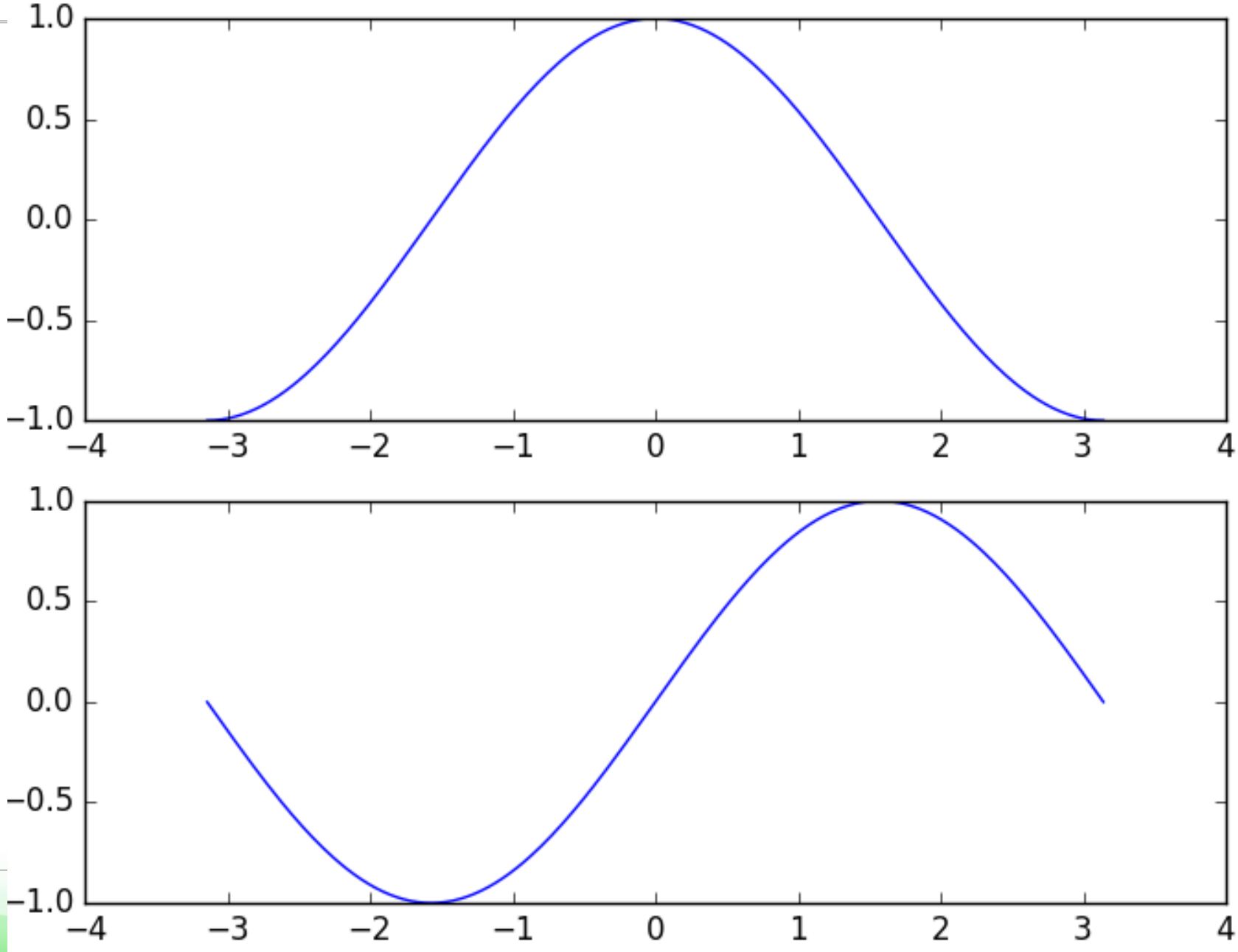
Subplots

```
# subplot1.py
import numpy as np
import matplotlib.pyplot as plt
X = np.linspace(-np.pi, np.pi, 256)
C = np.cos(X)
S = np.sin(X)

plt.subplot(2,1,1)    # (nrows, ncols, index)
plt.plot(X, C)
plt.subplot(2,1,2)
plt.plot(X, S)

plt.savefig("subplot1.png")
plt.show()
```

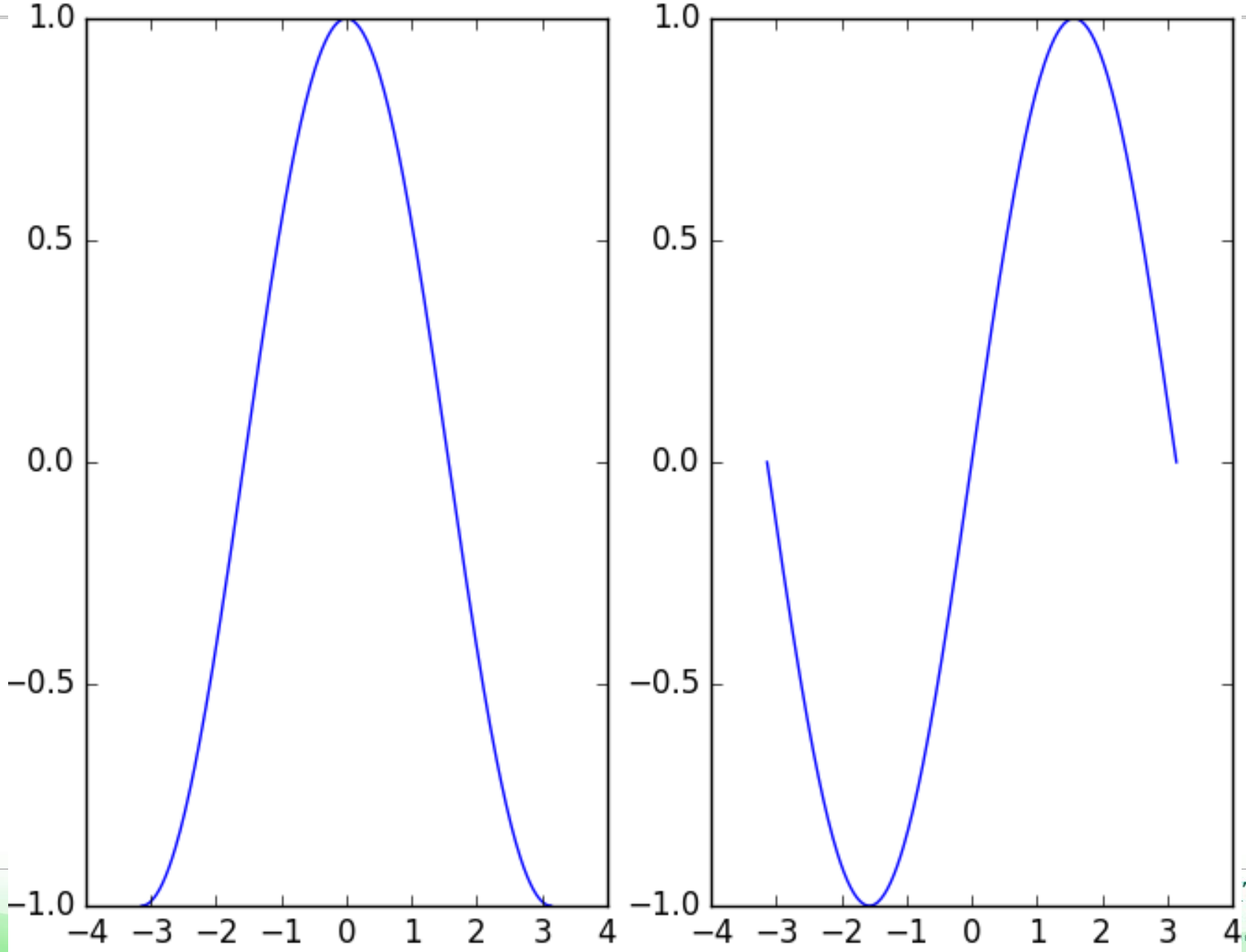
Subplots



Subplots

```
# subplot2.py
import numpy as np
import matplotlib.pyplot as plt
X = np.linspace(-np.pi, np.pi, 256)
C = np.cos(X)
S = np.sin(X)

plt.subplot(1,2,1)
plt.plot(X, C)
plt.subplot(1,2,2)
plt.plot(X, S)

plt.savefig("subplot2.png")
plt.show()
```

