









Python

数据分析及可视化-III

数据图表可视化



本章学习目标

-  掌握扩展库matplotlib及其依赖库的安装
-  了解matplotlib绘图的一般过程
-  熟练掌握常用图形的绘制与属性设置
-  了解多维图的内涵及绘制
-  熟练掌握绘图区域的切分与属性设置
-  熟练掌握图例属性的设置及绘图结果的保存





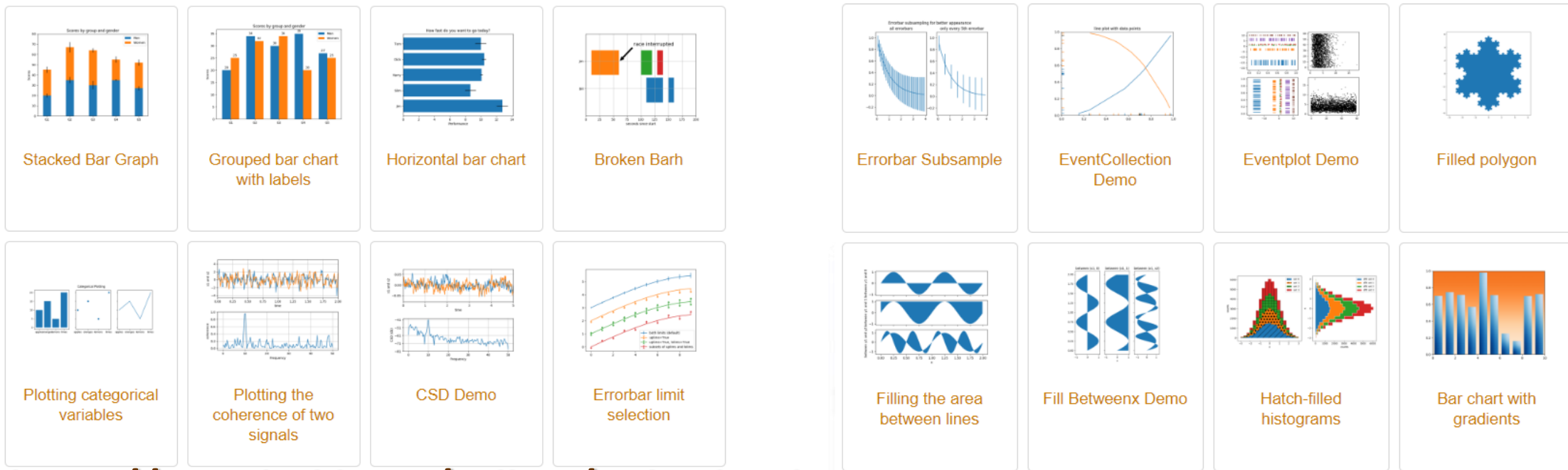
matplotlib 库概述

什么是Matplotlib ?

Matplotlib是一个Python的2D与3D绘图库，它以各种硬拷贝格式和跨平台的交互式环境生成出版质量级别的图形。



Lines, bars and markers



<https://matplotlib.org/gallery/index.html>





matplotlib 库概述

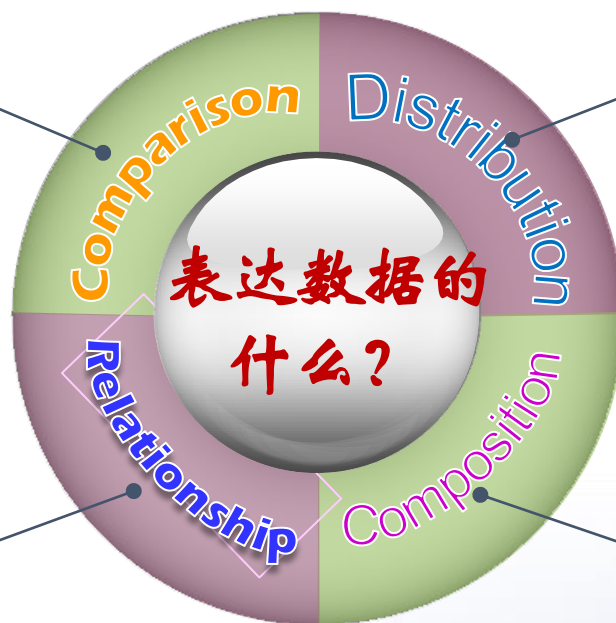
数据表达与图表逻辑

Comparison
- 比较

Distribution
- 分布

Relationship
- 关系

Composition
- 构成





matplotlib库概述

数据表达与图表逻辑



比较 Comparison

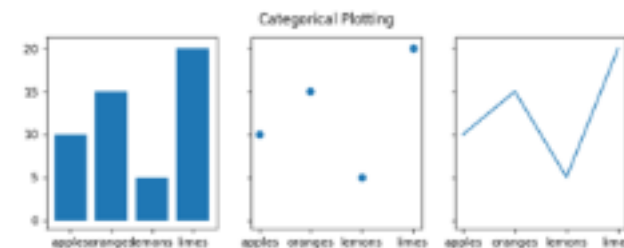
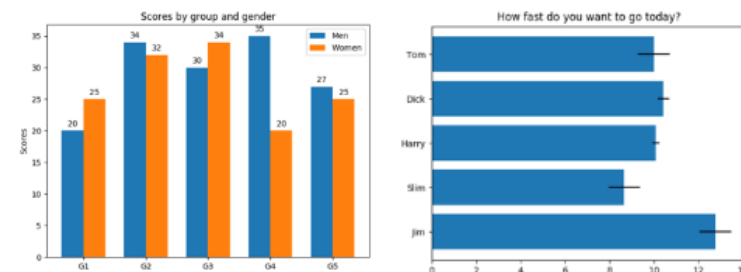
用于比较的图表类型：基于类别、基于时间

➤ 基于类型：

- ✓ 不等宽的柱形图（两个变量：高度/宽度）
- ✓ 矩阵图表、条形图、柱状图（一个变量：高度）

➤ 基于时间：

- ✓ 雷达图、折线图、柱状图（多周期/少周期）



分布 Distribution

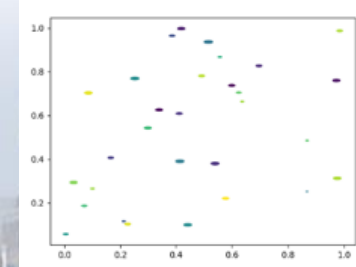
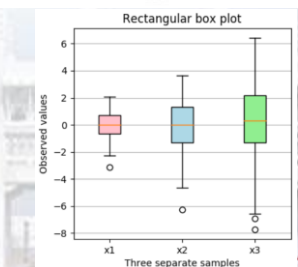
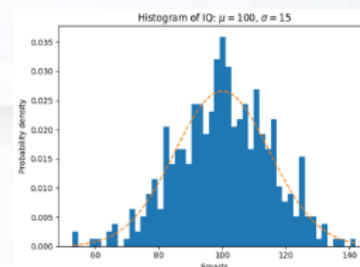
用于分布的图表类型：单个变量、多个变量

➤ 单个变量：

- ✓ 直方图、密度图、箱形图

➤ 多个变量：

- ✓ 散点图、多维密度图





matplotlib 库概述

数据表达与图表逻辑

构成 Composition

用于比较的图表类型：“静态”、“基于时间”

➤ 静态：

✓ 饼图、瀑布图、堆积图

➤ 基于时间：

✓ 堆积图、面积图（相对差异/绝对差异）



关系 Relationship

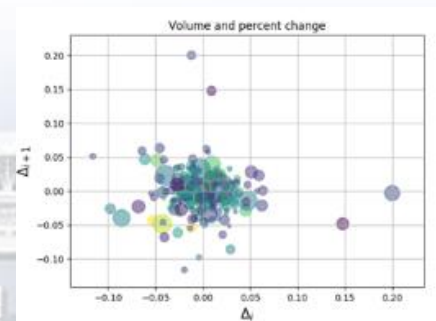
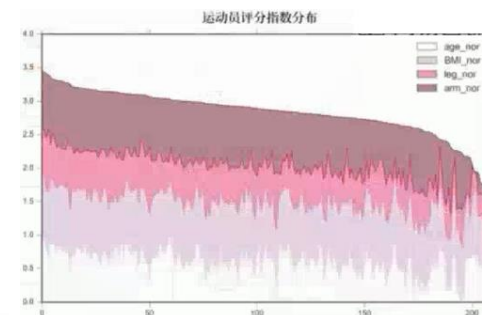
用于关系的图表类型：2个变量、多个变量

➤ 2个变量：

✓ 散点图

➤ 多个变量：

✓ 气泡图

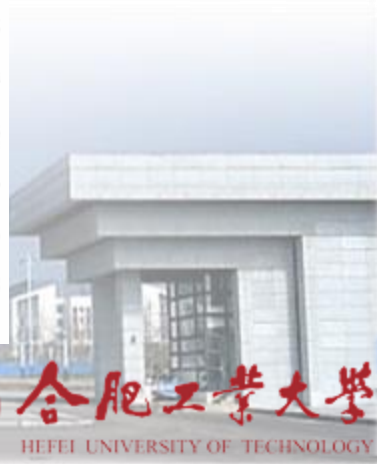




matplotlib库概述

数据表达与图表逻辑

主类	次类	用途
折线图	折线图	用于反映和时间相关的数据变化(趋势)
	面积图	用于反映主次之间的基于时间的对比
柱状图	柱状图	分类项目的数量比较, 也可能反映趋势
	条形图	分类项目的数量比较
	环状条形图	分类项目的数量比较, 更反映分类项目之间的数量关系
	南丁格尔玫瑰图	以夸张的形势来表示分类项目的数量比较
饼状图	饼状图	反映分类数据所占比例
	旭日图	表示比例对比的同时, 也表示层级关系
	树状图	表示比例, 可以多层级
散点图	散点图	反映相关性和分布关系, 两个变量
	气泡图	反映相关性和分布关系, 三个变量
	带线散点	趋势线
地图	地图	反映空间信息
其他图	股价图/箱线图	反映数据分散情况
	直方图	反映数据分布
	瀑布图	数量变化的过程

