第 15 章　数据可视化

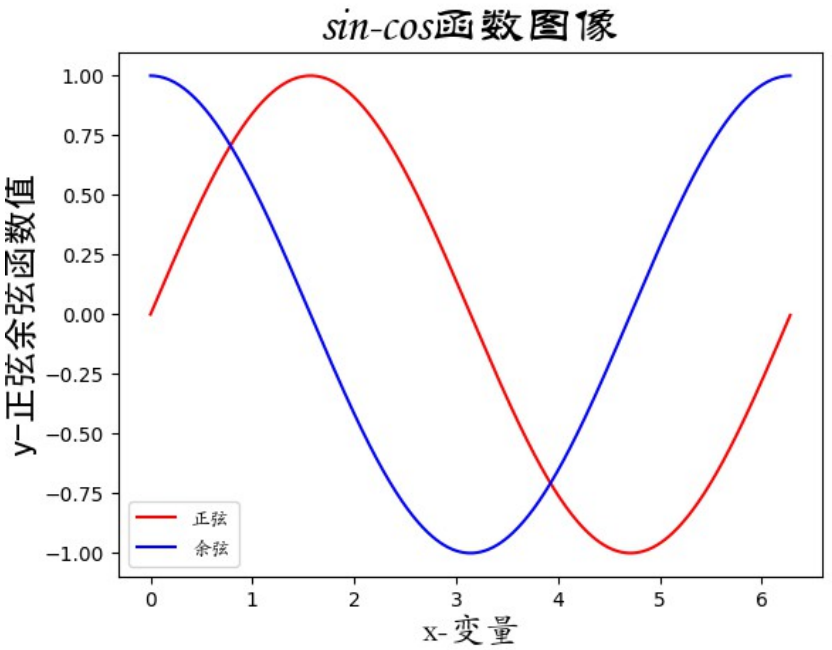
15.1 matplotlib 简介

- Python 扩展库 matplotlib 包括 pylab 、 pyplot 等绘图模块以及大量用于字体、颜色、图例等图形元素的管理与控制的模块。其中 pylab 和 pyplot 模块提供了类似于 MATLAB 的绘图接口，支持线条样式、字体属性、轴属性以及其他属性的管理和控制，可以使用非常简洁的代码绘制出优美的各种图案。
- 使用 pylab 或 pyplot 绘图时一般过程为：首先读入数据，然后根据实际需要绘制折线图、散点图、柱状图、饼状图、雷达图或三维曲线和曲面，接下来设置轴和图形属性，最后显示或保存绘图结果。

15.2 绘制带有中文标题、标签和图例的折线图

· **例 15-1** 绘制带有中文标题、标签和图例的正弦和余弦图像。

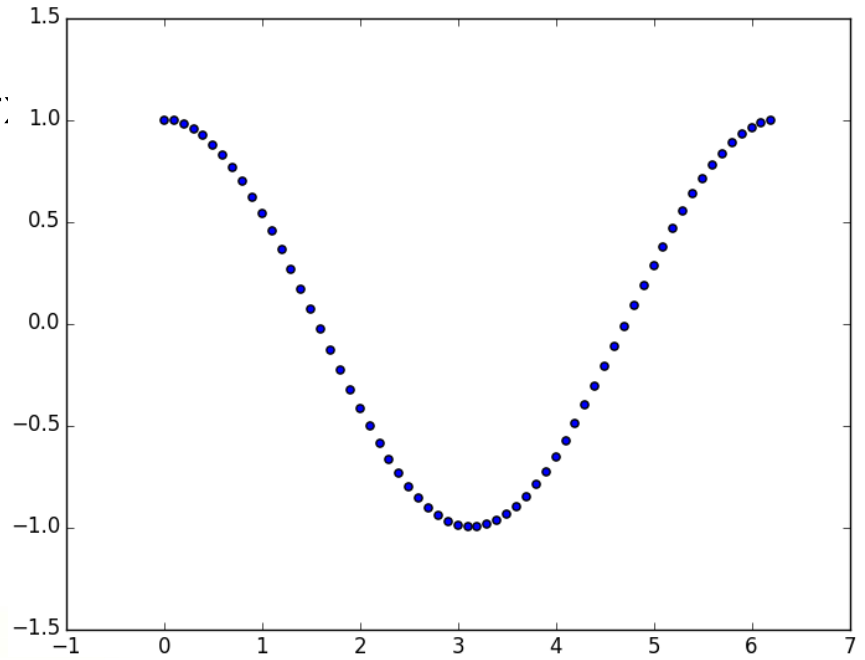
```
import numpy as np
import matplotlib.pylab as pl
import matplotlib.font_manager as fm


t = np.arange(0.0, 2.0*np.pi, 0.01)     # 自变量取值范围
s = np.sin(t)                           # 计算正弦函数值
z = np.cos(t)                           # 计算余弦函数值
pl.plot(t,                              #x 轴坐标
        s,                              #y 轴坐标
        label='正弦',                    # 标签
        color='red')                    # 颜色
pl.plot(t, z, label='余弦', color='blue')
pl.xlabel('x-变量',                      # 标签文本
```