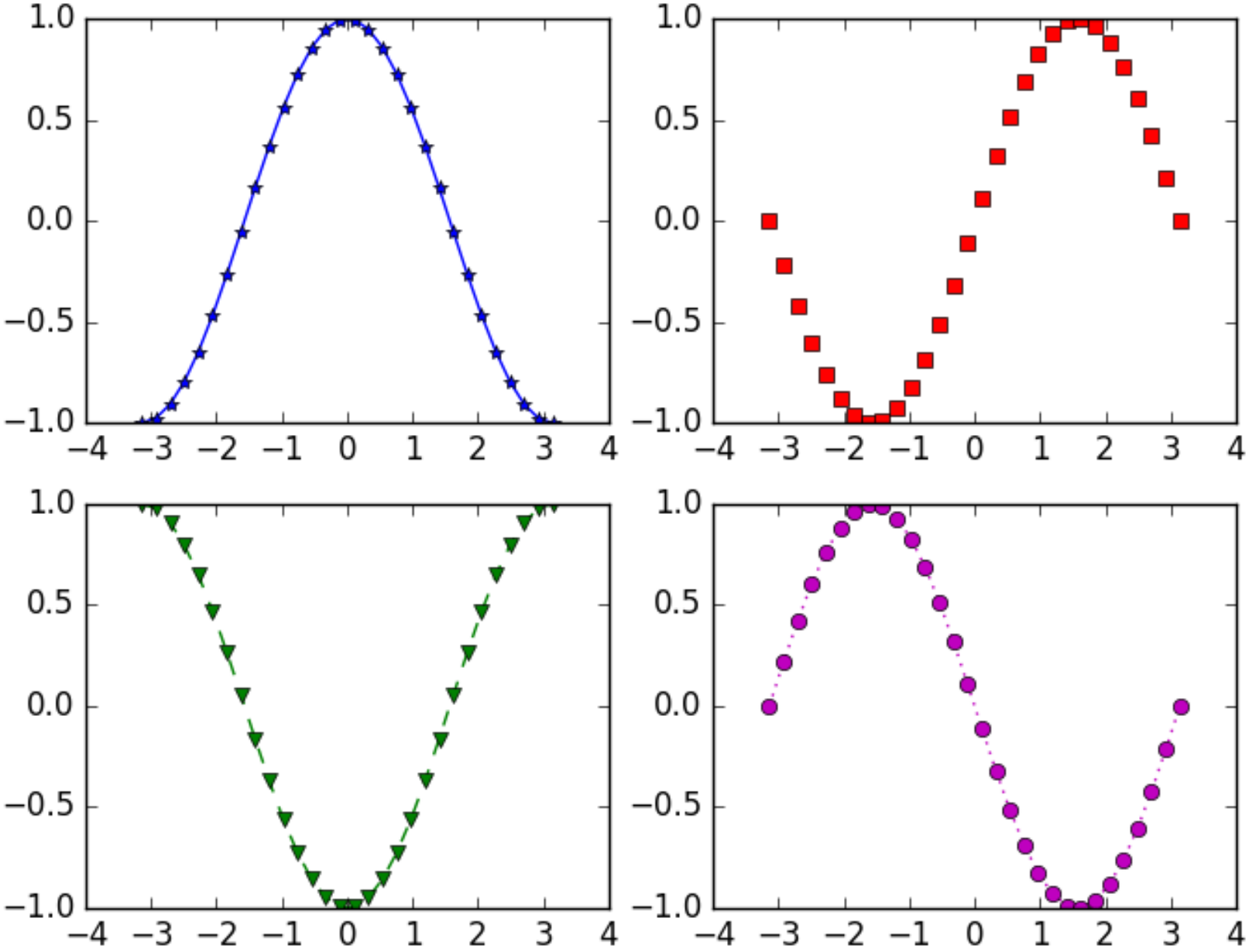
Subplots



Subplots

```
# subplot3.py
import numpy as np
import matplotlib.pyplot as plt
X = np.linspace(-np.pi, np.pi, 30)
C = np.cos(X)
S = np.sin(X)

plt.subplot(2,2,1)
plt.plot(X, C,'b-*')
plt.subplot(2,2,2)
plt.plot(X, S,'rs')
plt.subplot(2,2,3)
plt.plot(X, -C,'g--v')
plt.subplot(2,2,4)
plt.plot(X, -S, 'm:o')

plt.savefig("subplot3.png")
plt.show()
```

Subplots



Colors

'b'	blue
'g'	green
'r'	red
'c'	cyan
'm'	magenta
'y'	yellow
'k'	black
'w'	white

Line Styles

'-'	solid line
'--'	dashed line
'-.'	dash-dotted line
':'	dotted line
''	draw nothing

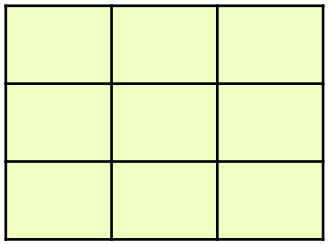
Markers

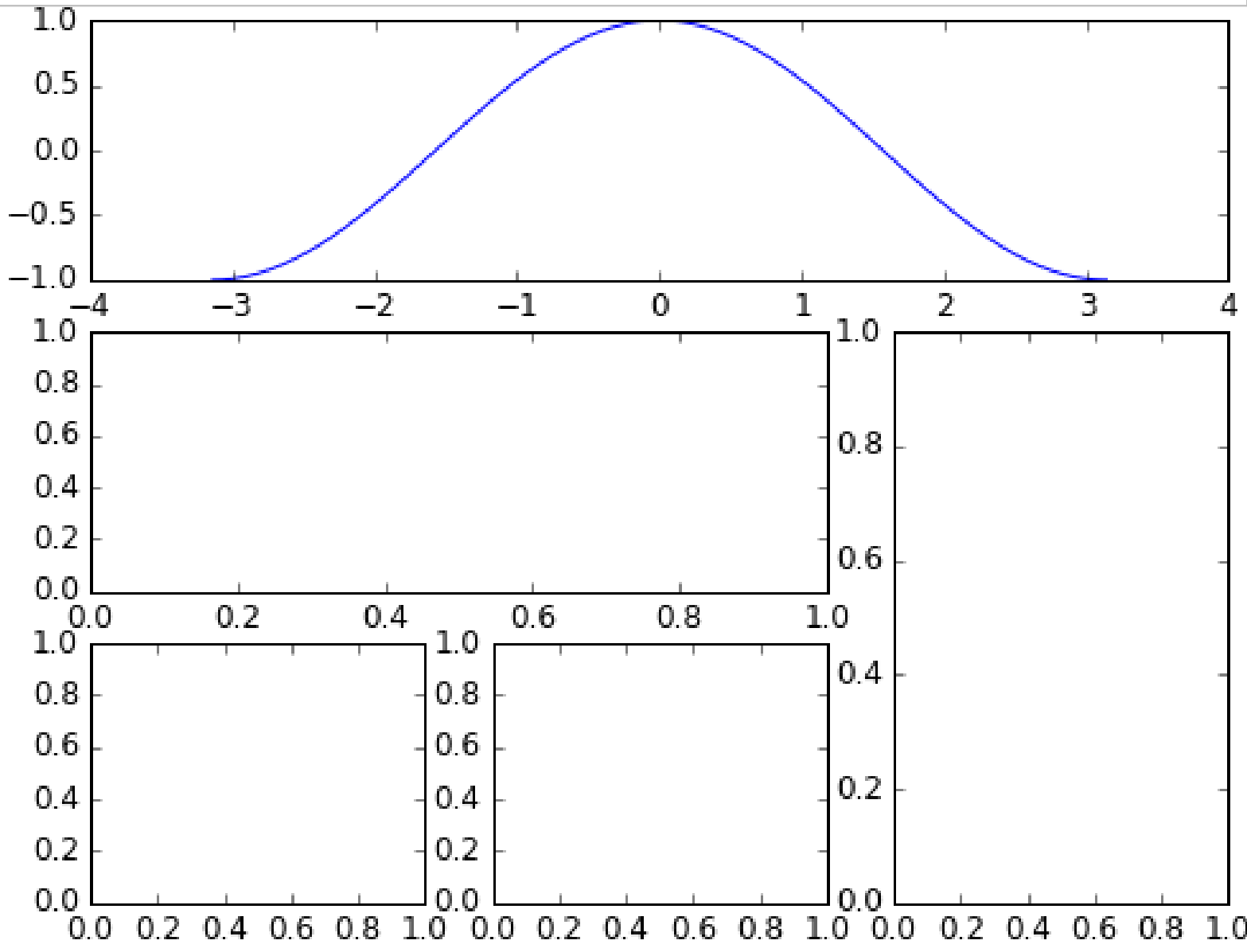
'.'	point	'*'	star
'o'	circle	'h'	hexagon1
'v'	triangle_down	'H'	hexagon2
'^'	triangle_up	'+'	plus
'<'	triangle_left	'x'	x
'>'	triangle_right	'D'	diamond
'8'	octagon	'd'	thin_diamond
's'	square	'|'	vline
'p'	pentagon	'_'	hline

GridSpec

```
# gridspec1.py
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.gridspec as gridspec

X = np.linspace(-np.pi, np.pi, 256)
C = np.cos(X)

gs = gridspec.GridSpec(3, 3)
ax1 = plt.subplot(gs[0, :])
plt.plot(X,C)
ax2 = plt.subplot(gs[1,:-1])
ax3 = plt.subplot(gs[1:, -1])
ax4 = plt.subplot(gs[-1,0])
ax5 = plt.subplot(gs[-1,-2])

plt.savefig("gridspec1.png", dpi=72)
plt.show()
```