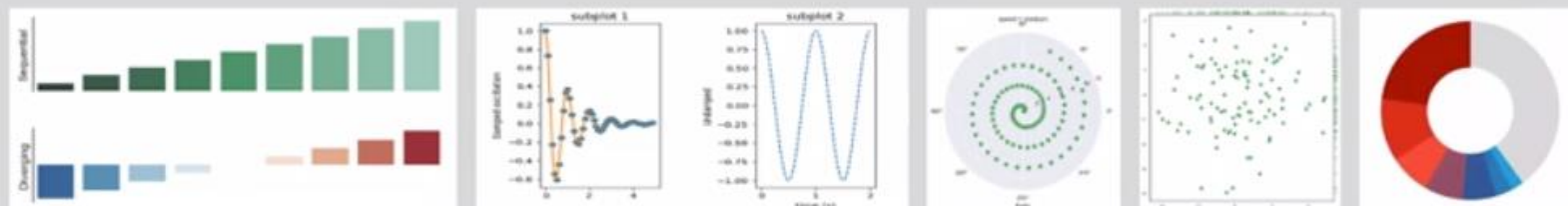




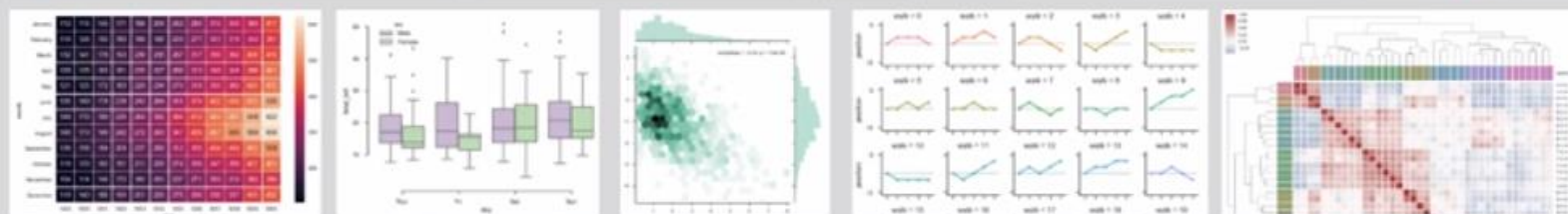
matplotlib库概述

数据表达与图表逻辑

单维度



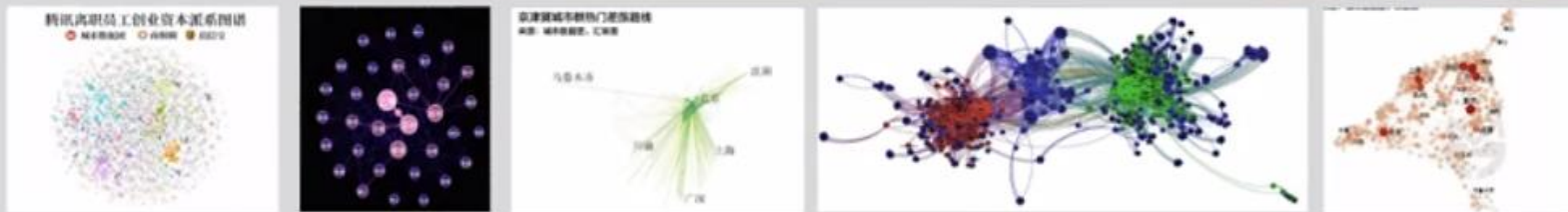
多维度



空间图表



关系图表





matplotlib库的使用

Pyplot子模块

matplotlib库由一系列有组织有隶属关系的对象构成，这对于基础绘图操作来说显得过于复杂。因此，matplotlib 提供了一套快捷命令式的绘图接口函数，即pyplot子模块。pyplot 将绘图所需要的对象构建过程封装在函数中，对用户提供了更加友好的接口。pyplot 模块提供一批预定义的绘图函数，大多数函数可以从函数名辨别它的功能。

 matplotlib.pyplot 是matplotlib 的子库，引用方式如下：

```
>>>import matplotlib.pyplot as plt
```

 为了正确显示中文字体，请用以下代码更改默认设置，其中'SimHei'表示黑体字。

```
>>> plt.rcParams['font.sans-serif']=['SimHei']
```

```
#在matplotlib中设置字体为SimHei黑体。
```

```
>>> plt.rcParams['axes.unicode_minus']=False
```

```
#解决matplotlib坐标轴负号 ‘-’ 显示为方块问题
```





matplotlib 库的使用

字体

字体是计算机显示字符的方式，均由人工设计，并采用字体库方式部署在计算机中。西文和中文字体都有很多种类，下表给出最常用的10 种中文字体及其英文表示，这些字体的英文表示在程序设计中十分常用，但需要注意，部分字体无法在matplotlib 库中使用。

字体名称	字体英文表示
宋体	SimSun
黑体	SimHei
楷体	KaiTi
微软雅黑	Microsoft YaHei
隶书	LiSu
仿宋	FangSong
幼圆	YouYuan
华文宋体	STSong
华文黑体	STHeiti
苹果丽中黑	Apple LiGothic Medium



matplotlib库的使用

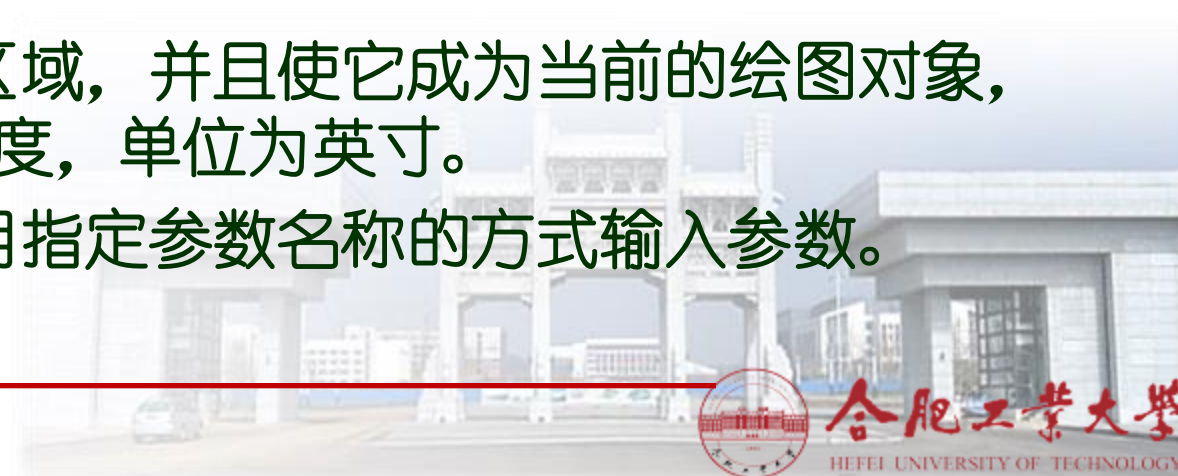
plt 库的绘图区域函数

<code>plt.figure(figsize=None, facecolor=None)</code>	创建一个全局绘图区域
<code>plt.axes(rect, axisbg='w')</code>	创建一个坐标系风格的子绘图区域
<code>plt.subplot(nrows, ncols, plot_number)</code>	在全局绘图区域中创建一个子绘图区域
<code>plt.subplots_adjust()</code>	调整子图区域的布局

使用`figure()`函数创建一个全局绘图区域，并且使它成为当前的绘图对象，`figsize`参数可以指定绘图区域的宽度和高度，单位为英寸。

鉴于`figure()`函数参数较多，这里采用指定参数名称的方式输入参数。

```
>>> plt.figure(figsize=(8, 4))
```



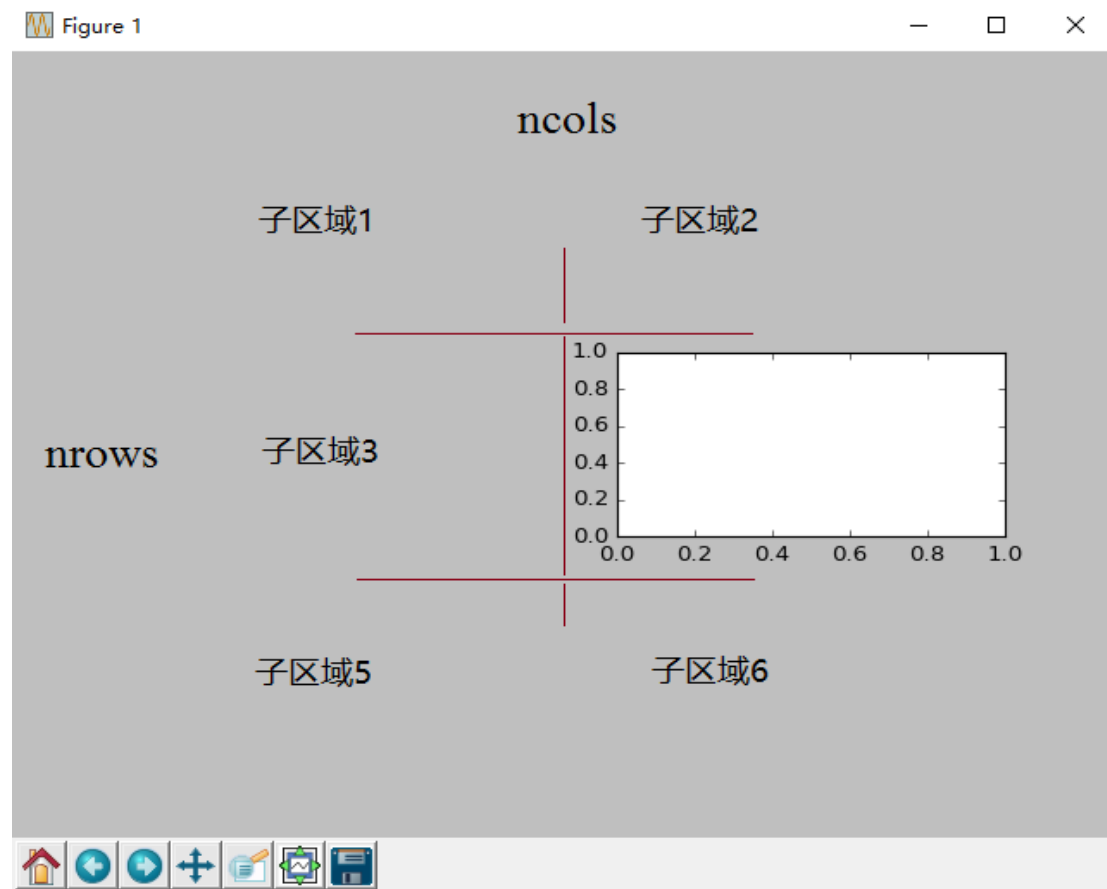


matplotlib 库的使用

plt 库的绘图区域函数

`subplot()` 都用于在全局绘图区域内创建子绘图区域，其参数表示将全局绘图区域分成 `nrows` 行和 `ncols` 列，并根据先行后列的计数方式在 `plot_number` 位置生成一个坐标系，实例代码如下，三个参数关系如图所示。其中，全局绘图区域被分割成 3×2 的网格，其中，在第4个位置绘制了一个坐标系。

```
>>> plt.subplot(324)
>>> plt.show()
```