15.3 绘制散点图

· **例 15-2** 绘制余弦曲线散点图。

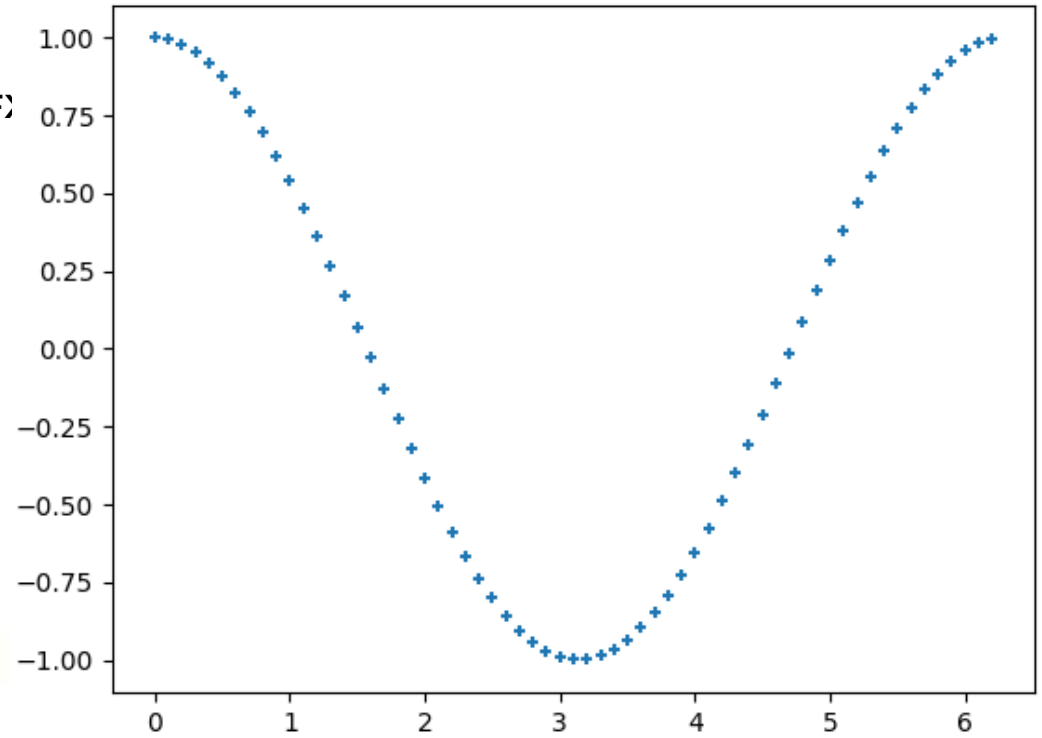
```
import numpy as np
import matplotlib.pylab as pl

x = np.arange(0, 2.0*np.pi, 0.1)   #
y = np.cos(x)      #y 轴数据
pl.scatter(x, y) # 绘制散点图
pl.show()          # 显示绘制的结果图像
```



15.3 绘制散点图

· **例 15-3** 设置散点图的线宽、散点符号及大小。

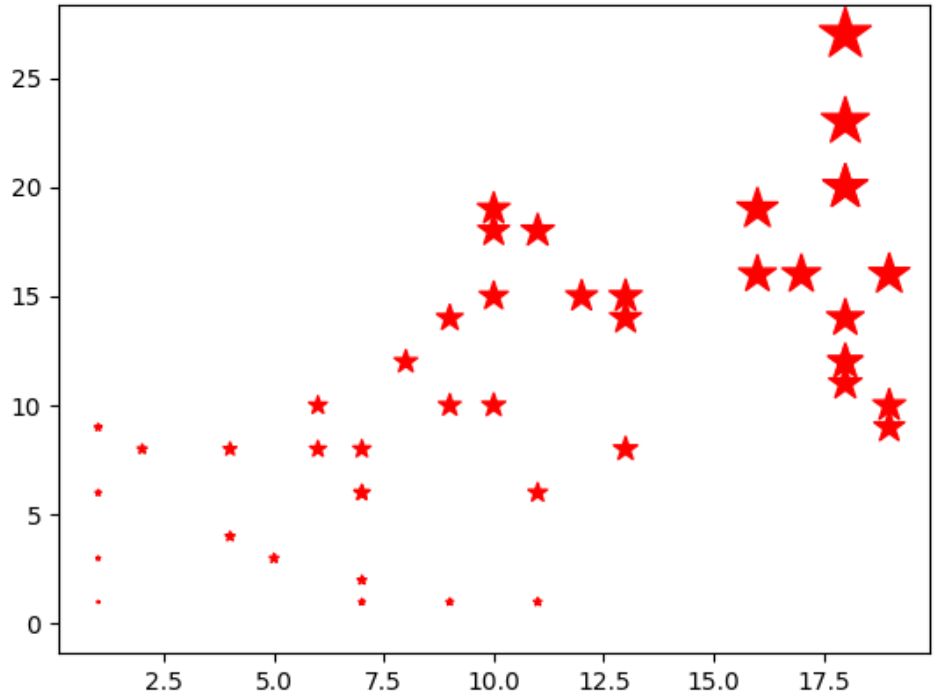
```
import numpy as np
import matplotlib.pylab as pl

x = np.arange(0, 2.0*np.pi, 0.1)   #x
y = np.cos(x)                      #y 轴数据
pl.scatter(x,                      #x 轴坐标
           y,                      #y 轴坐标
           s=40,                   # 散点大小
           linewidths=6,           # 线宽
           marker='+')             # 散点符号
pl.show()
```



15.3 绘制散点图

· **例 15-4** 绘制大小与位置有关的红

```
import matplotlib.pylab as pl
from numpy.random import randint


x = randint(1, 20, 50)          # 模拟 x 轴数
y = x + randint(-10, 10, 50)    # 生成 y 轴数
pl.scatter(x,
           y,
           s=x*y,       # 散点大小与位置有关
           c='r',       # 设置散点颜色
           marker='*') # 设置散点形状，五角星
pl.show()
```



15.4 绘制饼状图

- **例 15-5** 饼状图绘制与属性设置。

```
import numpy as np
import matplotlib.pyplot as plt


labels = ('Frogs', 'Hogs', 'Dogs', 'Logs')
colors = ('#FF0000', 'yellowgreen', 'gold', 'blue')
explode = (0, 0.02, 0, 0.08)          # 使所有饼状图中第 2 片和第 4 片裂开


fig = plt.figure(num=1,                 #num 为数字表示图像编号，
                                        # 如果是字符串则表示图形窗口标题
                 figsize=(10,8),        # 图形大小，格式为（宽度，高度），
                                        # 单位为英寸
```