GridSpec

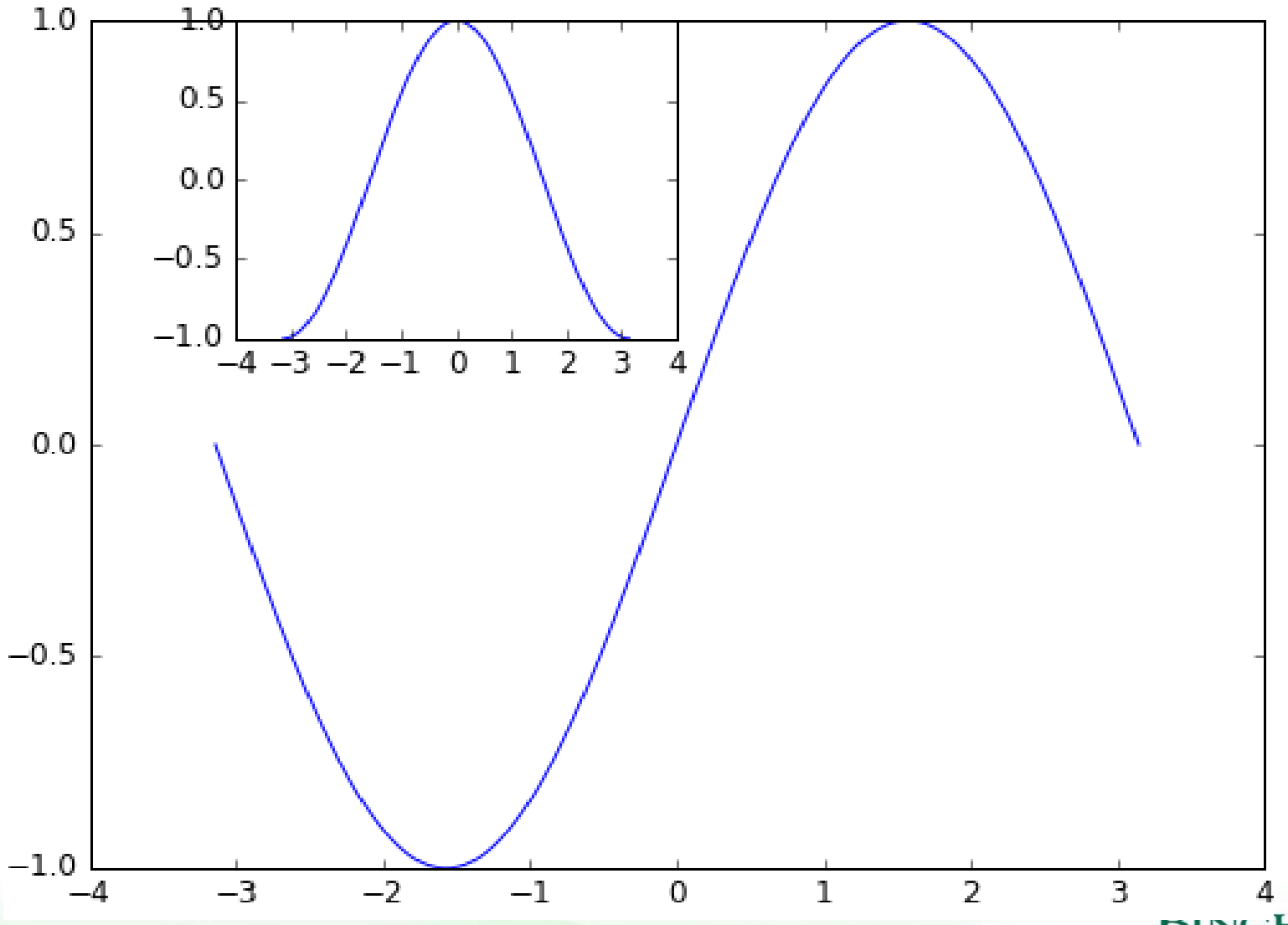




Axes

```
# axes1.py
import numpy as np
import matplotlib.pyplot as plt

X = np.linspace(-np.pi, np.pi, 256)
C = np.cos(X)
S = np.sin(X)
                    # [left, bottom, width, height]
ax1 = plt.axes([0.1, 0.1, 0.8, 0.8])
plt.plot(X,S)
ax2 = plt.axes([0.2, 0.6, 0.3, 0.3])
plt.plot(X,C)

plt.savefig("axes1.png", dpi=72)
plt.show()
```

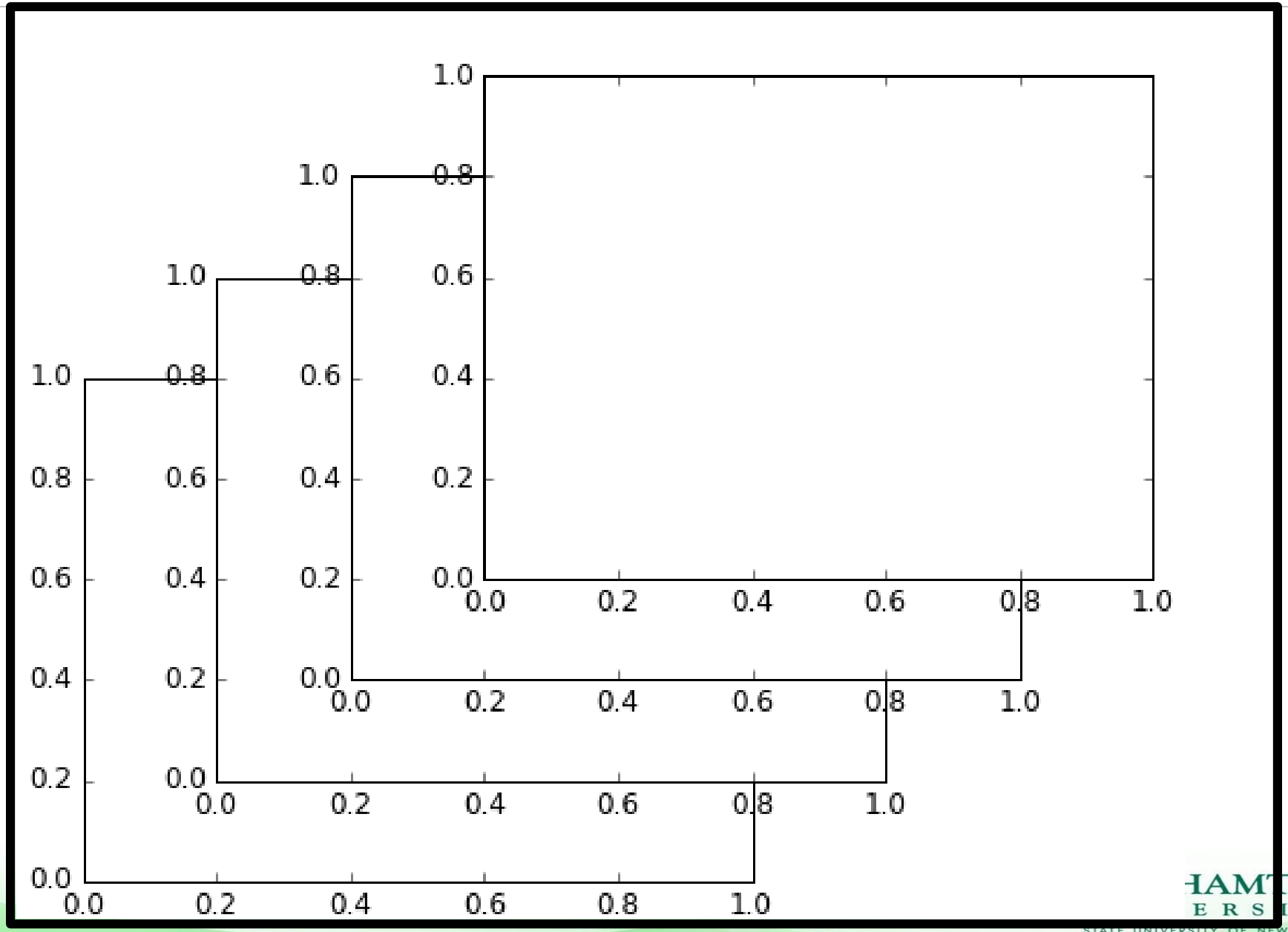
Axes



Axes

```
# axes2.py
import numpy as np
import matplotlib.pyplot as plt

ax1 = plt.axes([0.1, 0.1, 0.5, 0.5])
ax2 = plt.axes([0.2, 0.2, 0.5, 0.5])
ax3 = plt.axes([0.3, 0.3, 0.5, 0.5])
ax4 = plt.axes([0.4, 0.4, 0.5, 0.5])

plt.savefig("axes2.png", dpi=72)
plt.show()
```