matplotlib 库的使用

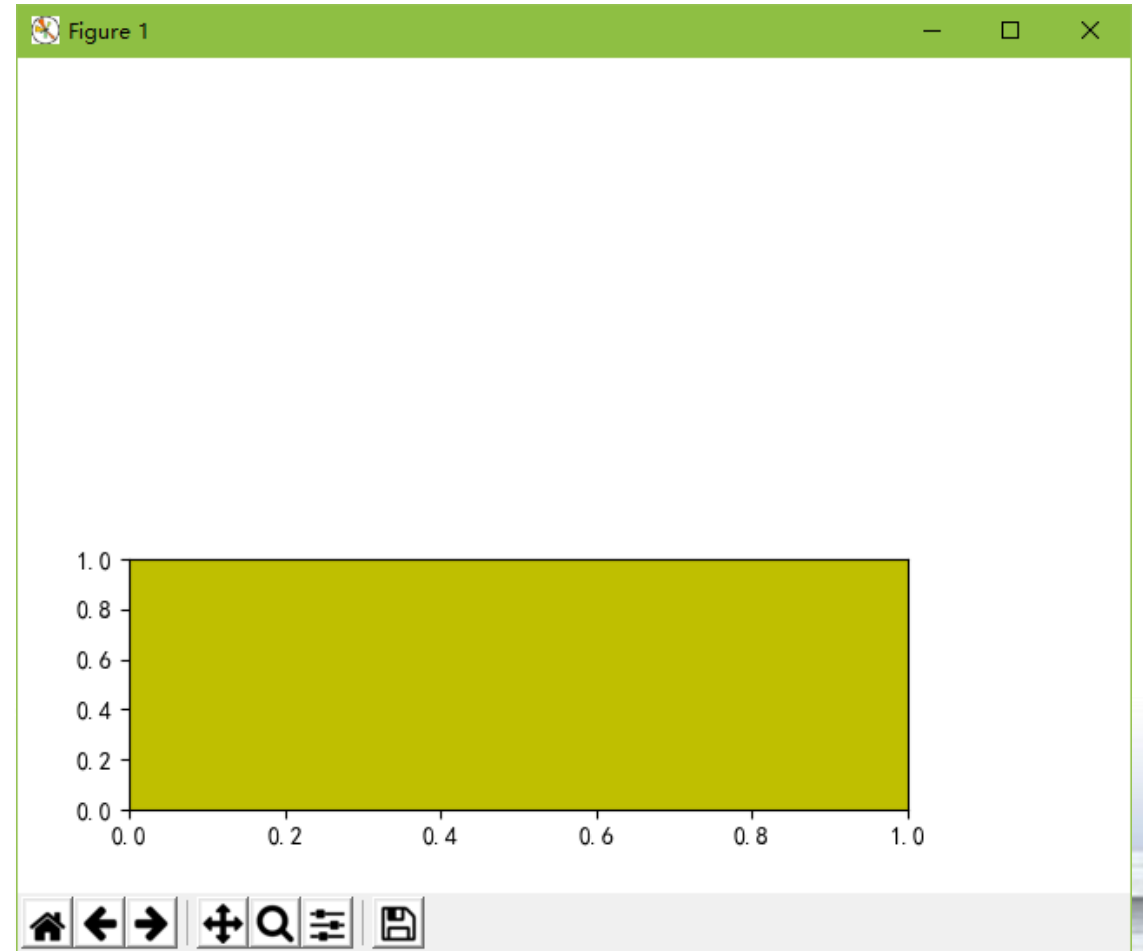
plt 库的绘图区域函数

`axes()` 默认创建一个 subplot (111)
坐标系，参数：

`rect=[left, bottom, width, height]`

其中，四个变量的范围都为 $[0, 1]$ ，
表示坐标系与全局绘图区域的关系；
`facecolor` 指背景色，默认为 white。

```
>>> plt.figure(figsize=(8, 6))  
>>> plt.axes([0.1, 0.1, 0.7, 0.3], facecolor= 'y' )  
>>> plt.show()
```





matplotlib 库的使用

plt 库的读取和显示函数

plt 子库提供了一组读取和显示相关函数，用于在绘图区域中增加显示内容及读入数据，如下表所示，这些函数需要与其他函数搭配使用。

函数	描述
plt.legend()	在绘图区域中方式绘图标签（也称图注）
plt.show()	显示创建的绘图对象
plt.matshow()	在窗口显示数组矩阵
plt.imshow()	在 axes 上显示图像
plt.imsave()	保存数组为图像文件
plt.imread()	从图像文件中读取数组





matplotlib库的使用

plt 库的基础图表函数

常用颜色字母

```
=====
character  color
=====
```

'b'	blue 蓝
'g'	green 绿
'r'	red 红
'c'	cyan 蓝绿
'm'	magenta 洋红
'y'	yellow 黄
'k'	black 黑
'w'	white 白

```
=====
```

操作	描述
plt.plot(x, y, label, color, width)	根据 x, y 数组绘制直/曲线
plt.boxplot(data, notch, position)	绘制一个箱型图 (Box-plot)
plt.bar(left, height, width, bottom)	绘制一个条形图
plt.barh(bottom, width, height, left)	绘制一个横向条形图
plt.polar(theta, r)	绘制极坐标图
plt.pie(data,explode)	绘制饼图
plt.psd(x, NFFT=256, pad_to, Fs)	绘制功率谱密度图
plt.specgram(x, NFFT=256, pad_to, F)	绘制谱图
plt.cohere (x, y, NFFT=256, Fs)	绘制 X-Y 的相关性函数
plt.scatter()	绘制散点图 (x, y 是长度相同的序列)
plt.step(x, y, where)	绘制步阶图
plt.hist(x, bins, normed)	绘制直方图





matplotlib 库的使用

plt 库的坐标轴设置函数

plt 库有两个坐标体系: **图像坐标**和**数据坐标**。

图像坐标将图像所在区域左下角视为原点, 将x 方向和y 方向长度设定为1。整体绘图区域有一个图像坐标, 每个axes()和subplot()函数产生的子图也有属于自己的图像坐标。axes()函数参数rect 指当前产生的子区域相对于整个绘图区域的图像坐标。

数据坐标以当前绘图区域的坐标轴为参考, 显示每个数据点的相对位置, 这与坐标系里面标记数据点一直。

函数	描述
plt.axis('v','off','equal','scaled','tight','image')	获取设置轴属性的快捷方法
plt.xlim(xmin, xmax)	设置当前 x 轴取值范围
plt.ylim(ymin, ymax)	设置当前 y 轴取值范围
plt.xscale()	设置 x 轴缩放
plt.yscale()	设置 y 轴缩放
plt.autoscale()	自动缩放轴视图的数据
plt.thetagrids(angles, labels, fnt, frac)	设置极坐标网格 theta 的位置
plt.grid(on/off)	打开或者关闭坐标网格



matplotlib库的使用

plt 库的标签设置函数

函数	描述
<code>plt.figlegend(handles,label,loc)</code>	为全局绘图区域放置图注
<code>plt.legend()</code>	为当前坐标图放置图注
<code>plt.xlabel(s)</code>	设置当前x轴的标签
<code>plt.ylabel(s)</code>	设置当前y轴的标签
<code>plt.xticks(array,'a','b','c')</code>	设置当前x轴刻度位置的标签和值
<code>plt.yticks(array,'a','b','c')</code>	设置当前y轴刻度位置的标签和值
<code>plt.get_figlabels()</code>	返回当前绘图区域的标签列表
<code>plt.figtext(x,y,s,fontdic)</code>	为全局绘图区域添加文字
<code>plt.title(s)</code>	设置标题
<code>plt.subtitle(s)</code>	为当前绘图区域添加中心标题
<code>plt.text(x,y,s,fontdic,withdash)</code>	为坐标图轴添加注释
<code>plt.annotate(note,xy,xytext,xycoords, textcoords,arrowprops)</code>	用移送在指定数据占创建一个注释或一段文本





实例 plt 库的基础图表

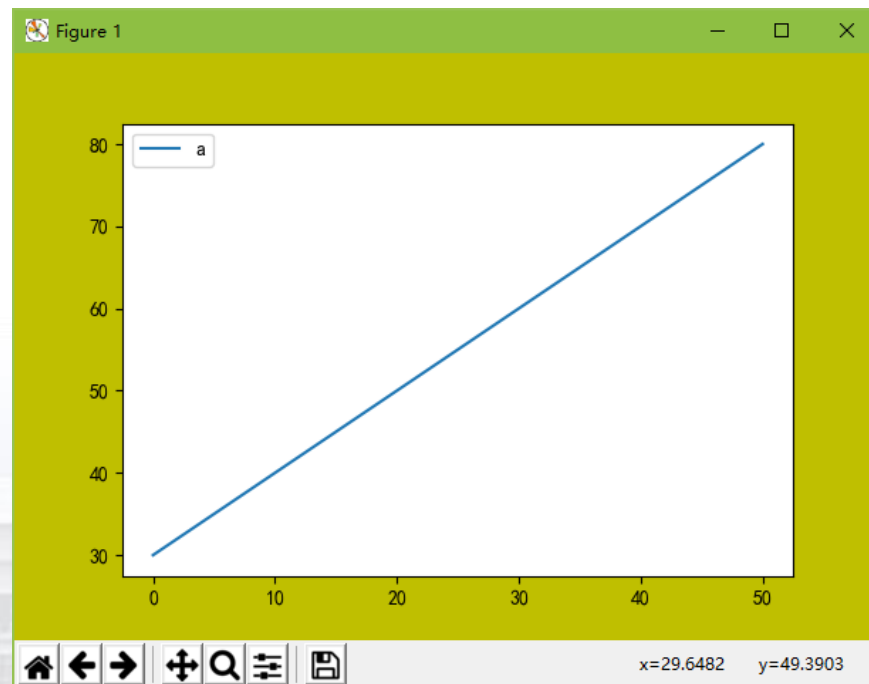
figure() 创建画布

在Pypplot模块中，默认拥有一个Figure对象，该对象可以理解为一块空白的画布，用于容纳图表的各种组件，比如图例、坐标轴等。

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

```
#Matplotlib中设置字体-黑体
plt.rcParams['font.sans-serif']=['Simhei']
#解决Matplotlib坐标轴负号'-'显示为方块
plt.rcParams['axes.unicode_minus']=False
```

```
>>> plt.figure(figsize=(8,6), facecolor='y')
<Figure size 800x600 with 0 Axes>
>>> plt.plot(data2)
[<matplotlib.lines.Line2D object at 0x00952590>]
>>> plt.legend('a')
<matplotlib.legend.Legend object at 0x07F303B0>
>>> plt.show()
```