15.4 绘制饼状图

```
ax.pie(np.random.random(4),          #4 个介于 0 和 1 之间的随机数据
       explode=explode,              # 设置每个扇形的裂出情况
       labels=labels,                # 设置每个扇形的标签
       colors=colors,                # 设置每个扇形的颜色
       pctdistance=0.8,              # 设置扇形内百分比文本与中心的距离
       autopct='%1.1f%%',            # 设置每个扇形上百分比文本的格式
       shadow=True,                  # 使用阴影，呈现一定的立体感
       startangle=90,                # 设置第一块扇形的起始角度
       radius=0.25,                  # 设置饼的半径
       center=(0, 0),                # 设置饼在图形窗口中的坐标
       counterclock=False,           # 顺时针绘制，默认是逆时针
       frame=True)                   # 是示图形边框
```

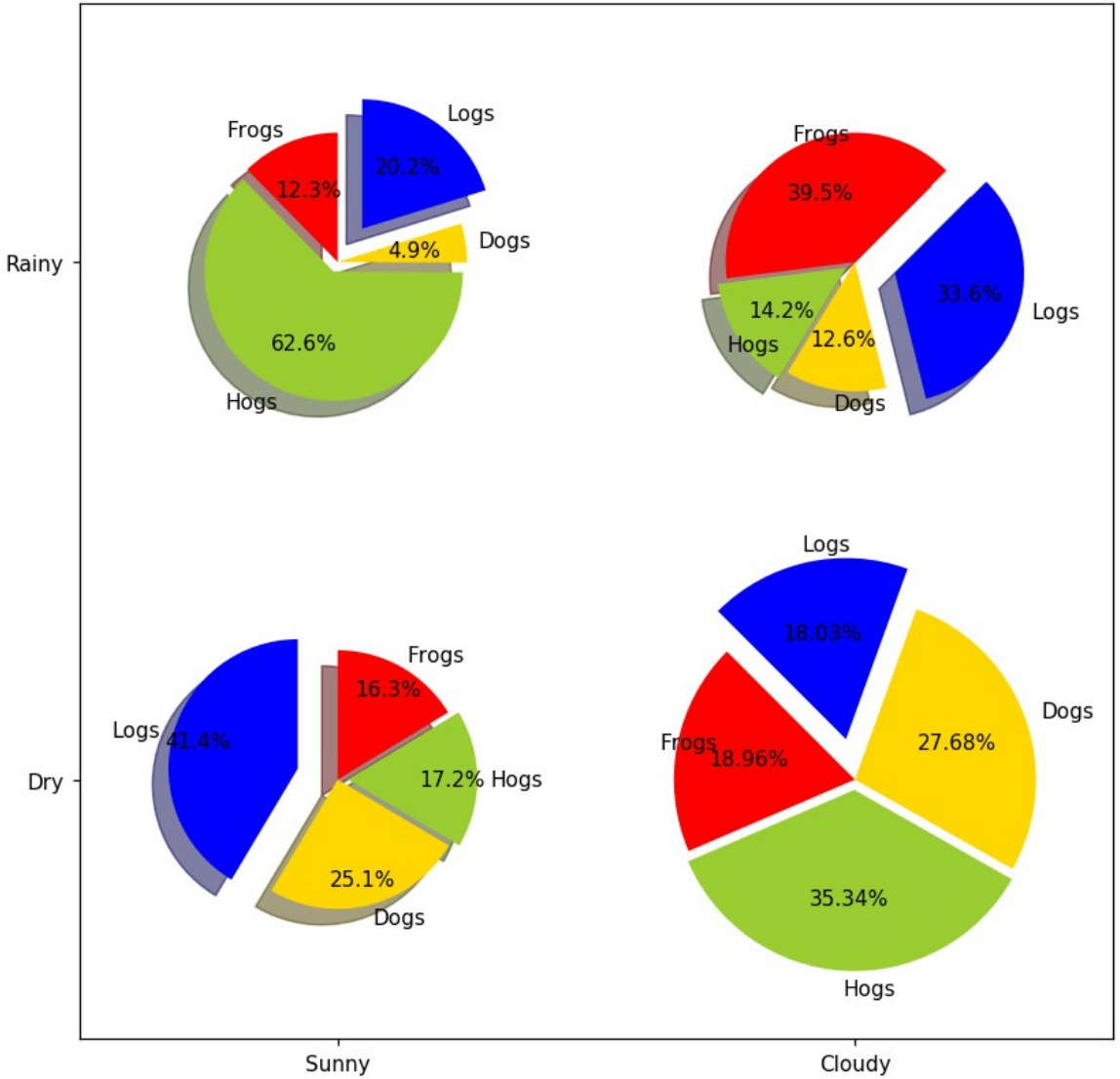
15.4 绘制饼状图

```
ax.pie(np.random.random(4), explode=explode, labels=labels,
       colors=colors, autopct='%1.1f%%', shadow=True,
       startangle=45, radius=0.25, center=(1, 1), frame=True)
ax.pie(np.random.random(4), explode=explode, labels=labels,
       colors=colors, autopct='%1.1f%%', shadow=True,
       startangle=90, radius=0.25, center=(0, 1), frame=True)
ax.pie(np.random.random(4), explode=explode, labels=labels,
       colors=colors, autopct='%1.2f%%', shadow=False,
       startangle=135, radius=0.35, center=(1, 0), frame=True)
```

15.4 绘制饼状图

```
ax.set_xticks([0, 1])                                    # 设置 x 坐标轴刻度
ax.set_yticks([0, 1])                                    # 设置 y 轴坐标轴刻度

ax.set_xticklabels(["Sunny", "Cloudy"])# 设置坐标轴刻度上的标签
ax.set_yticklabels(["Dry", "Rainy"])

ax.set_xlim((-0.5, 1.5))                                 # 设置坐标轴跨度
ax.set_ylim((-0.5, 1.5))

ax.set_aspect('equal')                                   # 设置纵横比相等
```

