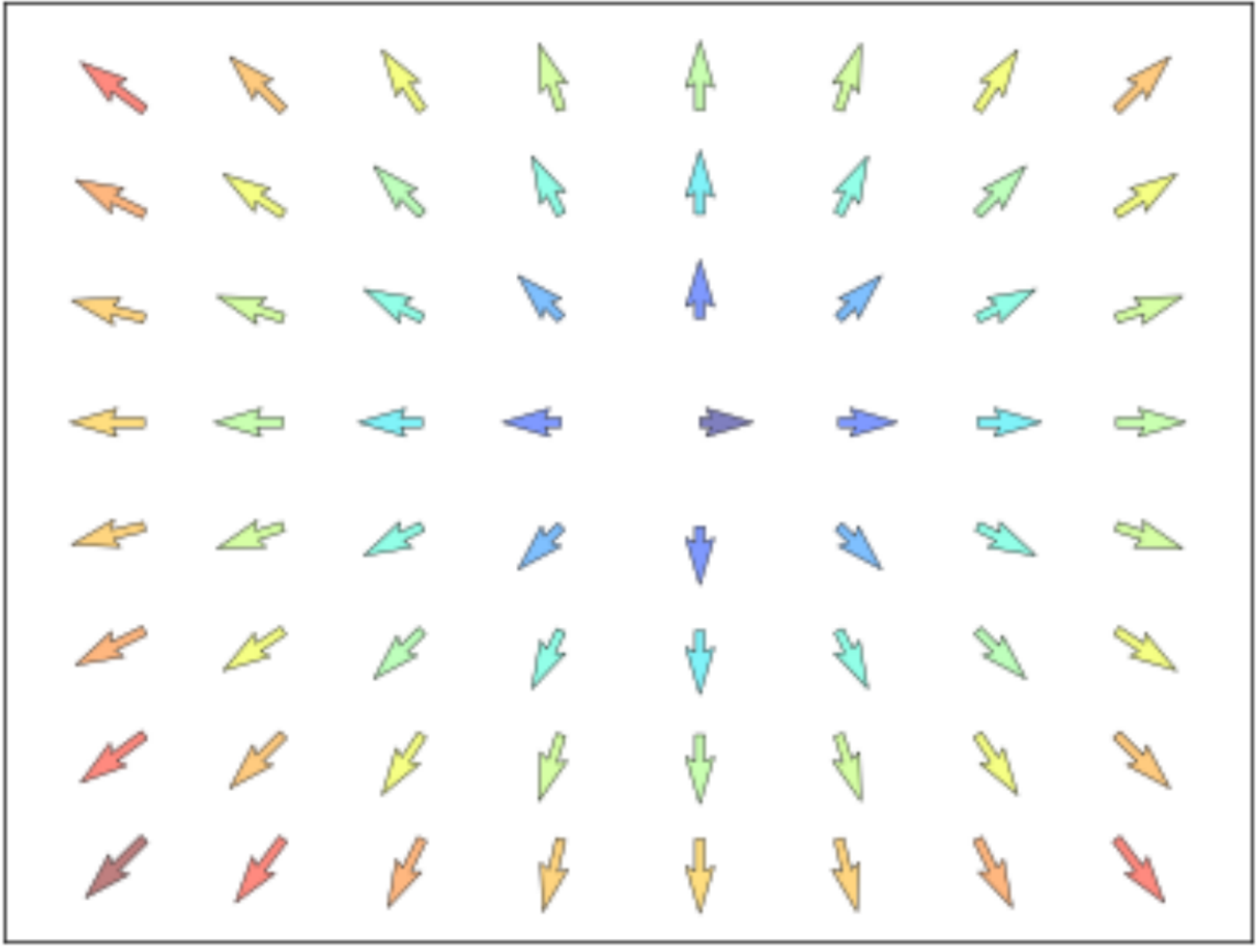
Pie Charts

```
import numpy as np
from pylab import *

n = 20
Z = np.ones(n)
Z[-1] *= 2

axes([0.025,0.025,0.95,0.95])

pie(Z, explode=Z*.05,
      colors = ['%f' % (i/float(n)) for i in range(n)])
gca().set_aspect('equal')

xticks([]), yticks([])
show()
```



Quiver Plots

```
import numpy as np
from pylab import *

n = 8
X,Y = np.mgrid[0:n,0:n]
T = np.arctan2(Y-n/2.0, X-n/2.0)
R = 10+np.sqrt((Y-n/2.0)**2+(X-n/2.0)**2)
U,V = R*np.cos(T), R*np.sin(T)

axes([0.025,0.025,0.95,0.95])
quiver(X,Y,U,V,R, alpha=.5)
quiver(X,Y,U,V, edgecolor='k', facecolor='None',
           linewidth=.5)

xlim(-1,n), xticks([])
ylim(-1,n), yticks([])
show()
```