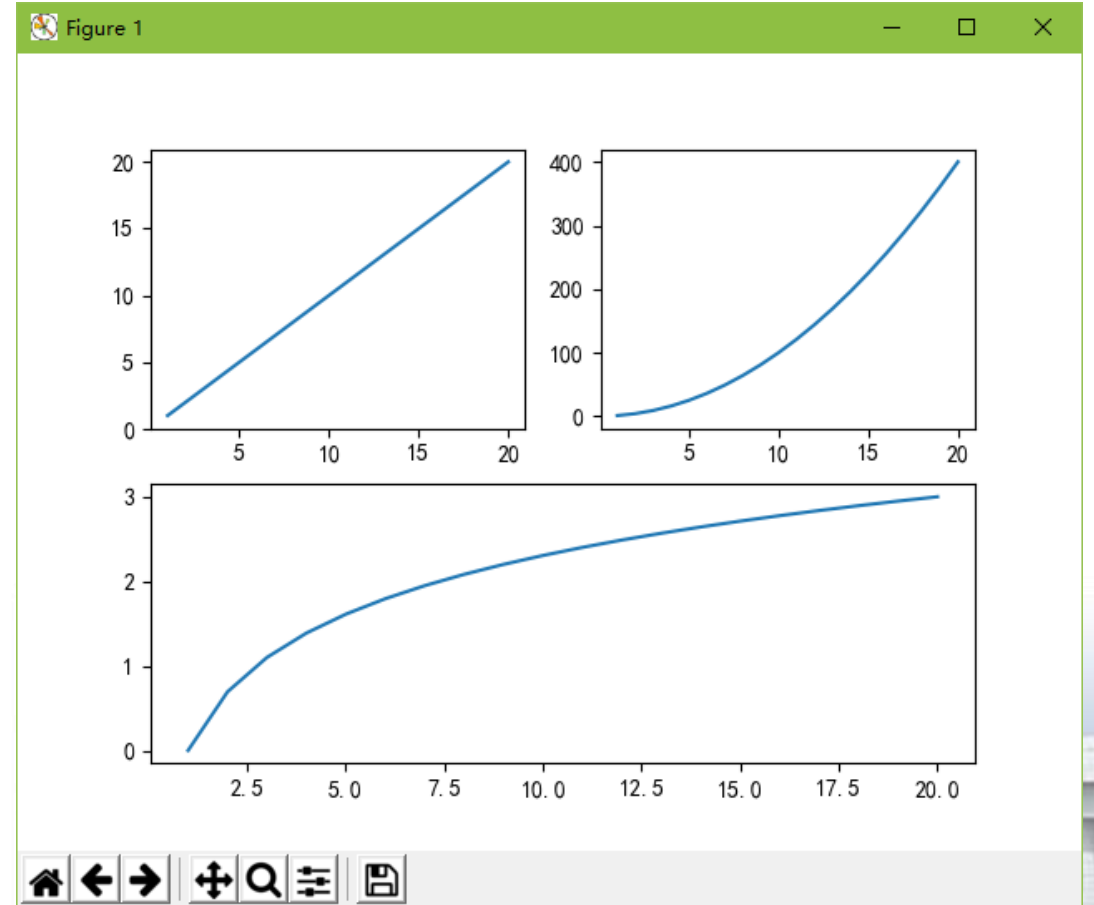




实例 plt 库的基础图表

subplot() 创建多个子图

```
>>> x=np.arange(1, 21)
>>> plt.subplot(221) #绘图一
<matplotlib.axes._subplots.AxesSubplot object at 0x07A7D750>
>>> plt.plot(x, x)
[<matplotlib.lines.Line2D object at 0x00DA86D0>]
>>> plt.subplot(222) #绘图二
<matplotlib.axes._subplots.AxesSubplot object at 0x009528B0>
>>> plt.plot(x, x**2)
[<matplotlib.lines.Line2D object at 0x00DA89B0>]
>>> plt.subplot(212) #绘图三
<matplotlib.axes._subplots.AxesSubplot object at 0x00DA86B0>
>>> plt.plot(x, np.log(x))
[<matplotlib.lines.Line2D object at 0x00EF6CB0>]
>>> plt.show()
```

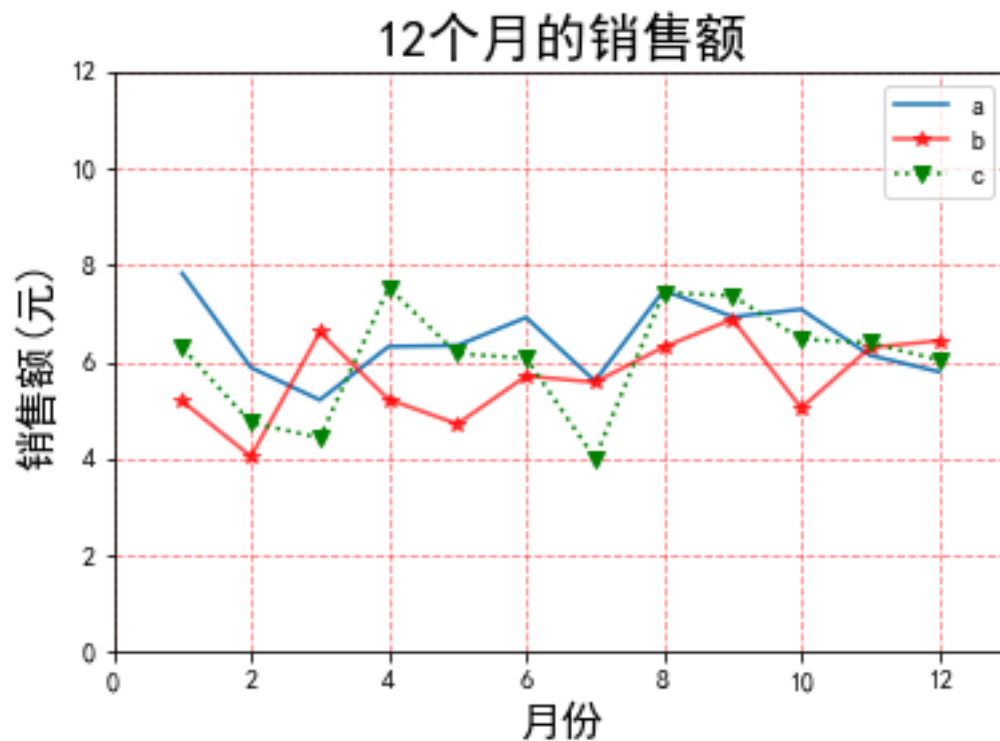




实例 plt 库的基础图表

绘制折线图

```
y=np.random.uniform(5, 8, 12)
x=np.arange(1, 13)
plt.plot(x, y)
plt.xlim(0, 13)
plt.ylim(0, 12)
plt.title("12个月的销售额", fontsize=20)
plt.xlabel('月份', fontsize=16)
plt.ylabel('销售额(元)', fontsize=16)
plt.grid(linestyle='--', color='r', alpha=0.5)
m=np.random.uniform(4, 7, 12)
n=np.random.uniform(4, 8, 12)
plt.plot(x, m, 'r-*', alpha=0.7)
plt.plot(x, n, 'g:v')
plt.legend(['a', 'b', 'c'])
plt.show()
```





实例 plt 库的基础图表



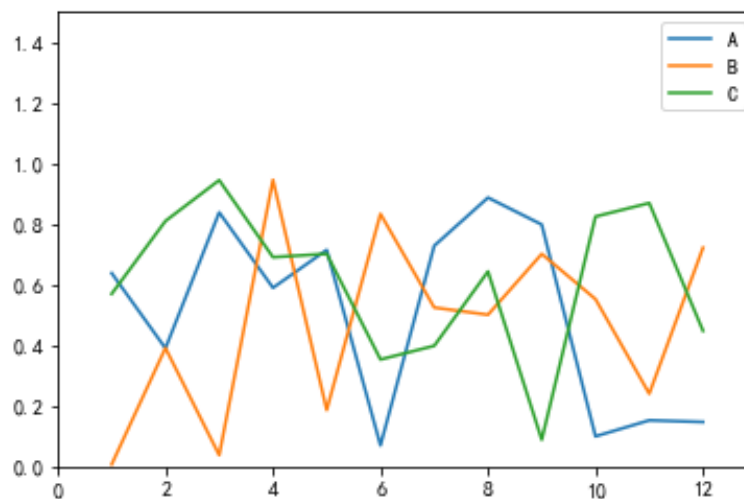
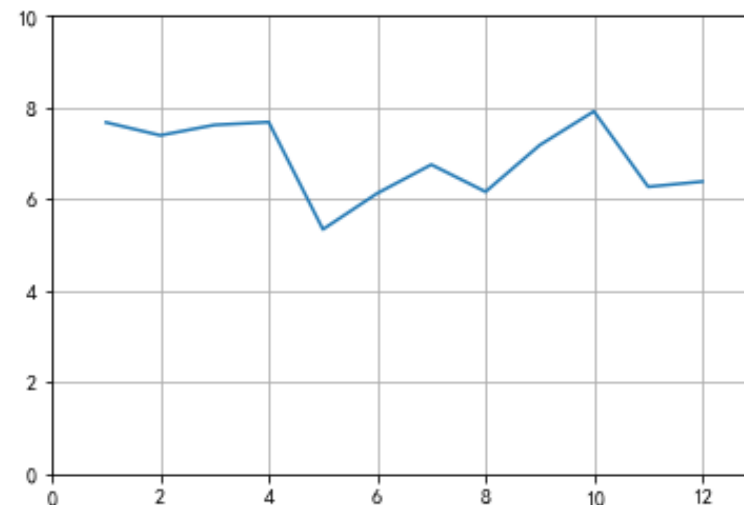
绘制折线图

```
ds=pd.Series(np.random.uniform(5, 8, 12),  
             index=np.arange(1, 13))
```

```
ds.plot()  
plt.xlim(0, 13)  
plt.ylim(0, 10)  
plt.grid(True)  
plt.show()
```

```
df=pd.DataFrame(np.random.rand(12, 3),  
               columns=['A', 'B', 'C'],  
               index=np.arange(1, 13))
```

```
df.plot()  
plt.xlim(0, 13)  
plt.ylim(0, 1.5)  
plt.show()
```