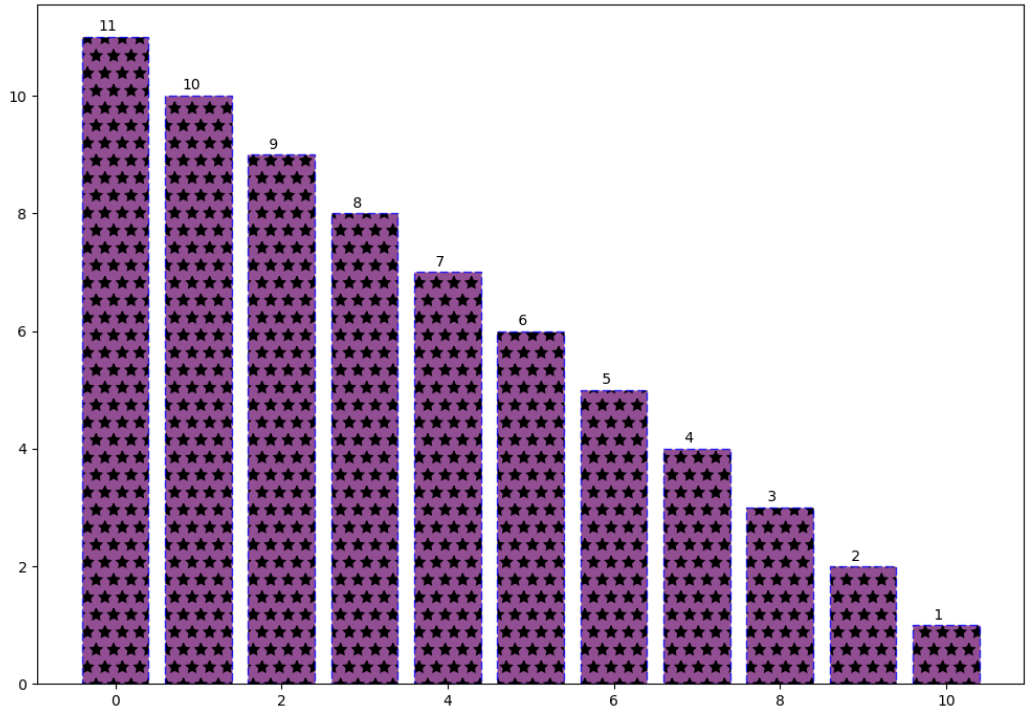
15.4 绘制饼状图


Rainy
Dry
Sunny
Cloudy
Frogs
Logs
Dogs
Hogs
20.2%
12.3%
4.9%
62.6%
39.5%
33.6%
14.2%
12.6%
16.3%
41.4%
17.2%
25.1%
18.03%
18.96%
27.68%
35.34%

15.5 绘制柱状图

· **例 15-6** 绘制柱状图并设置图形属性和文本标注。

```
import numpy as np
import matplotlib.pyplot as plt


# 生成测试数据
x = np.linspace(0, 10, 11)
y = 11-x


# 绘制柱状图
plt.bar(x,
        y,
        color='#772277',          # 柱的颜色
        alpha=0.8,                # 透明度
```



工于建构 成于创造

15.6 绘制雷达图

· **例 15-7** 绘制雷达图。

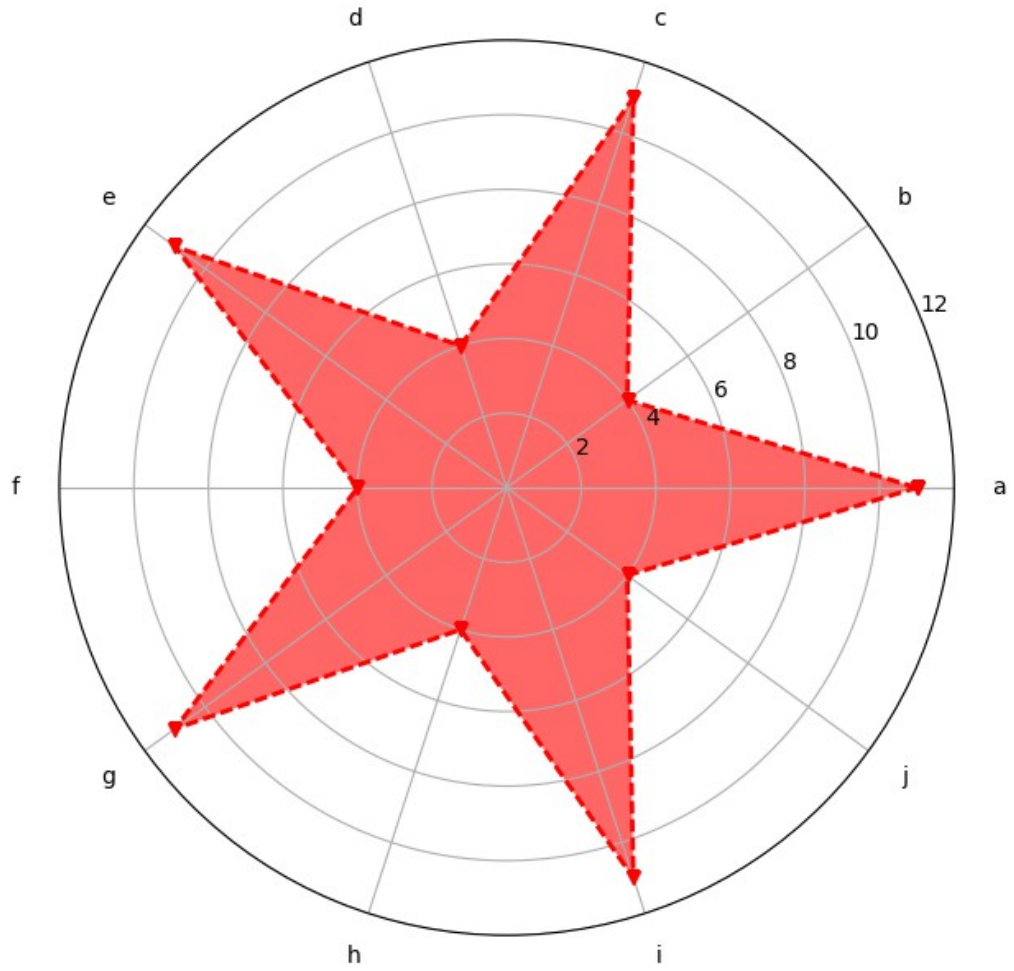
```
import numpy as np
import matplotlib.pyplot as plt


labels = np.array(list('abcdefghij')) # 设置标签
data = np.array([11,4]*5)             # 创建模拟数据
dataLength = len(labels)              # 数据长度


#angles 数组把圆周等分为 dataLength 份
angles = np.linspace(0,               # 数组第一个数据
                     2*np.pi,         # 数组最后一个数据
```