Axes



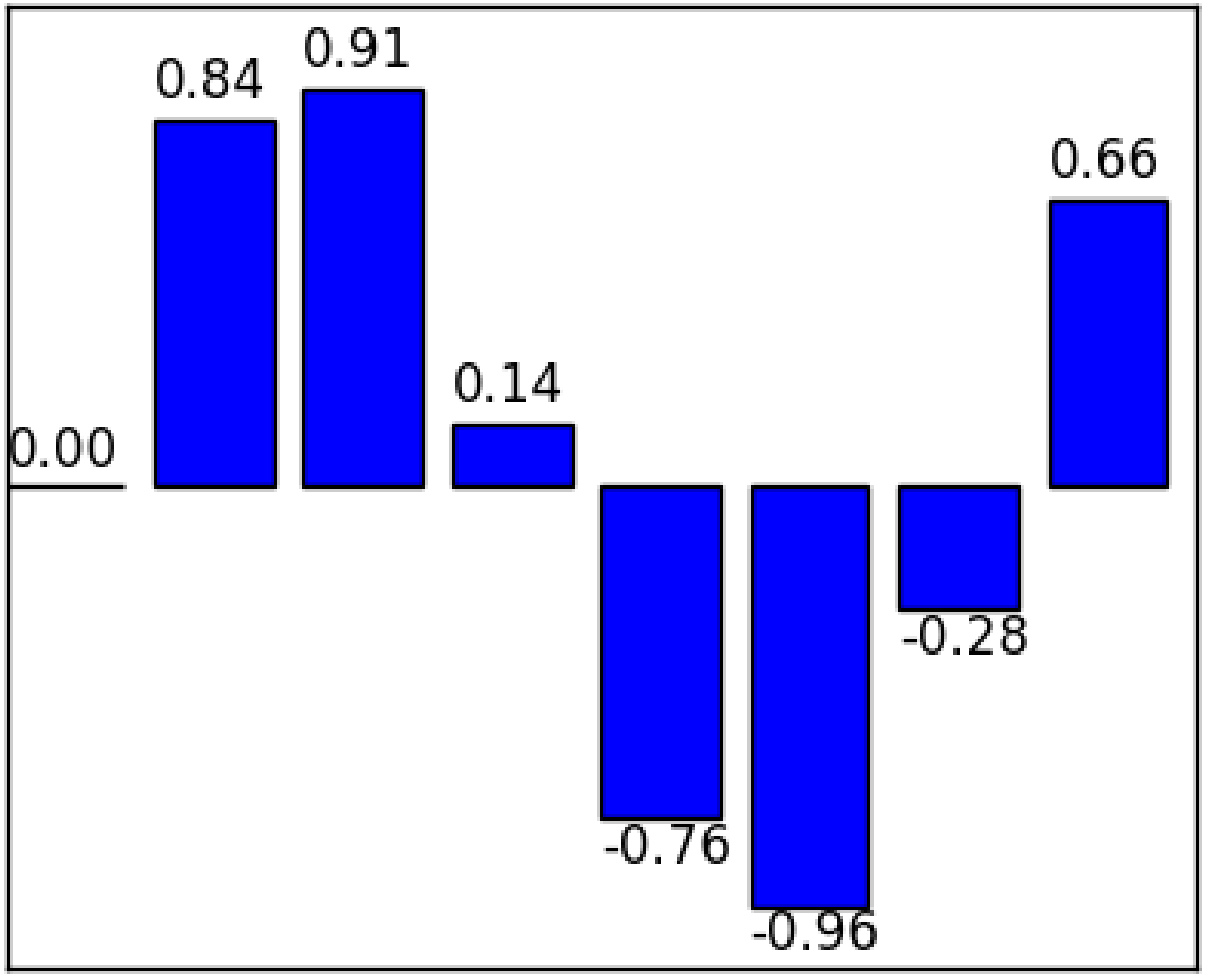
Matplotlib Gallery

- The matplotlib gallery is useful when you want to know how to create a figure
- Each example comes with its source code

Gallery Examples

```
# barv.py
import numpy as np
import matplotlib.pyplot as plt

X = np.arange(8)
Y = np.sin(X)
plt.figure(figsize=(5, 4))
plt.bar(X, Y)
for i in X:
    if Y[i] >= 0:
        plt.text(X[i], Y[i] + 0.05, '{0:0.2f}'.format(Y[i]))
    else:
        plt.text(X[i], Y[i] - 0.10, '{0:0.2f}'.format(Y[i]))

plt.ylim(-1.1, 1.1)
plt.xticks(())
plt.yticks(())
plt.show()
```

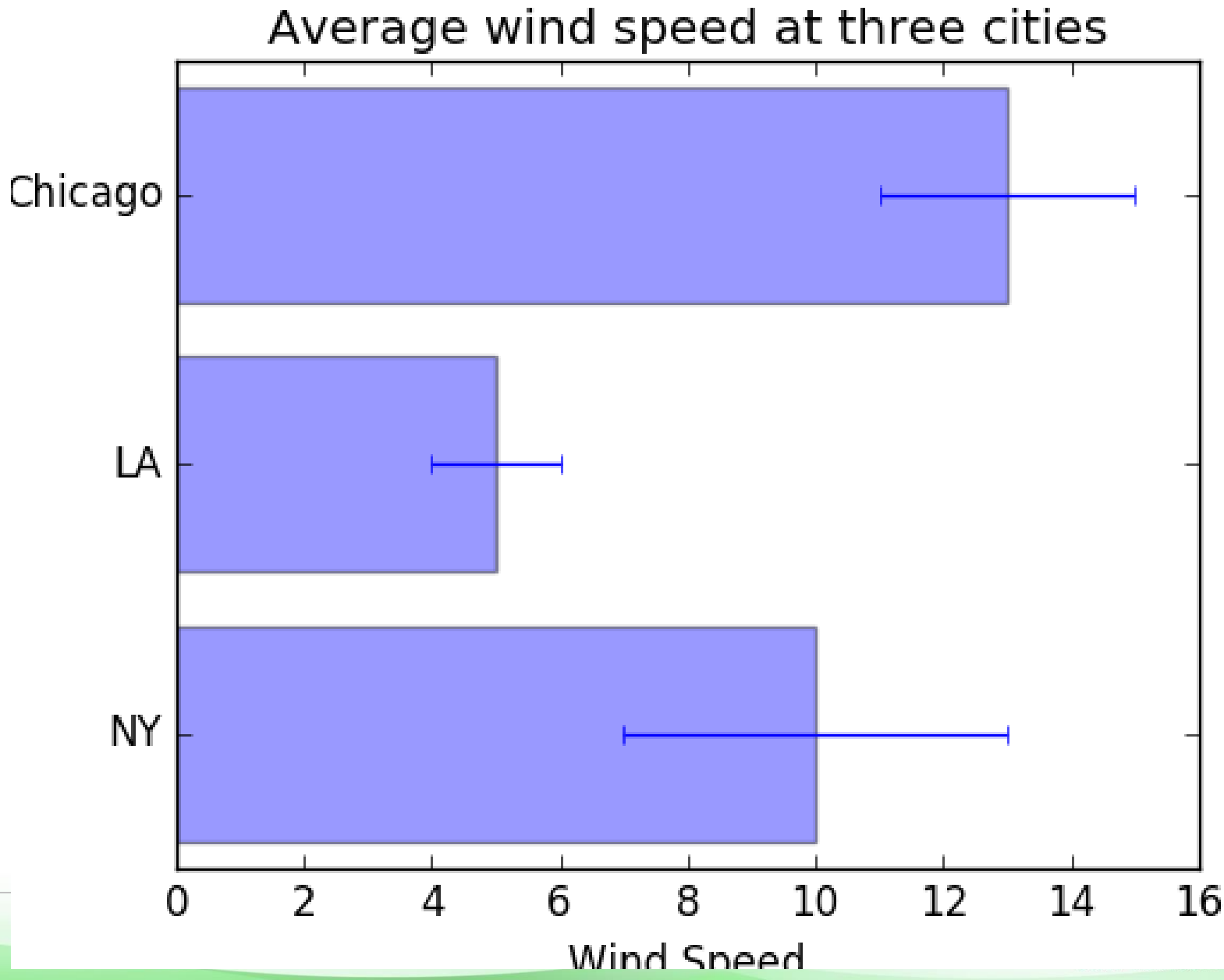
Gallery Examples



Gallery Examples

```
# barh.py
import numpy as np
import matplotlib.pyplot as plt

cities = ('NY', 'LA', 'Chicago')
y_pos = np.arange(len(cities))
wind = [10, 5, 13]
error = [3, 1, 2]
plt.barh(y_pos, wind, xerr=error, \
         align='center', alpha=0.4)
plt.yticks(y_pos, cities)
plt.xlabel('Wind Speed')
plt.title('Average wind speed at three cities')
plt.show()
```

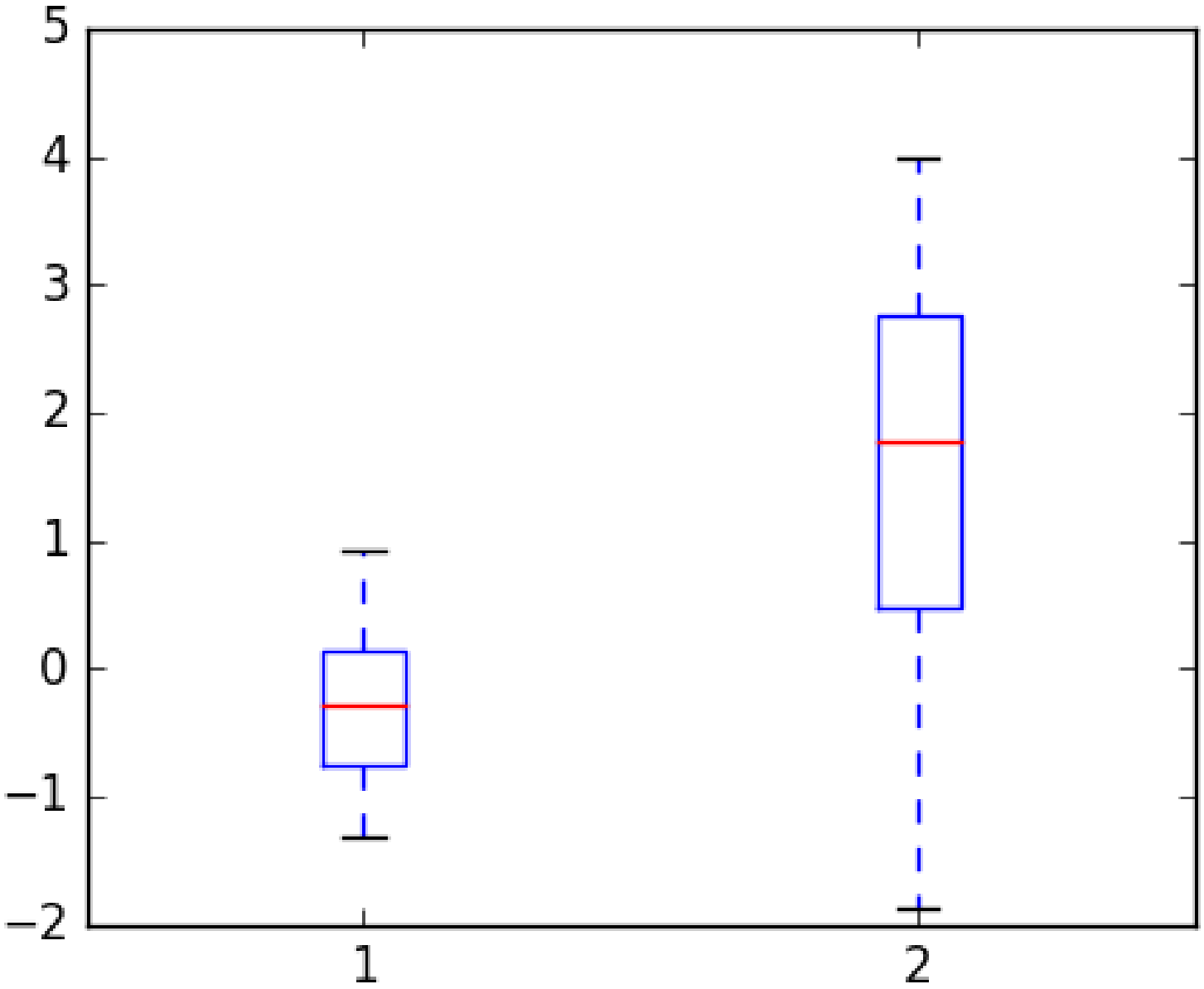
Gallery Examples



Gallery Examples

```
# boxplot.py
import matplotlib.pyplot as plt
import numpy as np

# Random test data
data1 = np.random.normal(0, 1, 20)
data2 = np.random.normal(2, 2, 20)
data = [data1, data2]

plt.figure(figsize=(5, 4))
plt.boxplot(data)

plt.show()
```