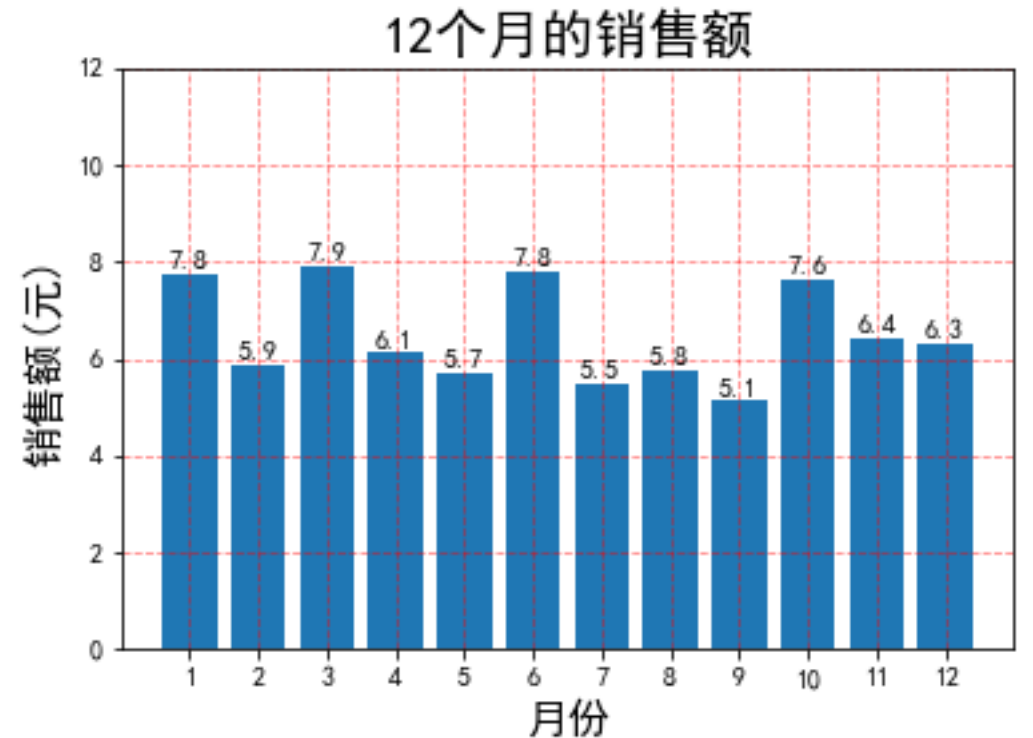


实例 plt 库的基础图表



绘制柱状图

```
y=np.random.uniform(5, 8, 12)
x=np.arange(1, 13)
plt.bar(x, y)
plt.xlim(0, 13)
plt.ylim(0, 12)
plt.title("12个月的销售额", fontsize=20)
plt.xlabel('月份', fontsize=16)
plt.ylabel('销售额(元)', fontsize=16)
plt.grid(linestyle='--', color='r', alpha=0.5)
plt.xticks(list(x))
for a, b in zip(x, y):
    plt.text(a-0.3, b+0.1, '%.1f' %b)
plt.show()
```



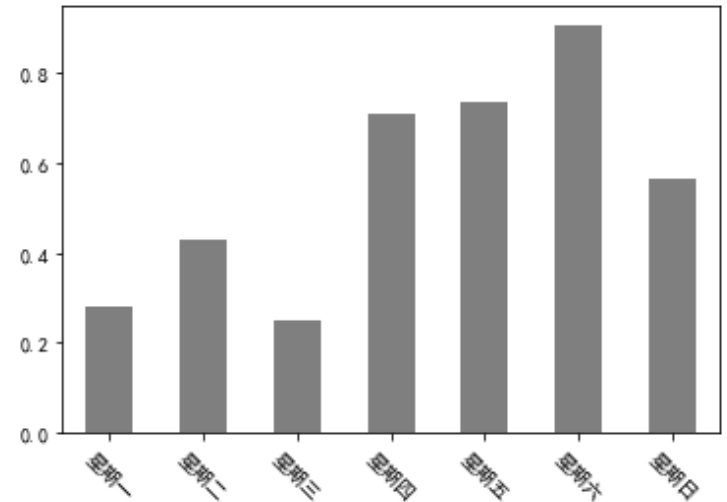


实例 plt 库的基础图表

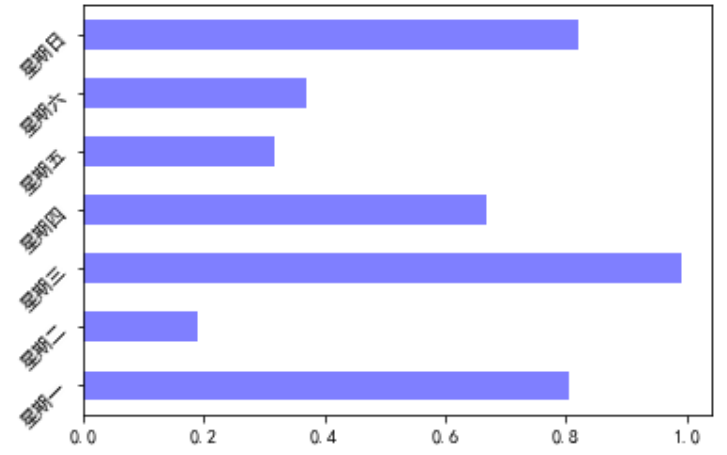


绘制柱状图

```
ds=pd.Series(np.random.rand(7),  
             index=['星期一','星期二','星期三',  
                   '星期四','星期五','星期六','星期日'])  
ds.plot(kind="bar",color='k',alpha=0.5)  
plt.xticks(rotation=-45)  
plt.show()
```



```
ds=pd.Series(np.random.rand(7),  
             index=['星期一','星期二','星期三',  
                   '星期四','星期五','星期六','星期日'])  
ds.plot(kind="barh",color='b',alpha=0.5)  
plt.yticks(rotation=45)  
plt.show()
```



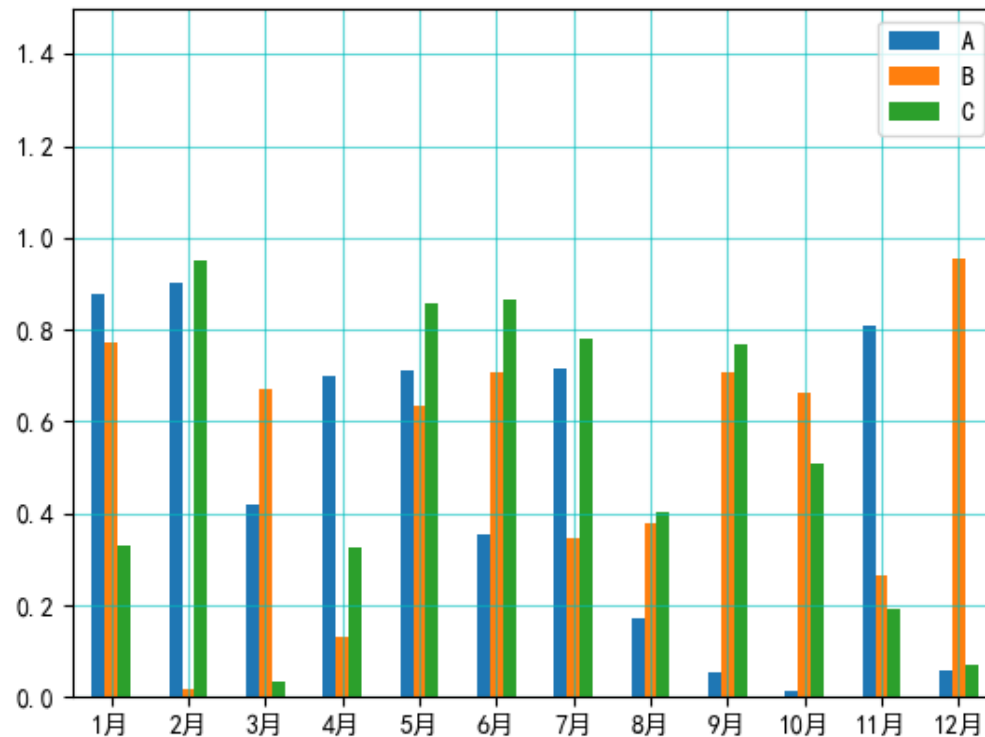


实例 plt 库的基础图表



绘制柱状图

```
month=['1月','2月','3月','4月','5月','6月','7月',  
       '8月','9月','10月','11月','12月']  
df=pd.DataFrame(np.random.rand(12,3),  
                 columns=['A','B','C'],  
                 index=month)  
  
df.plot(kind='bar')  
plt.ylim(0,1.5)  
plt.xticks(rotation=0)  
plt.grid(color='c',alpha=0.6)  
plt.show()
```