Grids

```
from pylab import *
ax = axes([0.025,0.025,0.95,0.95])
ax.set_xlim(0,4)
ax.set_ylim(0,3)
ax.xaxis.set_major_locator(MultipleLocator(1.0))
ax.xaxis.set_minor_locator(MultipleLocator(0.1))
ax.yaxis.set_major_locator(MultipleLocator(1.0))
ax.yaxis.set_minor_locator(MultipleLocator(0.1))
ax.grid(which='major', axis='x', linewidth=0.75,
        linestyle='-', color='0.75')
ax.grid(which='minor', axis='x', linewidth=0.25,
        linestyle='-', color='0.75')
ax.grid(which='major', axis='y', linewidth=0.75,
        linestyle='-', color='0.75')
ax.grid(which='minor', axis='y', linewidth=0.25,
        linestyle='-', color='0.75')
ax.set_xticklabels([])
ax.set_yticklabels([])
show()
```

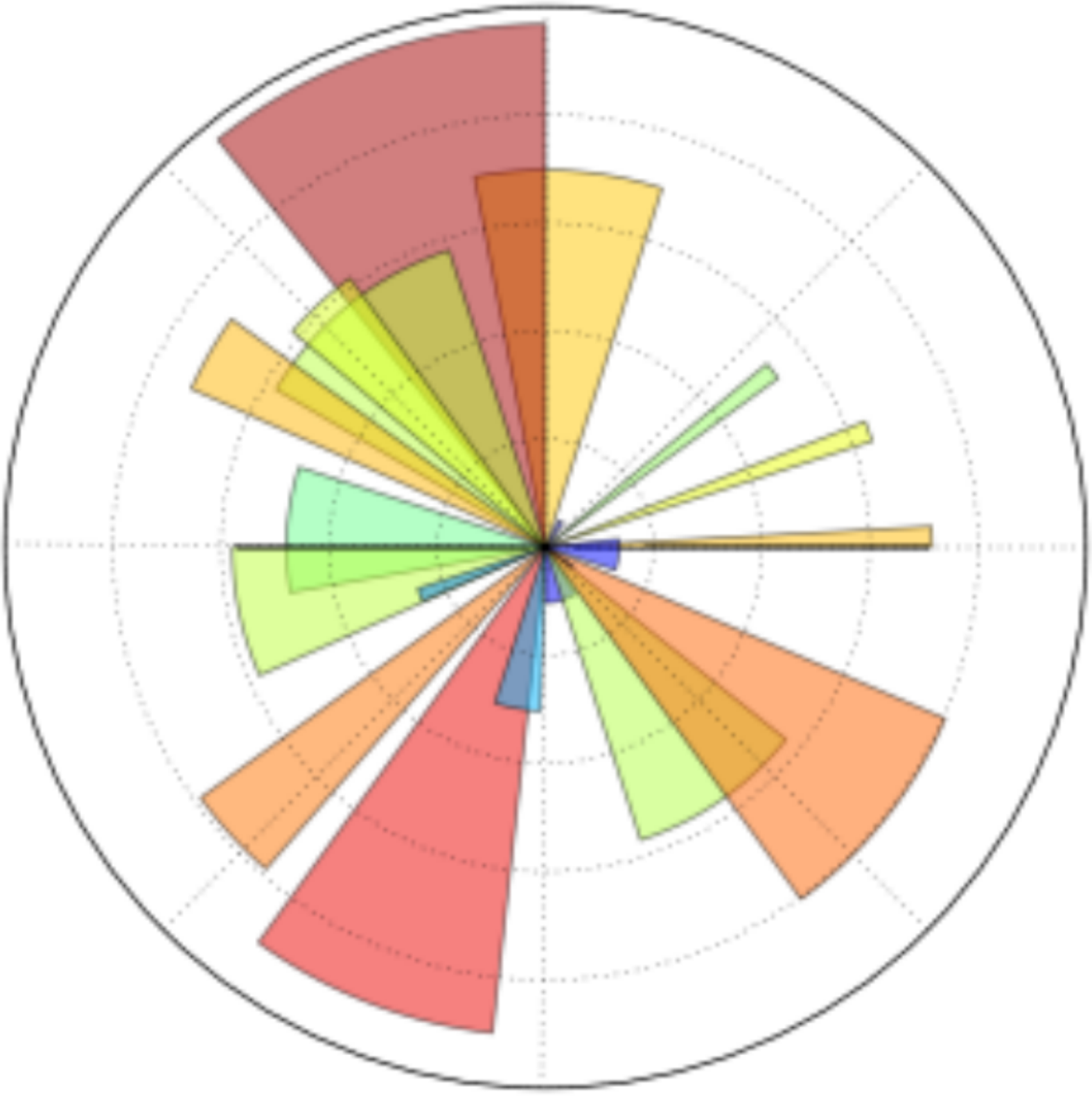
Multi Plots

```
from pylab import *

fig = figure()
subplots_adjust(bottom=0.025, left=0.025,
                        top = 0.975, right=0.975)

subplot(2,1,1)
xticks([]), yticks([])
subplot(2,3,4)
xticks([]), yticks([])
subplot(2,3,5)
xticks([]), yticks([])
subplot(2,3,6)
xticks([]), yticks([])

show()
```