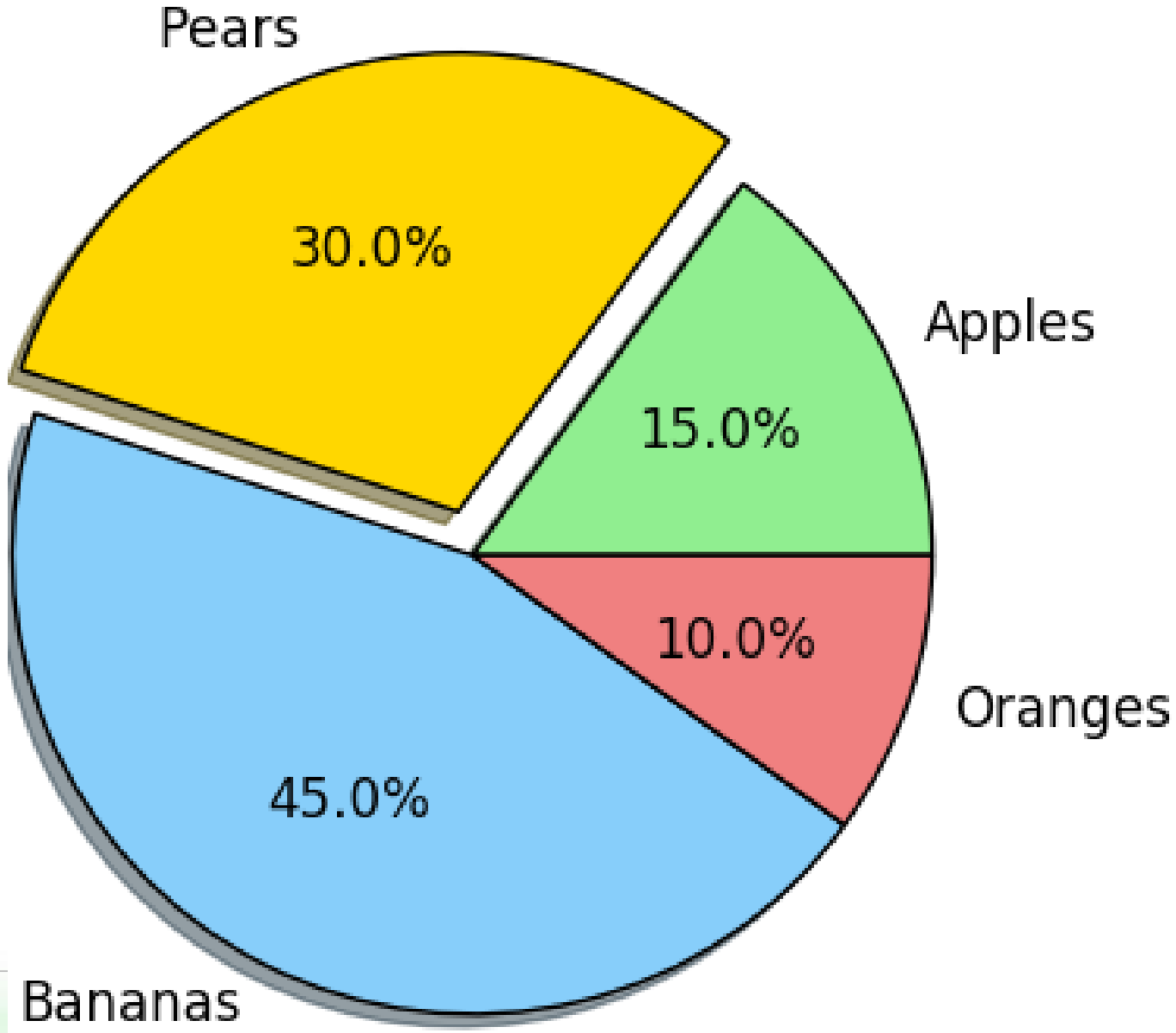
Gallery Examples



Gallery Examples

```
# piechart.py
import matplotlib.pyplot as plt

# The slices will be plotted counter-clockwise.
labels = ['Apples', 'Pears', 'Bananas', 'Oranges']
sizes = [15, 30, 45, 10]
colors = ['lightgreen', 'gold', 'lightskyblue', 'lightcoral']
explode = (0, 0.1, 0, 0)  # only "explode" the 2nd slice
plt.figure(figsize=(5, 4))
plt.pie(sizes, explode=explode,
        labels=labels,colors=colors,
        autopct='%0.1f%%',
        shadow=True, startangle=0)
# Set aspect ratio to be equal
# so that pie is drawn as a circle.
plt.axis('equal')
plt.show()
```

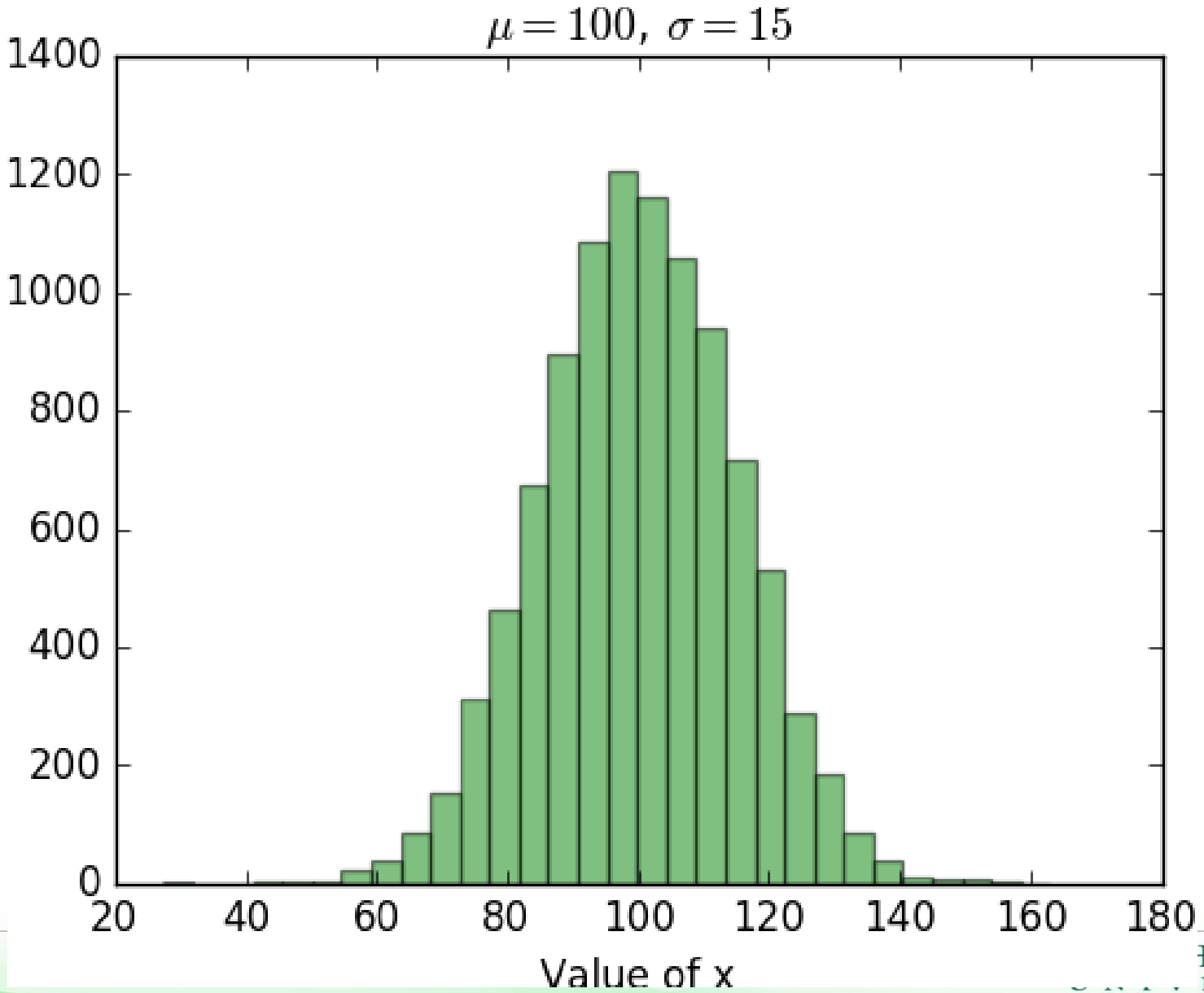
Gallery Examples



Gallery Examples

```
# histogram.py
import numpy as np
import matplotlib.pyplot as plt

mu = 100  # mean of distribution
sigma = 15  # std
x = mu + sigma * np.random.randn(10000)
num_bins = 30
plt.figure(figsize=(5, 4))
n, bins, patches = plt.hist(x, num_bins,
    facecolor='green', alpha=0.5)
plt.xlabel('Value of x')
plt.ylabel('Count')
plt.title(r'$\mu=100$, $\sigma=15$')
plt.show()
```