实例 plt 库的基础图表

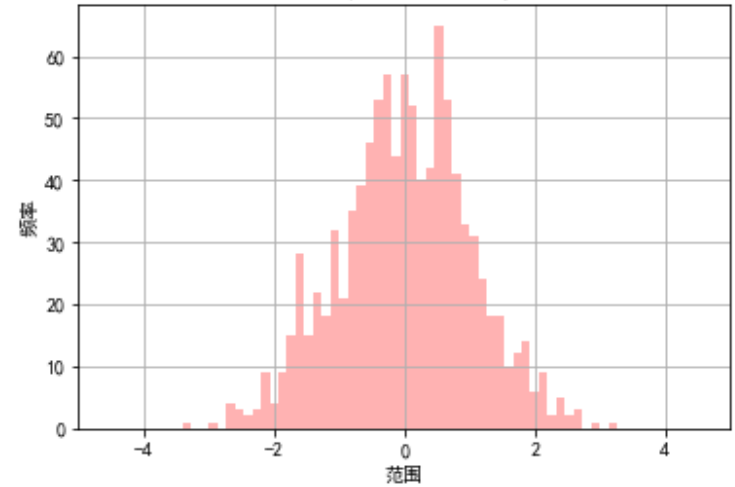


绘制直方图

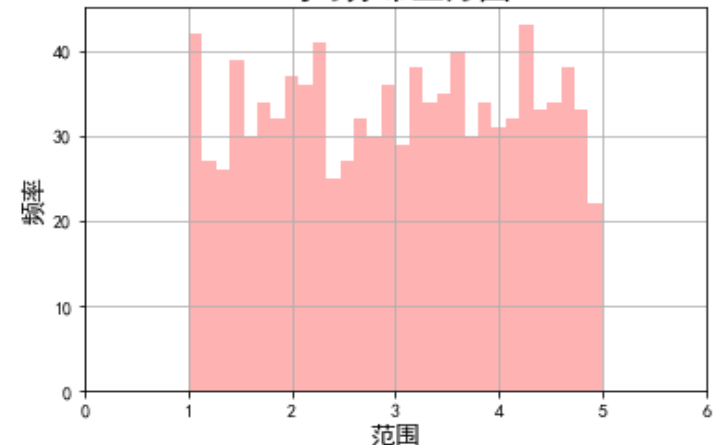
```
y=np.random.normal(0, 1, 1000)
plt.xlim(-5, 5)
plt.hist(y, color='r', alpha=0.3, bins=50)
plt.title("正态分布直方图", fontsize=20)
plt.xlabel("范围")
plt.ylabel("频率")
plt.grid(True)
```

```
y=np.random.uniform(1, 5, 1000)
plt.xlim(0, 6)
plt.hist(y, color='r', alpha=0.3, bins=30)
plt.title("均匀分布直方图", fontsize=18)
plt.xlabel("范围", fontsize=14)
plt.ylabel("频率", fontsize=14)
plt.grid(True)
```

正态分布直方图



均匀分布直方图



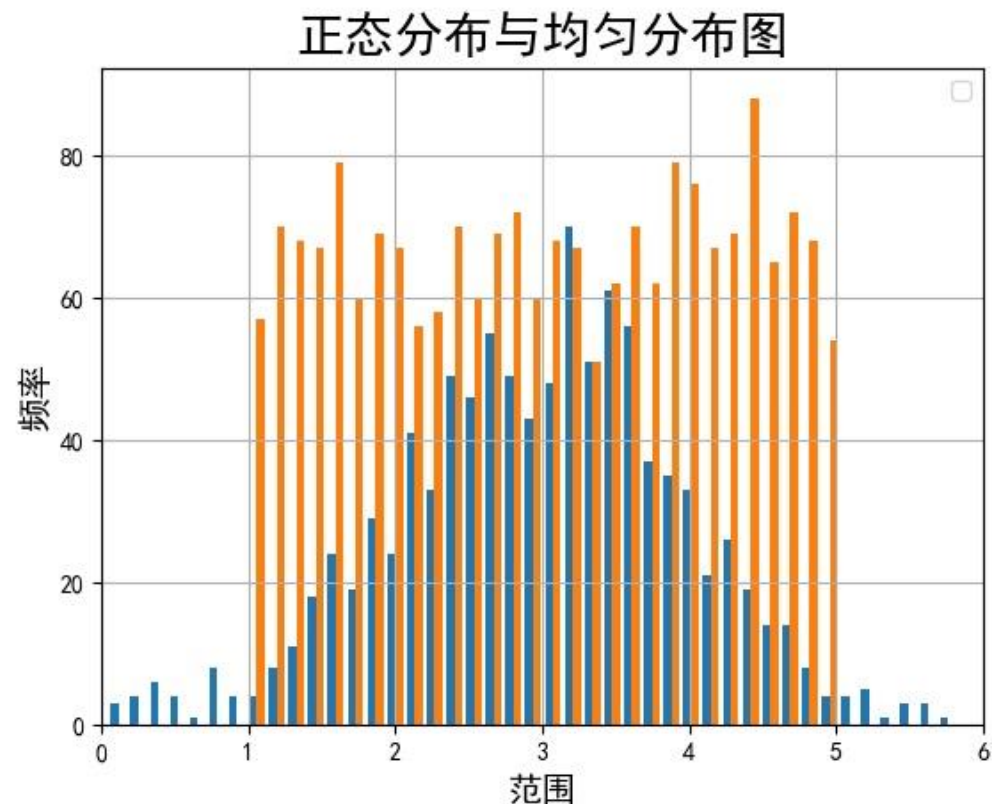


实例 plt 库的基础图表



绘制直方图

```
y=np.random.normal(3, 1, 1000)
z=np.random.uniform(1, 5, 2000)
plt.xlim(0, 6)
plt.xlabel("范围", fontsize=14)
plt.title("正态分布与均匀分布图", fontsize=20)
plt.ylabel("频率", fontsize=14)
plt.grid(True)
plt.hist([y, z], bins=50)
plt.show()
```



合肥工业大学
HEFEI UNIVERSITY OF TECHNOLOGY



实例 plt 库的基础图表



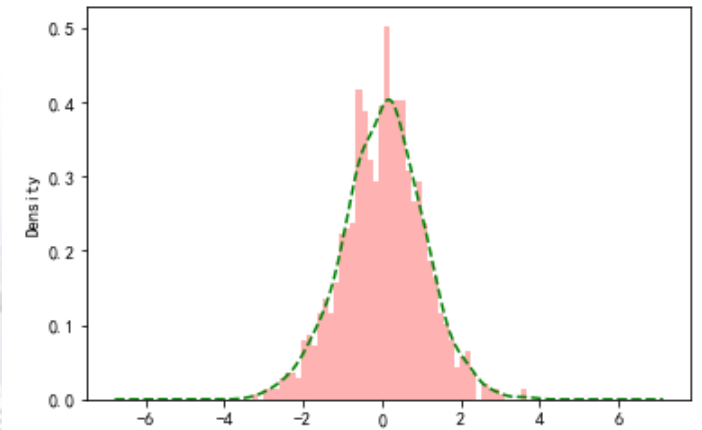
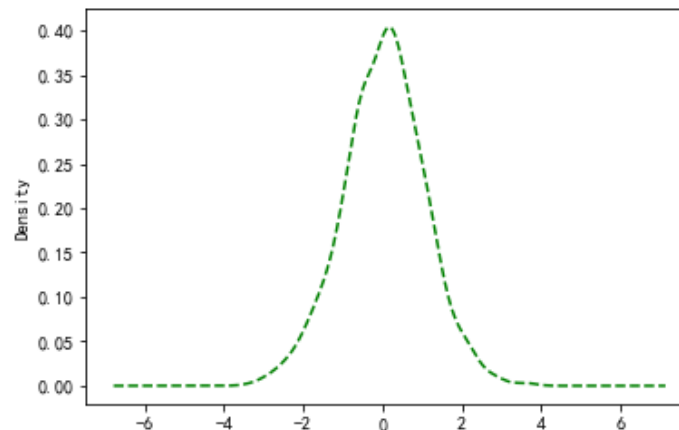
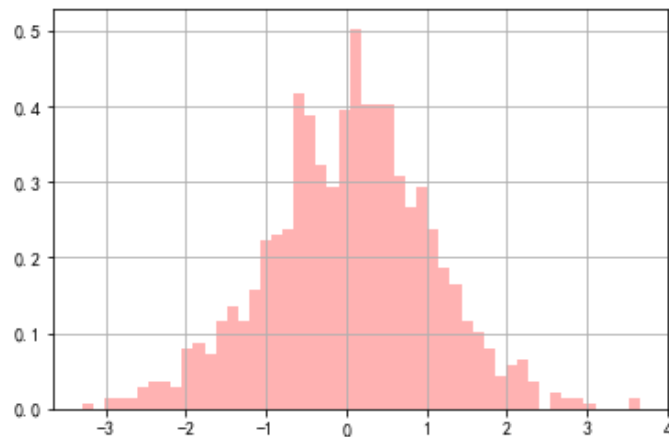
绘制直方图-正态分布

```
ds=pd.Series(np.random.normal(0,1,1000))
```

```
ds.hist(bins=50,alpha=0.3,color='r',normed=True)
```

```
ds.plot(kind='kde',style='g--')
```

```
ds.hist(bins=50,alpha=0.3,color='r',normed=True)  
ds.plot(kind='kde',style='g--')
```





实例 plt 库的基础图表

绘制散点图

#绘制散点图

```
x=np. random. rand(30)
```

```
y=np. random. rand(30)
```

```
plt. scatter(x, y)
```

```
plt. title("散点图", fontsize=18)
```

```
plt. xlabel("x坐标", fontsize=14)
```

```
plt. ylabel("y坐标", fontsize=14)
```