工于建构 成于创造

15.6 绘制雷达图

```
# 绘制雷达图
plt.polar(angles,                          # 设置角度
          data,                            # 设置各角度上的数据
          'rv--',                          # 设置颜色、线型和端点符号
          linewidth=2)                     # 设置线宽

# 设置角度网格标签
plt.thetagrids(angles*180/np.pi,           # 角度
               labels)                     # 标签

# 设置填充色
plt.fill(angles,                           # 设置角度
```

15.6 绘制雷达图



工于建构 成于创造

15.7 绘制三维图形

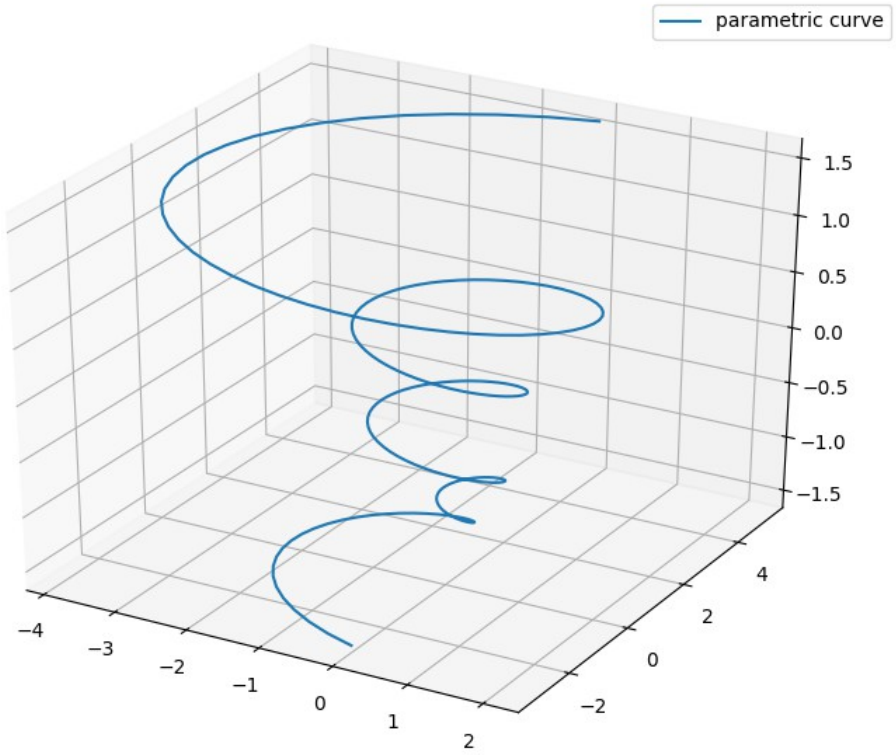
· **例 15-8** 绘制三维曲线。

```
import matplotlib as mpl
from mpl_toolkits.mplot3d import Axes3D
import numpy as np
import matplotlib.pyplot as plt


mpl.rcParams['legend.fontsize'] = 10     # 设置图例字号
fig = plt.figure()
ax = fig.gca(projection='3d')            # 绘制三维图形
theta = np.linspace(-4 * np.pi, 4 * np.pi, 200)
z = np.linspace(-4, 4, 200)*0.4          # 创建模拟数据
```

#z 的步长应与 theta 一致

15.7 绘制三维图形

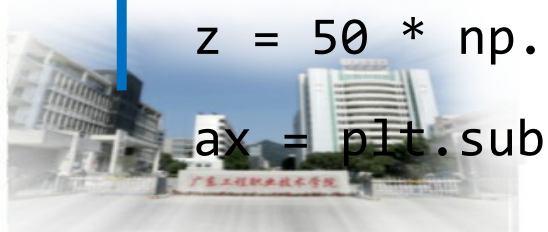
```
ax.plot(x,                                  # 设置 x 轴数据
        y,                                  # 设置 y 轴数据
        z,                                  # 设置 z 轴数据
        label='parametric curv
ax.legend()      # 显示图例

plt.show()       # 显示绘制结果
```



15.7 绘制三维图形

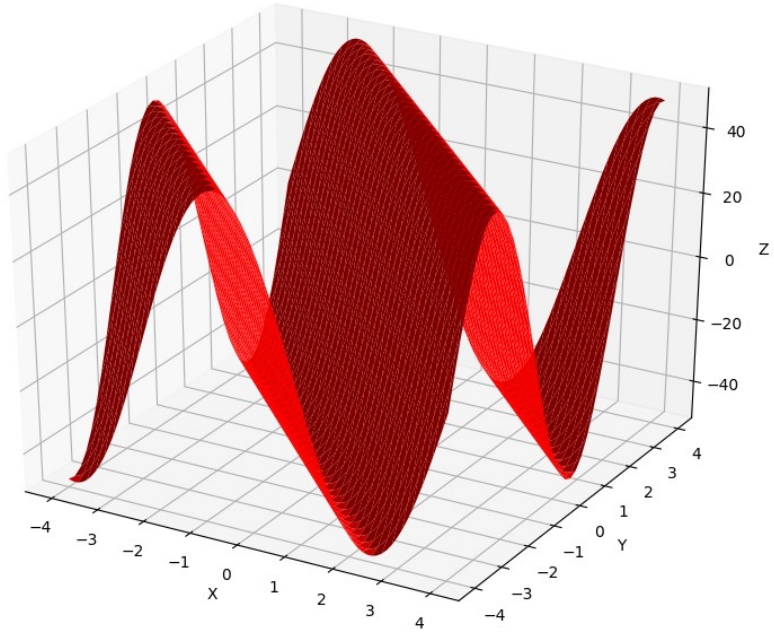
- **例 15-9** 绘制三维曲面。

```
import numpy as np
import matplotlib.pyplot as plt
import mpl_toolkits.mplot3d


x,y = np.mgrid[-4:4:80j, -4:4:40j]    # 创建 x 和 y 的网格数据
                                      # 步长使用虚数时
                                      # 虚部表示点的个数
                                      # 并且包含 end
z = 50 * np.sin(x+y)                  # 创建测试数据
ax = plt.subplot(projection='3d')     # 绘制三维图形
```