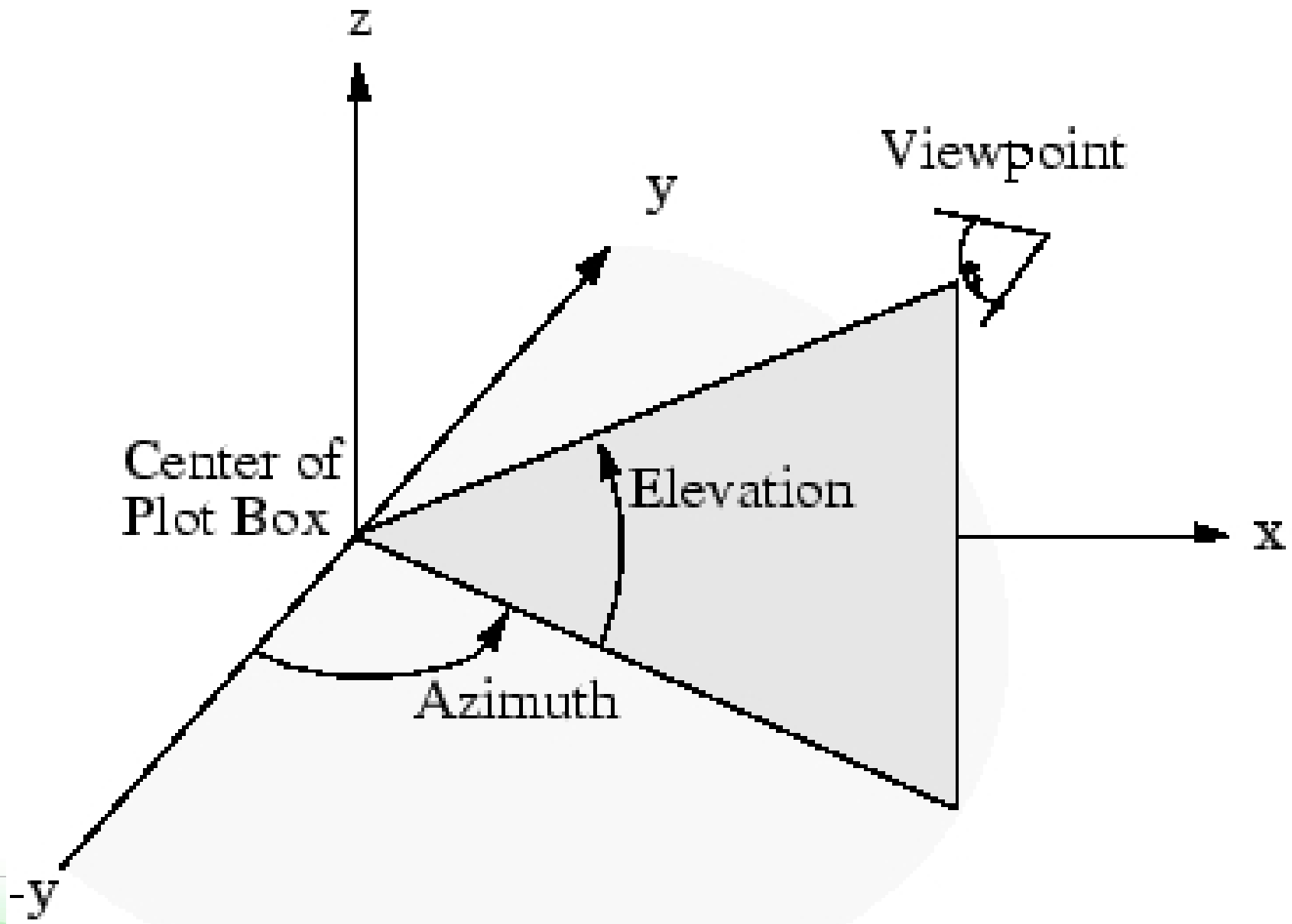
Gallery Examples



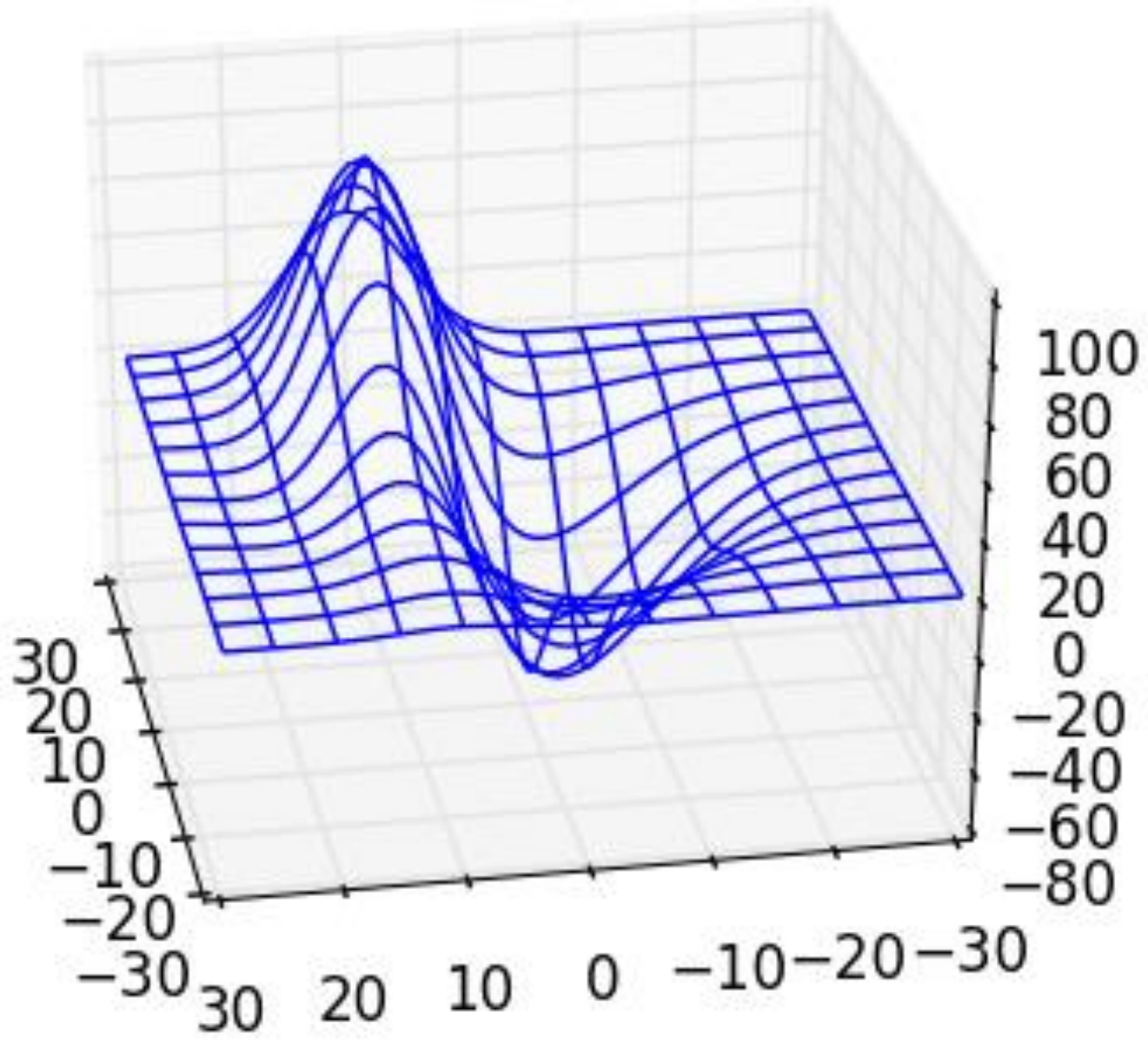
Gallery Examples

```
# wireframe3d.py
from mpl_toolkits.mplot3d import axes3d
import matplotlib.pyplot as plt
import numpy as np

fig = plt.figure(figsize=(5, 4))
ax = fig.add_subplot(111, projection='3d')
X, Y, Z = axes3d.get_test_data(0.1)
# rstride and cstride controls
# the row and col step sizes
ax.plot_wireframe(X, Y, Z, rstride=5, cstride=5)

ax.view_init(azim=170, elev=30)
plt.show()
```

Gallery Examples



Gallery Examples



Numpy Meshgrid

```
>>> nx, ny = 3, 2
>>> x = np.linspace(0, 1, nx)
>>> y = np.linspace(0, 1, ny)
>>> x
array([ 0. ,  0.5,  1. ])
>>> y
array([ 0.,  1.])

>>> xv, yv = meshgrid(x, y, sparse=True)
>>> xv
array([[ 0. ,  0.5,  1. ]])
>>> yv
array([[ 0.],
       [ 1.]])
```