#绘制散点图

```
x=np. random. rand(30)
```

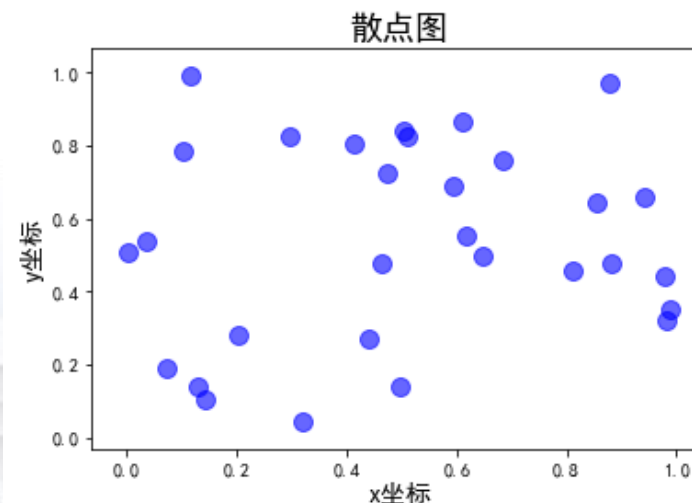
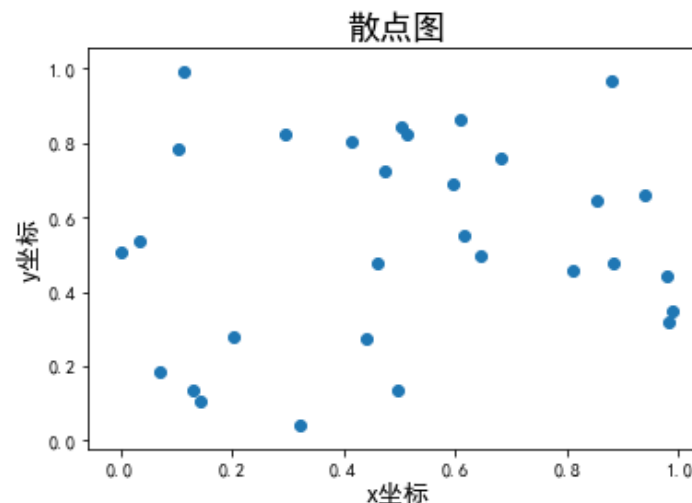
```
y=np. random. rand(30)
```

```
plt. scatter(x, y, s=100, c=' b', marker=' o', alpha=0. 6)
```

```
plt. title("散点图", fontsize=18)
```

```
plt. xlabel("x坐标", fontsize=14)
```

```
plt. ylabel("y坐标", fontsize=14)
```



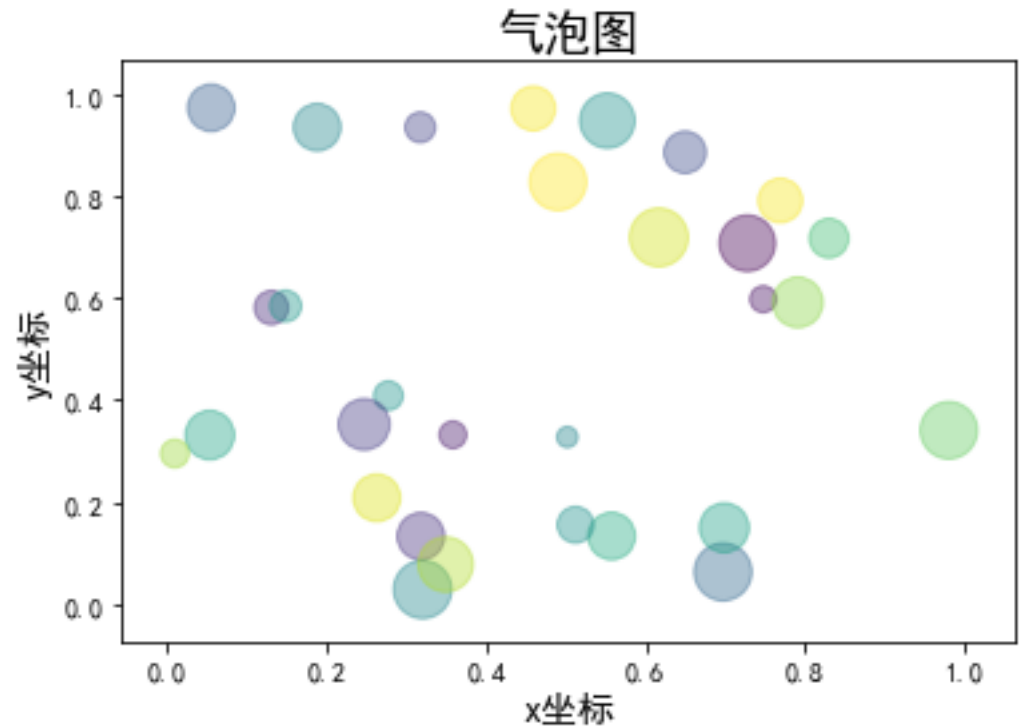


实例 plt 库的基础图表



绘制气泡图

```
x=np.random.rand(30)
y=np.random.rand(30)
s1=np.random.uniform(60, 500, 30)
c1=np.random.uniform(1, 10, 30)
plt.scatter(x, y, s=s1, c=c1, marker='o', alpha=0.4)
plt.title("气泡图", fontsize=18)
plt.xlabel("x坐标", fontsize=14)
plt.ylabel("y坐标", fontsize=14)
```



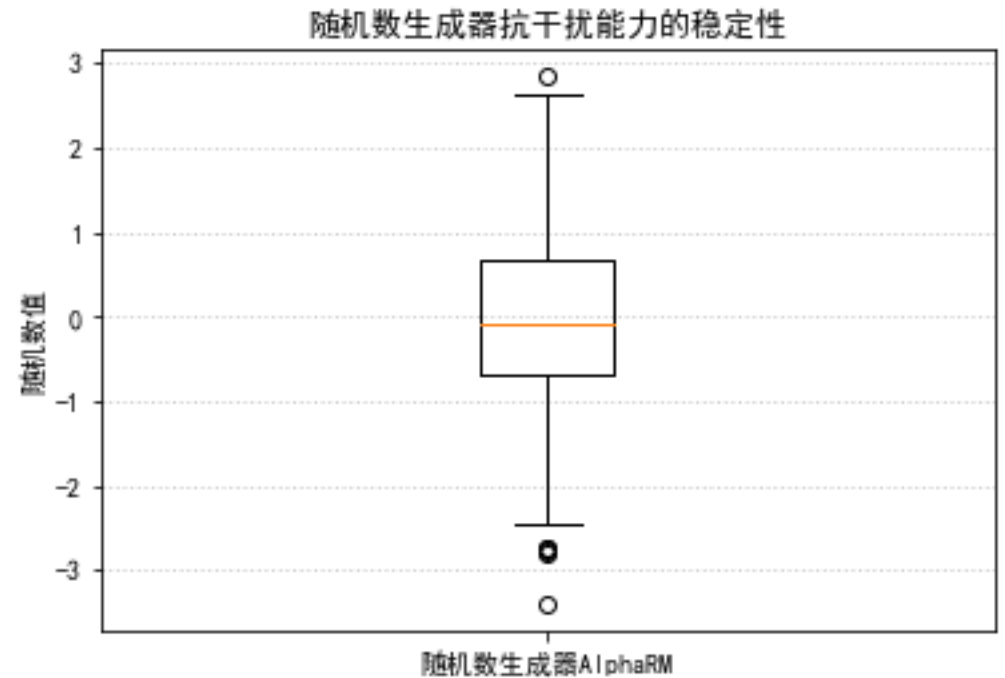


实例 plt 库的基础图表



绘制箱式图

```
x=np.random.randn(1000)
plt.boxplot(x)
plt.xticks([1], ["随机数生成器AlphaRM"])
plt.ylabel("随机数值")
plt.title("随机数生成器抗干扰能力的稳定性")
plt.grid(axis="y", ls=":", lw=1, color="gray",
alpha=0.4)
plt.show()
```

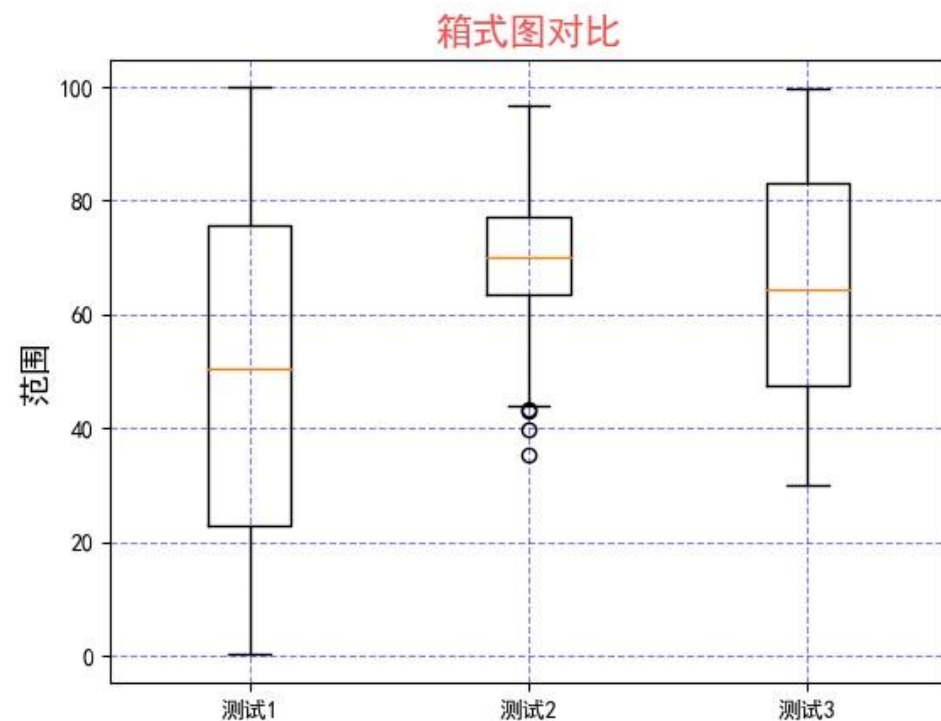




实例 plt 库的基础图表

绘制箱式图

```
ar1=np.random.rand(1000)*100
ar2=np.random.normal(70,10,1000)
ar3=np.random.uniform(30,100,100)
plt.boxplot([ar1,ar2,ar3])
plt.title("箱式图对比",fontsize=16,color='r',alpha=0.7)
plt.ylabel("范围",fontsize=14)
plt.grid(linestyle='--',color='b',alpha=0.5)
plt.xticks([1,2,3],['测试1','测试2','测试3'])
plt.show()
```