15.7 绘制三维图形

```
ax.plot_surface(x,                  # 设置 x 轴数据
                y,                  # 设置 y 轴数据
                z,                  # 设置 z 轴数据
                rstride=2,          # 行方向的步长
                cstride=1,
                color='red',
                )
ax.set_xlabel('X')
ax.set_ylabel('Y')
ax.set_zlabel('Z')
```



工于建构 成于创造

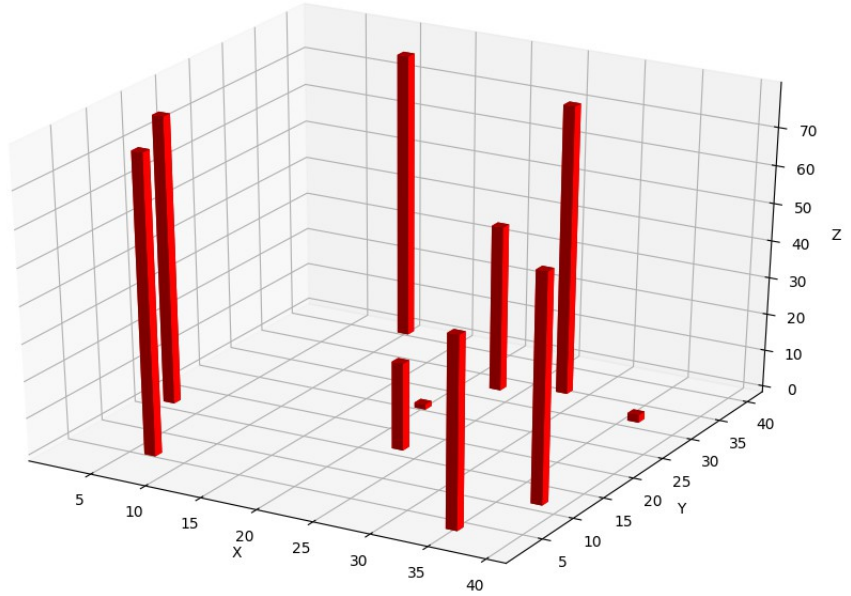
15.7 绘制三维图形

· **例 15-10** 绘制三维柱状图。

```
import numpy as np
import matplotlib.pyplot as plt
import mpl_toolkits.mplot3d


x = np.random.randint(0, 40, 10)       # 创建测试数据
y = np.random.randint(0, 40, 10)
z = 80*abs(np.sin(x+y))
ax = plt.subplot(projection='3d')    # 绘制三维图形
```

15.7 绘制三维图形

```
ax.bar3d(x,                    # 设置 x 轴数据
         y,                    # 设置 y 轴数据
         np.zeros_like(z), # 设置柱的 z 轴起始坐标为 0
         dx=1,           #x 方向的宽度
         dy=1,           #y 方向的
         dz=z,           #z 方向的
         color='red') # 设置面
ax.set_xlabel('X')
ax.set_ylabel('Y')
ax.set_zlabel('Z')
```



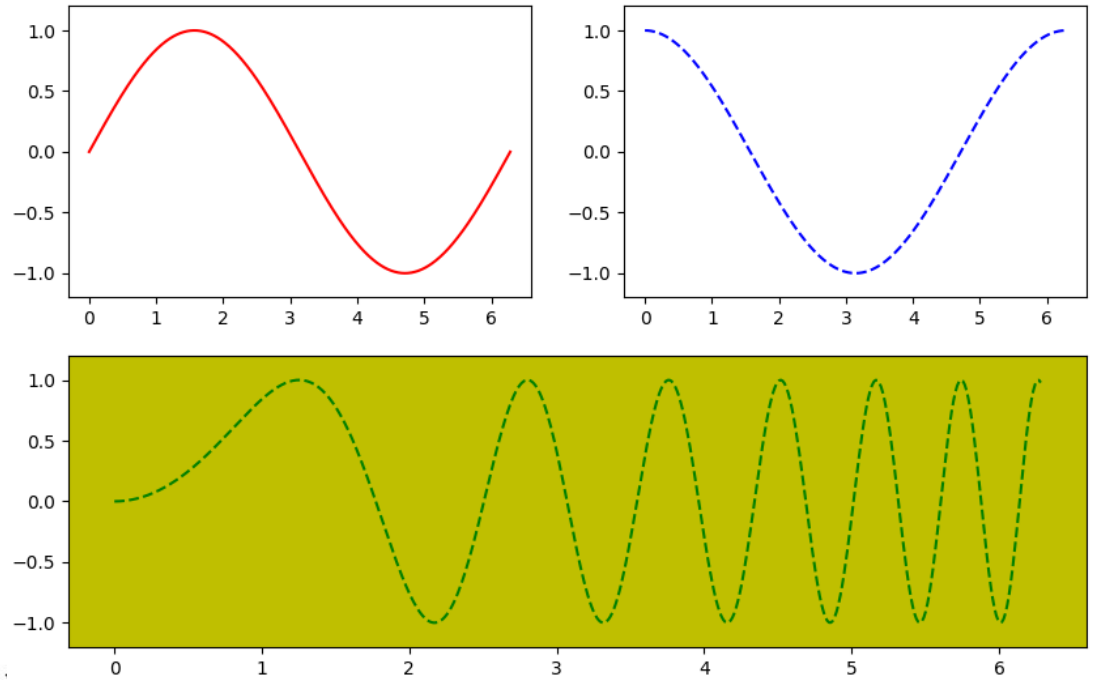
15.8 切分绘图区域

· **例 15-11** 切分绘图区域并绘制图形。

```
import numpy as np
import matplotlib.pyplot as plt


x= np.linspace(0, 2*np.pi, 500)      # 创建自变量数组
y1 = np.sin(x)                       # 创建函数值数组
y2 = np.cos(x)
y3 = np.sin(x*x)


plt.figure()                          # 创建图形
```