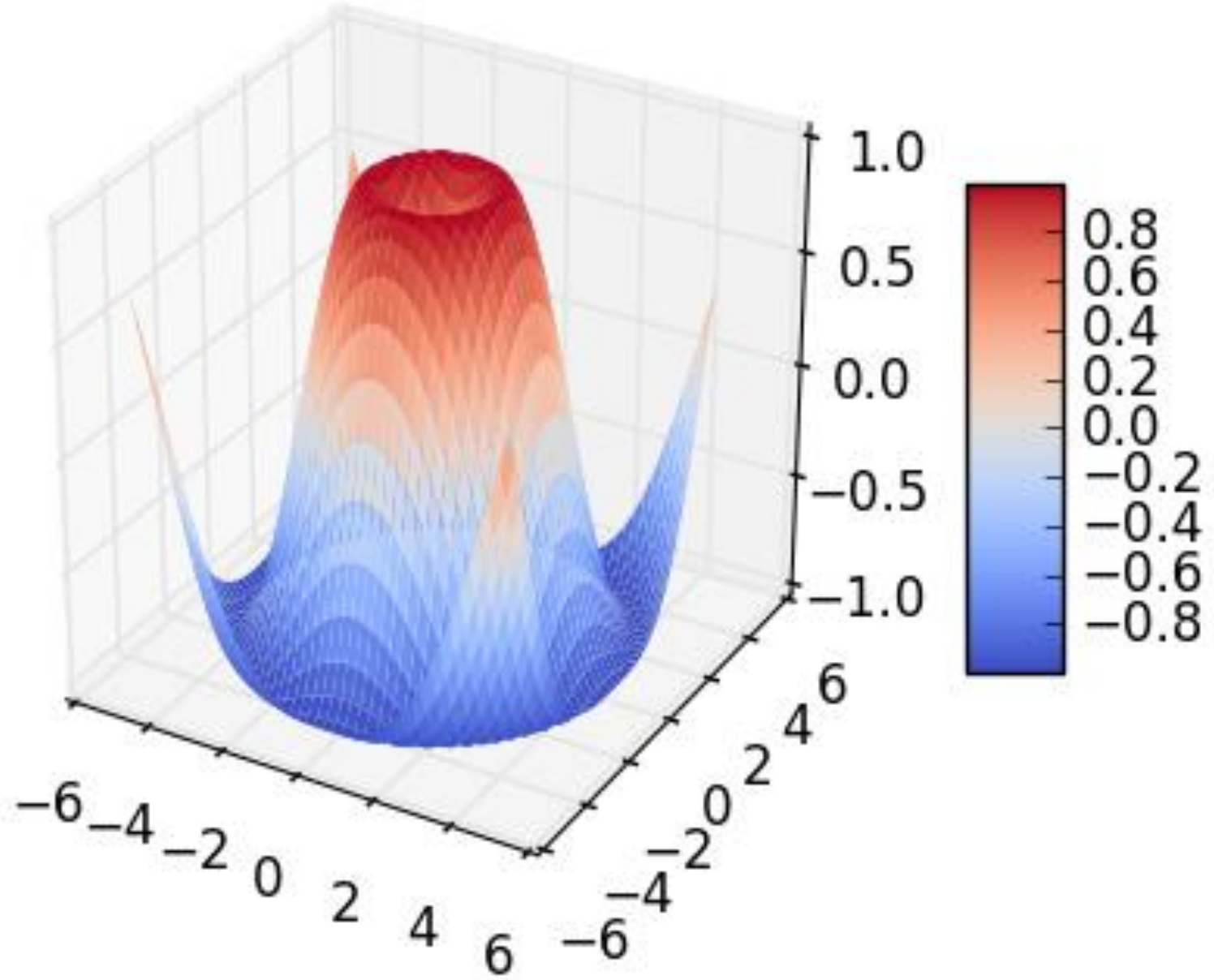
Gallery Examples

```
>>> xv, yv = np.meshgrid(x, y)
>>> xv
array([[ 0. ,  0.5,  1. ],
       [ 0. ,  0.5,  1. ]])
>>> yv
array([[ 0.,  0.,  0.],
       [ 1.,  1.,  1.]])
```

Gallery Examples

```
# surface3d.py
from mpl_toolkits.mplot3d import axes3d
import matplotlib.pyplot as plt
import numpy as np

fig = plt.figure(figsize=(5, 4))
ax = fig.gca(projection='3d')
X = np.arange(-5, 5, 0.25)
Y = np.arange(-5, 5, 0.25)
X, Y = np.meshgrid(X, Y)
Z = np.sin(np.sqrt(X**2 + Y**2))
surf = ax.plot_surface(X, Y, Z,
    rstride=1, cstride=1,
    cmap=plt.cm.coolwarm, linewidth=0)
ax.set_zlim(-1.01, 1.01)
fig.colorbar(surf, shrink=0.5, aspect=5)
plt.show()
```

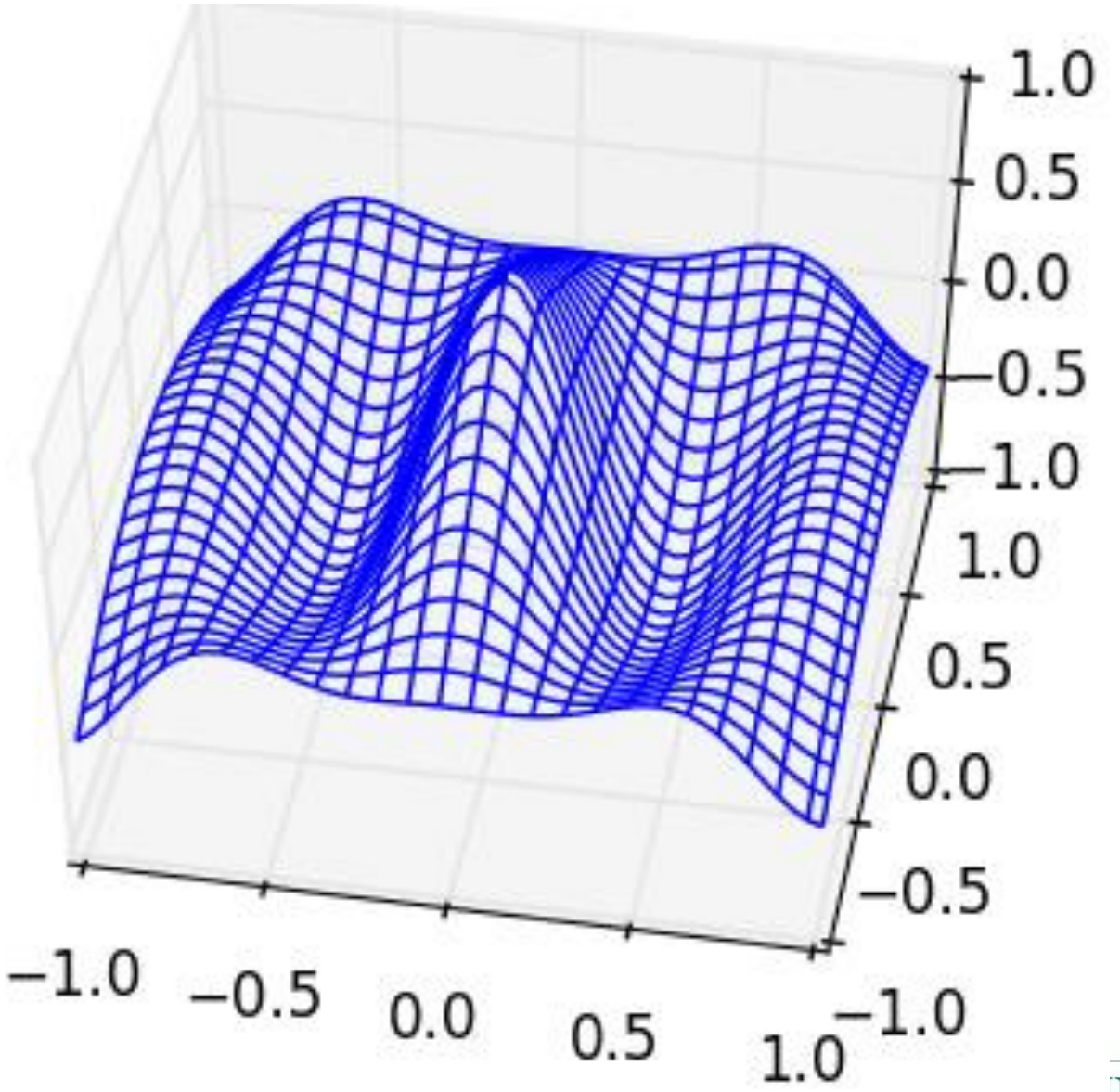
Gallery Examples



Gallery Examples

```
# animation3d.py
from mpl_toolkits.mplot3d import axes3d
import matplotlib.pyplot as plt
import numpy as np
fig = plt.figure(figsize=(10, 8))
ax = fig.add_subplot(111,projection='3d')
ax.set_zlim(-1, 1)
xs =ys= np.linspace(-1, 1, 30)
X, Y = np.meshgrid(xs, ys)
wframe = None
for phi in np.arange(360):  #360 degrees
    if wframe != None: # Remove old lines before drawing
        ax.collections.remove(wframe)
    Z = np.cos(2 * np.pi * X + phi) * \
        (1 - np.sqrt(X**2 + Y**2))
    wframe = ax.plot_wireframe(X, Y, Z)
    ax.view_init(azim=phi, elev=50)
    plt.pause(0.1)
```

Gallery Examples



Electric Load in California

Wang, Y., Li, L. (2016). Critical peak electricity pricing for sustainable manufacturing: Modeling and case studies. Applied Energy, 175, 40-53.

```
"""Plot the CAISO hourly load data in 3D."""
import csv
import numpy as np
import matplotlib as mpl
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import axes3d
```