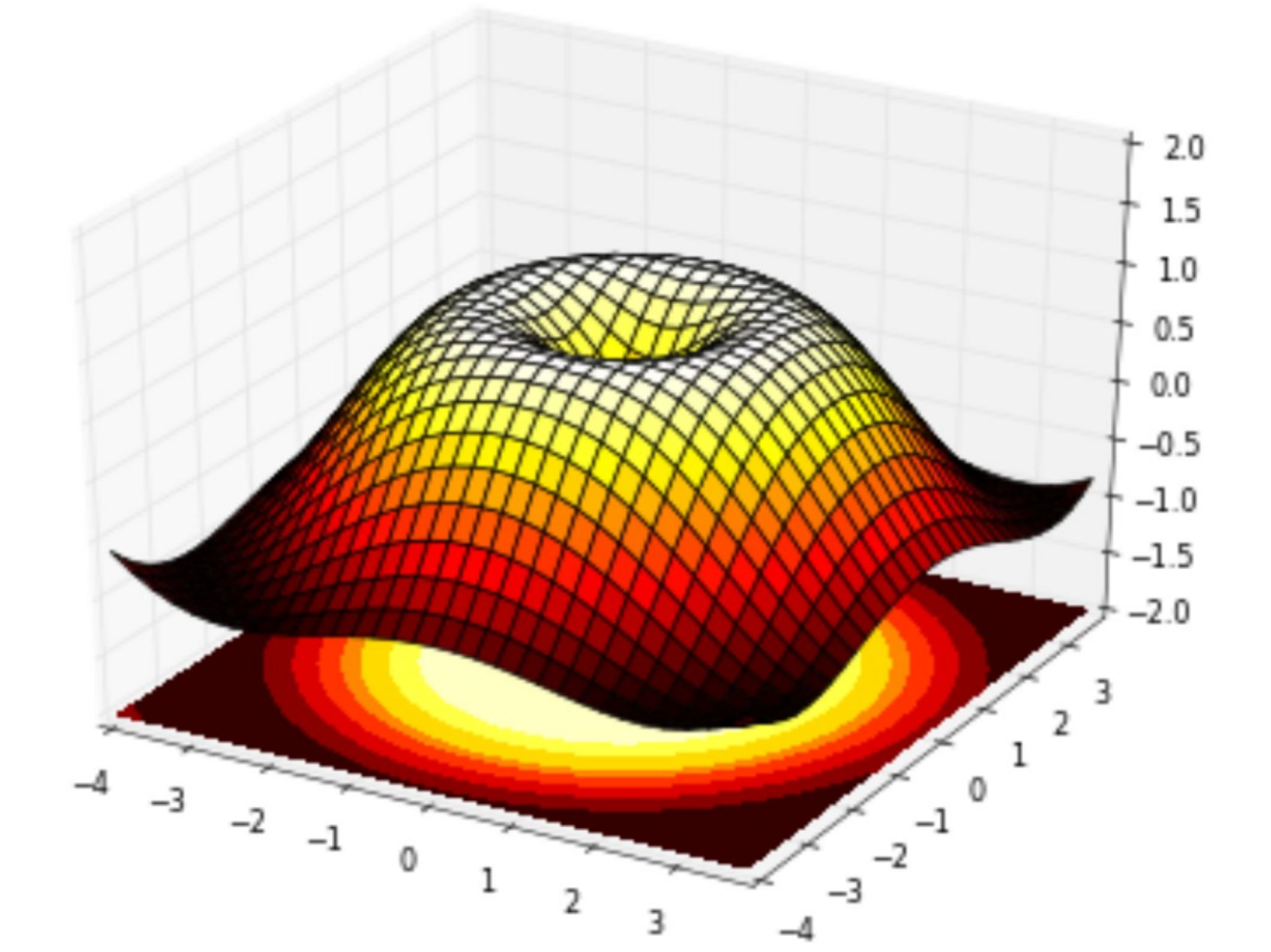

Polar Axis

```
import numpy as np
from pylab import *

ax = axes([0.025,0.025,0.95,0.95], polar=True)

N = 20
theta = np.arange(0.0, 2*np.pi, 2*np.pi/N)
radii = 10*np.random.rand(N)
width = np.pi/4*np.random.rand(N)
bars = bar(theta, radii, width=width, bottom=0.0)

for r,bar in zip(radii, bars):
    bar.set_facecolor( cm.jet(r/10.))
    bar.set_alpha(0.5)

ax.set_xticklabels([])
ax.set_yticklabels([])
show()
```



3D Plots

```
import numpy as np
from pylab import *
from mpl_toolkits.mplot3d import Axes3D

fig = figure()
ax = Axes3D(fig)
X = np.arange(-4, 4, 0.25)
Y = np.arange(-4, 4, 0.25)
X, Y = np.meshgrid(X, Y)
R = np.sqrt(X**2 + Y**2)
Z = np.sin(R)

ax.plot_surface(X, Y, Z, rstride=1, cstride=1,
                         cmap=plt.cm.hot)
ax.contourf(X, Y, Z, zdir='z', offset=-2,
                     cmap=plt.cm.hot)
ax.set_zlim(-2,2)
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
−3
−2
−1
0
1
2
3