15.8 切分绘图区域

```
plt.sca(ax1)                          # 选择 ax1
plt.plot(x, y1, color='red')          # 绘制红色曲线
plt.ylim(-1.2, 1.2)                   # 限制 y 坐标轴范围


plt.sca(ax2)
plt.plot(x, y2, 'b--')
plt.ylim(-1.2,1.2)


plt.sca(ax3)  # 选择 ax3
plt.plot(x, y3, 'g--')
plt.ylim(-1.2, 1.2)
```



15.9 设置图例

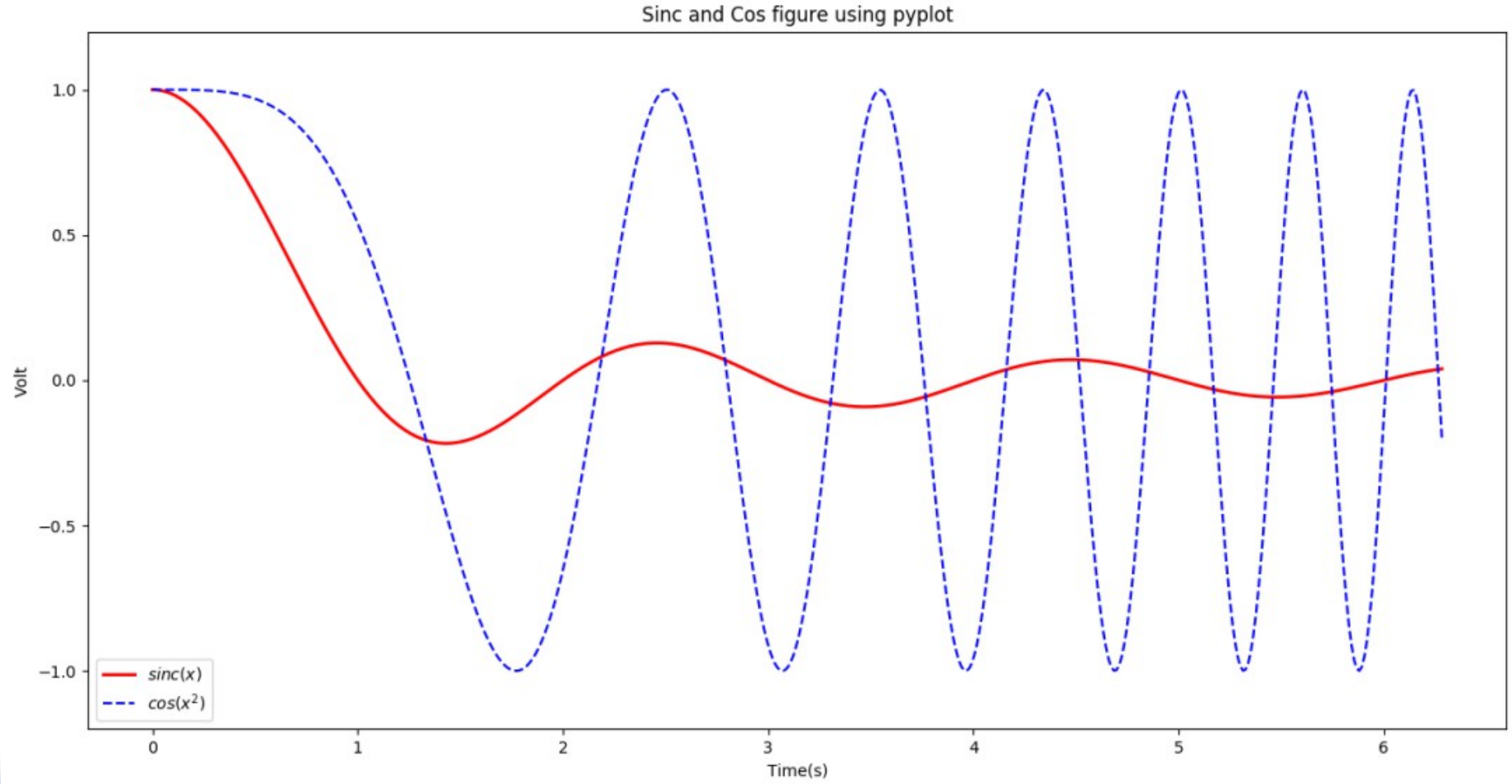
· **例 15-12** 设置图例显示公式。

```
import numpy as np
import matplotlib.pyplot as plt


x = np.linspace(0, 2*np.pi, 500)
y = np.sinc(x)
z = np.cos(x*x)
plt.figure(figsize=(8,4))


plt.plot(x,                    #x 轴数据
         y,                    #y 轴数据
```

15.9 设置图例

```
plt.plot(x,
         z,
         'b--',                  # 蓝色虚线
         label='$cos(x^2)$') # 把标签渲染为公式


plt.xlabel('Time(s)')
plt.ylabel('Volt')
plt.title('Sinc and Cos figure using pyplot')
plt.ylim(-1.2,1.2)
plt.legend()                     # 显示图例
```