Electric Load in California

```
# Read the load data.
year = 2012
mat = list()
f = open('caiso_load_{}.csv'.format(year))
rows = csv.reader(f)
for row in rows:
    # print(row)
    mat.append(row)
f.close()
mat=np.array(mat)
xnum, ynum = np.shape(mat)
print('xnum =', xnum)
input("Press Enter to continue...")
print('ynum =', ynum)
input("Press Enter to continue...")
```

Electric Load in California

```
X = np.arange(1, xnum)
Y = np.arange(1, ynum)
Y, X = np.meshgrid(Y, X)
Z = mat[1:, 1:]
Z = Z.astype(np.float)
```

Electric Load in California

```
# Plot the data.
fig = plt.figure(figsize=(11,3))
ax = fig.add_axes([-0.12, 0.0, 1.22, 1.10],
                  projection='3d')

cmap = mpl.colors.ListedColormap(["#efe8f2",
                                  "#c2a5cf",
                                  "#7b3294"])
#cmap = mpl.colors.ListedColormap(["g","b","r"])
bounds = np.max(Z) * np.array([0, 0.6, 0.9, 1])
norm = mpl.colors.BoundaryNorm(bounds, cmap.N)
surf = ax.plot_surface(X, Y, Z, rstride=1,
                       cstride=1, cmap=cmap,
                       norm=norm, linewidth=0.1,
                       antialiased=True, alpha=0.9)
```

Electric Load in California

```
ax.set_zlabel('Load (MW)')
ax.set_xlim(0, xnum-1)
ax.set_ylim(0, ynum-1)
ax.set_zlim(0.1, np.max(Z)) #to avoid overlapping
ax.view_init(azim=-80, elev=30)
```

Electric Load in California

```
# Fine tune the x and y tick labels.
# Find the start of each month.
loc = np.zeros(12)
labels = np.array(['']*12, dtype='U10')
date = mat[1:, 0]
j = 0
for i in range(0, len(date)):
    if '/1/'+str(year) in date[i]:
        loc[j] = i
        labels[j] = date[i]
        j = j + 1
ax.set_xticks(loc)
ax.set_xticklabels(labels, rotation=15,
                   verticalalignment='center')
```

Electric Load in California

```
loc = np.arange(0, 25, 4)
labels = loc.astype('str')
ax.set_yticks(loc)
ax.set_yticklabels(labels, rotation=0,
                   verticalalignment='center')
```

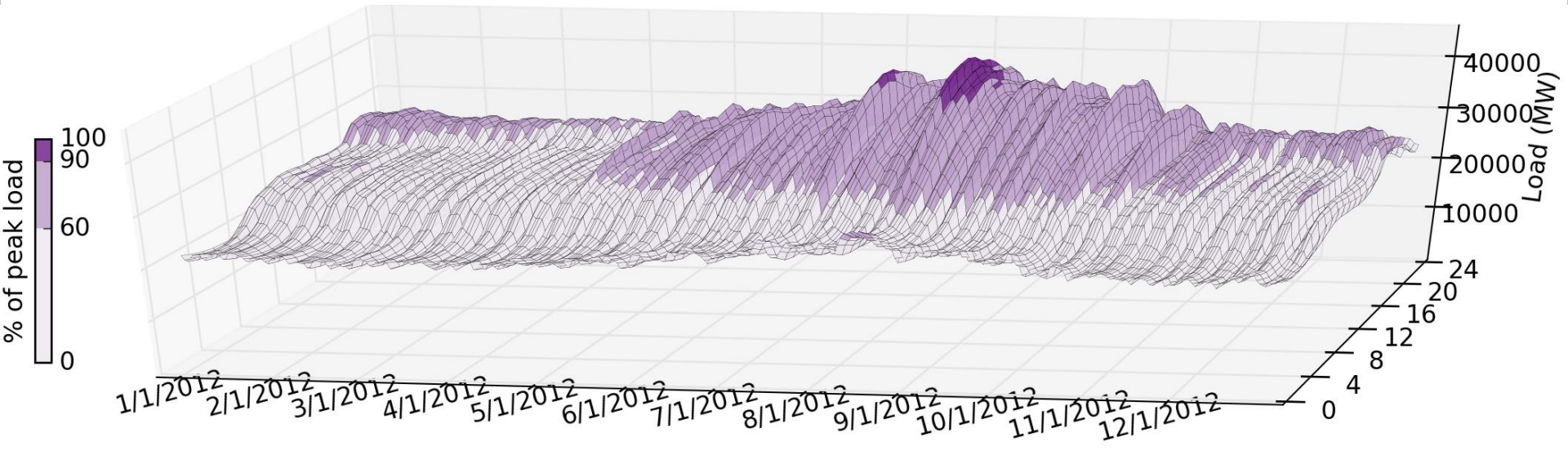
Electric Load in California

```
# Add the colorbar.
cax = fig.add_axes([0.03, 0.2, 0.01, 0.5])
cbar = plt.colorbar(surf, cax=cax, ax=ax,
                    cmap=cmap, norm=norm,
                    ticks=bounds,
                    spacing='proportional',
                    orientation='vertical')
cbar.ax.set_yticklabels(['0', '60', '90', '100'])
cbar.set_label('% of peak load', labelpad=-50)
```

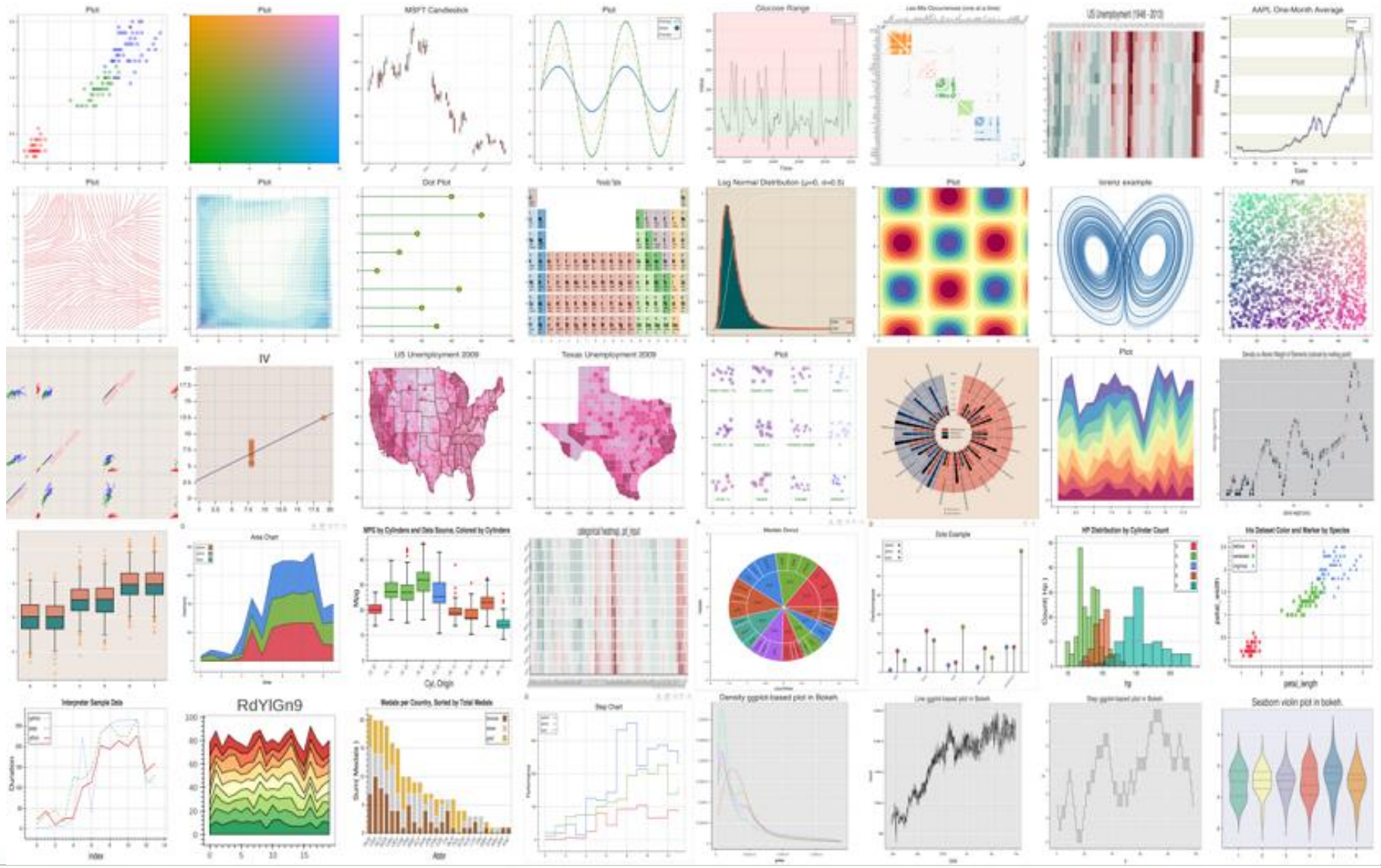
Electric Load in California

```
# Calculate hours with load > 90% of peak load.
hours = Z[Z>bounds[2]].size
percent = hours / Z.size * 100
print('Hours with load >90%: {0}, which is {1:0.2f}%.'
      .format(hours, percent))

# Show and save the figure.
#plt.show()
fig.savefig('caiso_load.png', dpi=200)
```

Electric Load in California



Other Packages - Bokeh



Other Packages - Seaborn