15.9 设置图例



工于建构 成于创造

15.9 设置图例

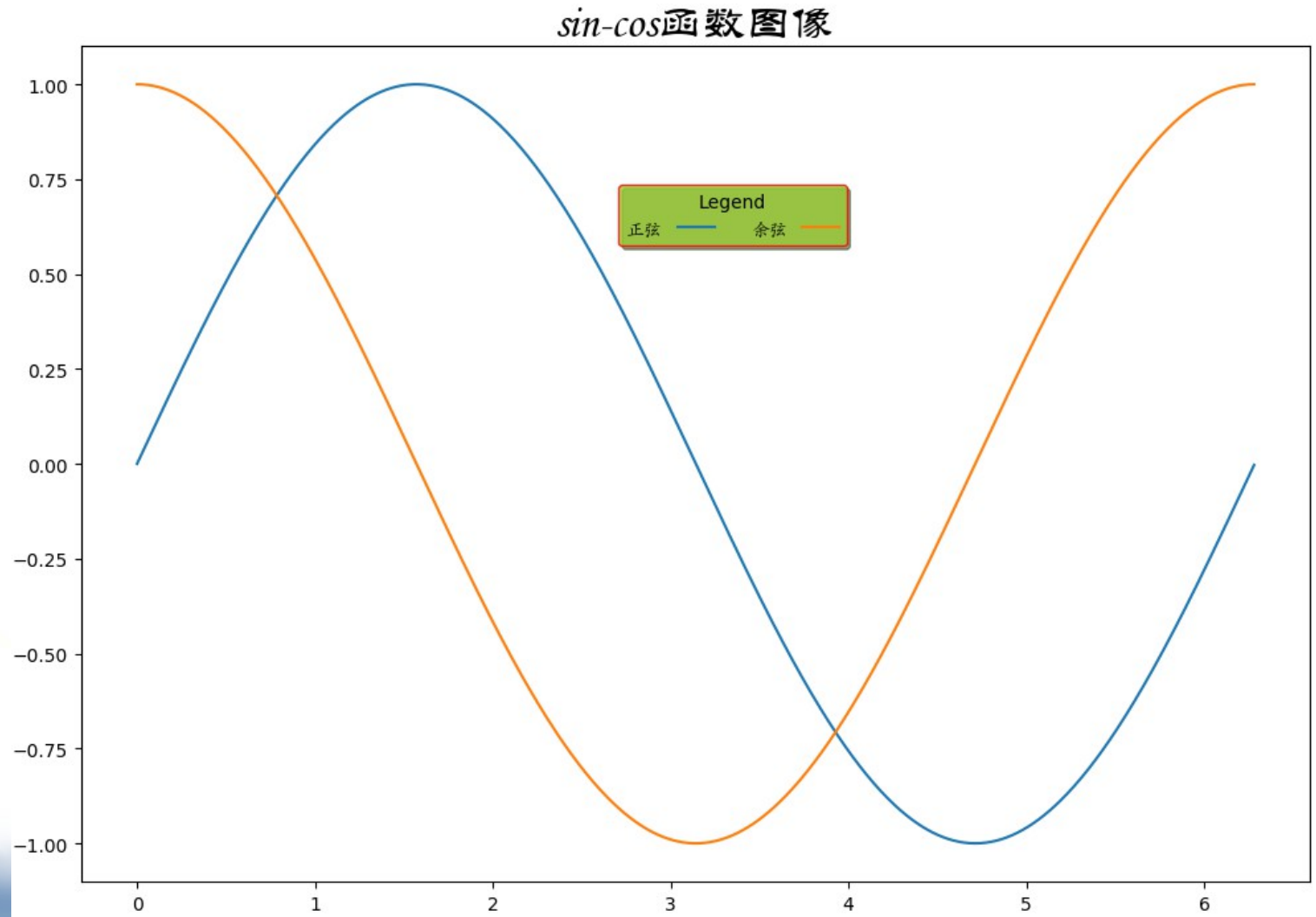
· **例 15-13** 设置图例位置、背景颜色、边框颜色等属性。

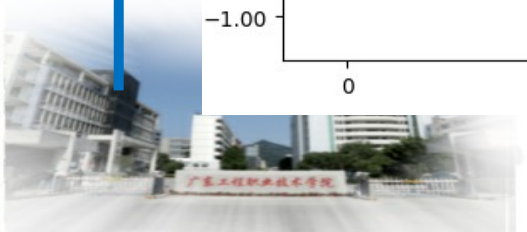
```
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.font_manager as fm


t = np.arange(0.0, 2*np.pi, 0.01)
s = np.sin(t)
z = np.cos(t)


plt.plot(t, s, label='正弦')
plt.plot(t, z, label='余弦')
plt.title('sin,cos函数图像')                    #设置图形标题文本
```

15.9 设置图例

```
myfont = fm.FontProperties(fname=r'C:\Windows\Fonts\STKAITI.ttf')
plt.legend(prop=myfont,                          # 设置图例字体
           title='Legend',                       # 设置图例标题
           loc='lower left',                     # 设置图例参考位置
           bbox_to_anchor=(0.43,0.75),# 设置图例位置偏移量
           shadow=True,                          # 显示阴影
           facecolor='yellowgreen',    # 设置图例背景色
           edgecolor='red',                      # 设置图例边框颜色
           ncol=2,                               # 显示为两列
           markerfirst=False)                    # 设置图例文字在前，符号在后
```

15.9 设置图例





工于建构 成于创造

15.10 设置坐标轴刻度距离和文本

· **例 15-14** 设置坐标轴刻度距离和文本。

```
import numpy as np
import matplotlib.pyplot as plt


x = np.arange(0, 2*np.pi, 0.01)
y = np.sin(x)
plt.plot(x, y)


plt.xticks(np.arange(0, 2*np.pi, 0.5))
plt.yticks([-1, -0.5, 0, 0.75, 1],
           ['负一', '负零点五', '零', '零点七五', '一'],
```

15.10 设置坐标轴刻度距离和文本