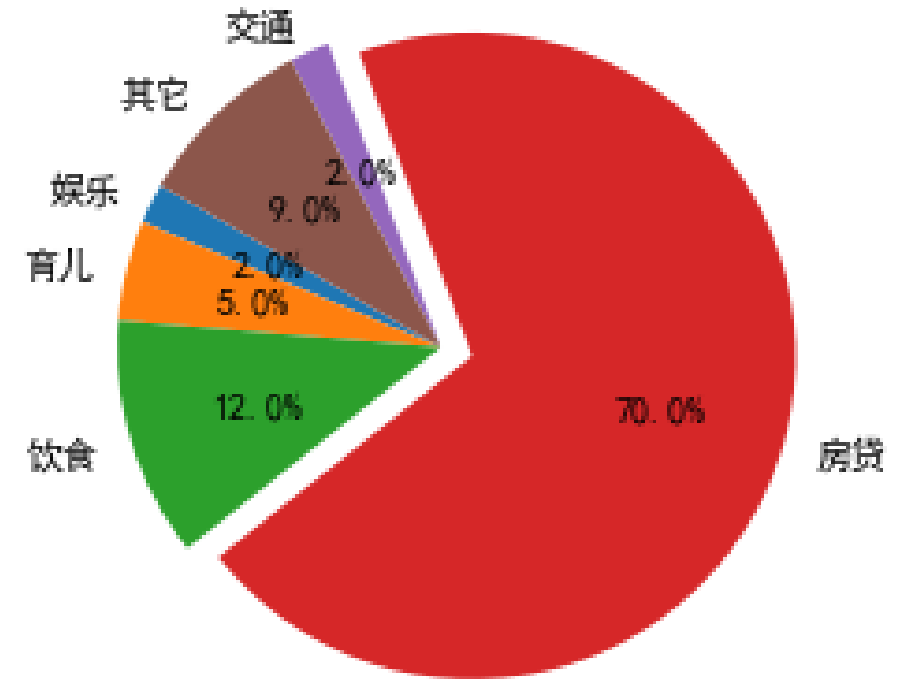
实例 plt 库的基础图表



绘制饼图

```
labels = ['娱乐', '育儿', '饮食', '房贷', '交通', '其它']  
sizes = [2, 5, 12, 70, 2, 9]  
explode = (0, 0, 0, 0.1, 0, 0)  
plt.pie(sizes, explode=explode, labels=labels,  
        autopct='%.1f%%', shadow=False, startangle=150)  
plt.title("饼图示例-8月份家庭支出")  
plt.show()
```

饼图示例-8月份家庭支出

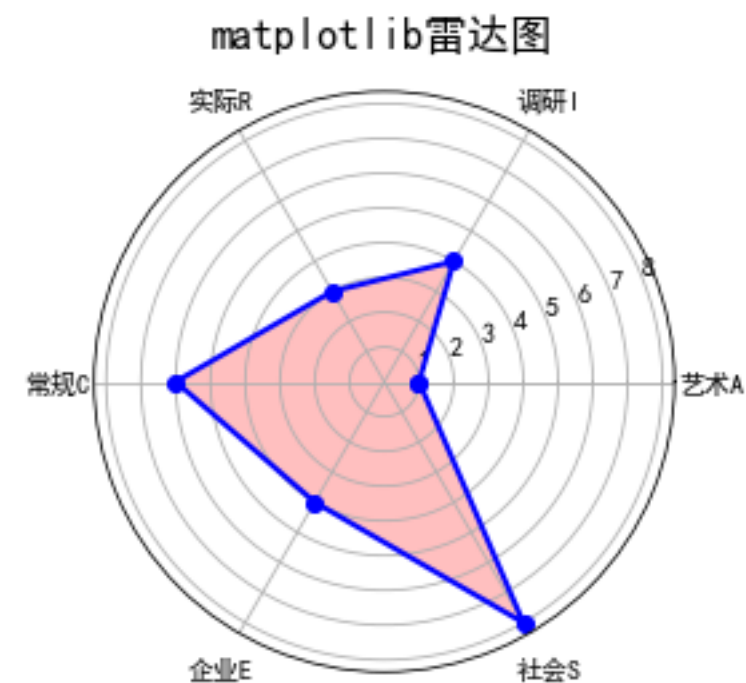




实例 plt 库的基础图表

绘制雷达图

```
labels = np.array(['艺术A', '调研I', '实际R', '常规C',  
                  '企业E', '社会S']) #数据个数  
dataLenth = 6 #数据  
data = np.array([1, 4, 3, 6, 4, 8])  
  
angles = np.linspace(0, 2*np.pi, dataLenth, endpoint=False)  
data = np.concatenate((data, [data[0]])) # 闭合  
angles = np.concatenate((angles, [angles[0]])) # 闭合  
plt.polar(angles, data, 'bo-', linewidth=2) #绘制雷达图  
plt.thetagrids(angles*180/np.pi, labels) #设置角度网络标签  
plt.fill(angles, data, facecolor='r', alpha=0.25) #填充内容颜色  
plt.title("matplotlib雷达图", fontsize=16) #设置图标题  
plt.grid(True)  
plt.show()
```



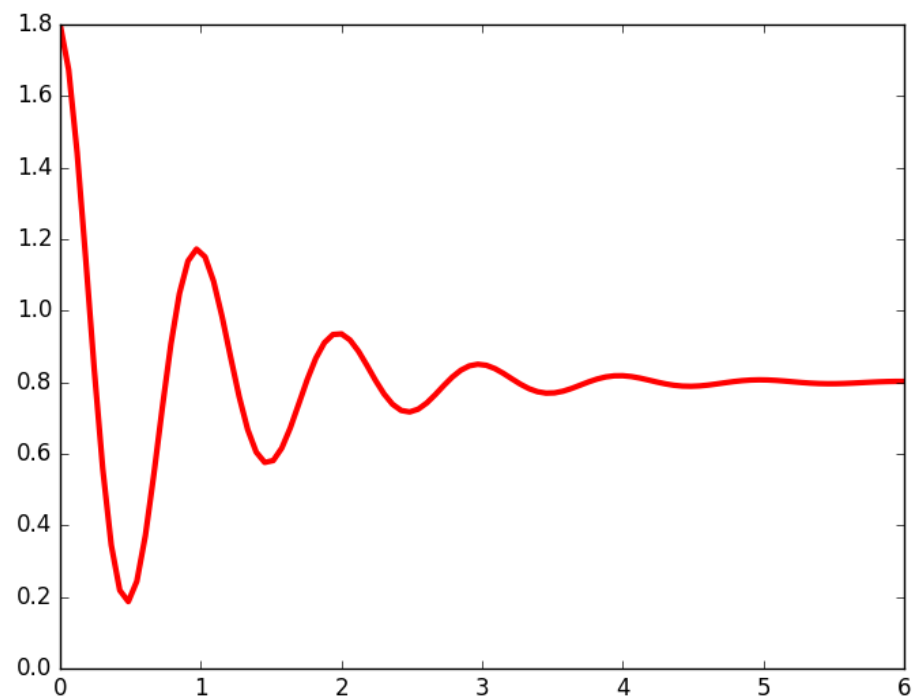


实例 绘制基本的三角函数



plt绘制基本的三角函数

```
1 import numpy as np
2 import matplotlib.pyplot as plt
3 x = np.linspace(0, 6, 100)
4 y = np.cos(2 * np.pi * x) * np.exp(-x)+0.8
5 plt.plot(x, y, 'k', color='r', linewidth=3, linestyle="-")
6 plt.show()
```





matplotlib 库的使用

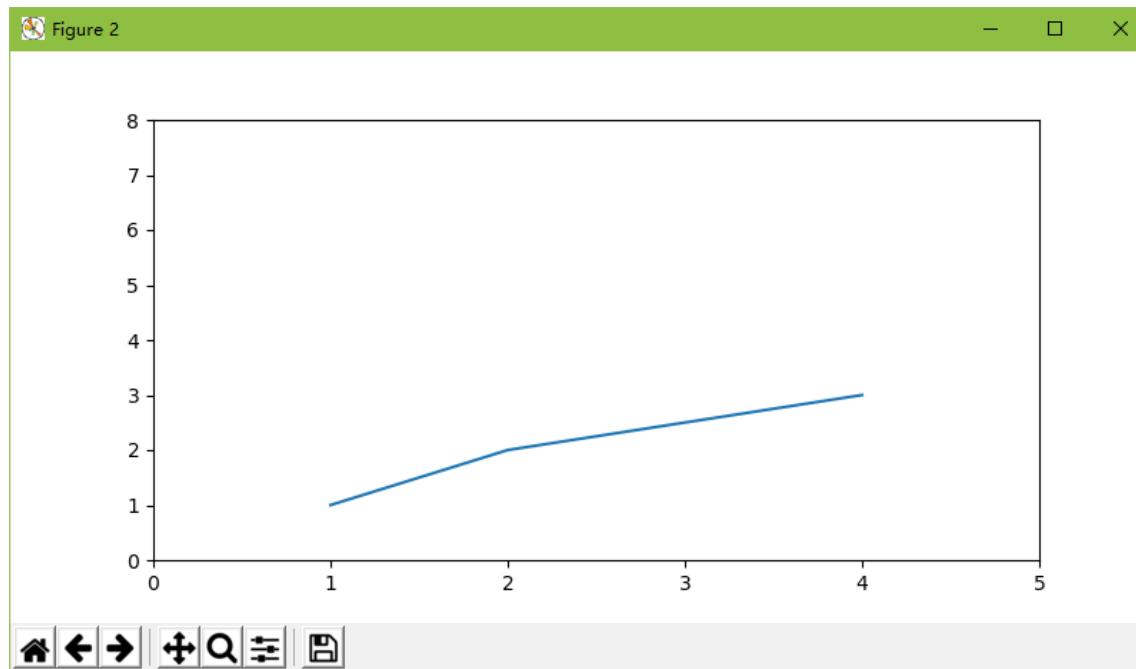
plt 库的坐标轴设置函数

```
>>> plt.plot([1, 2, 4], [1, 2, 3])
```

```
>>> plt.axis()      #获得当前坐标轴范围
```

```
(1.0, 4.0, 1.0, 3.0)
```

```
>>> plt.axis([0, 5, 0, 8])  #4个变量分别是[xmin, xmax, ymin, ymax]
```





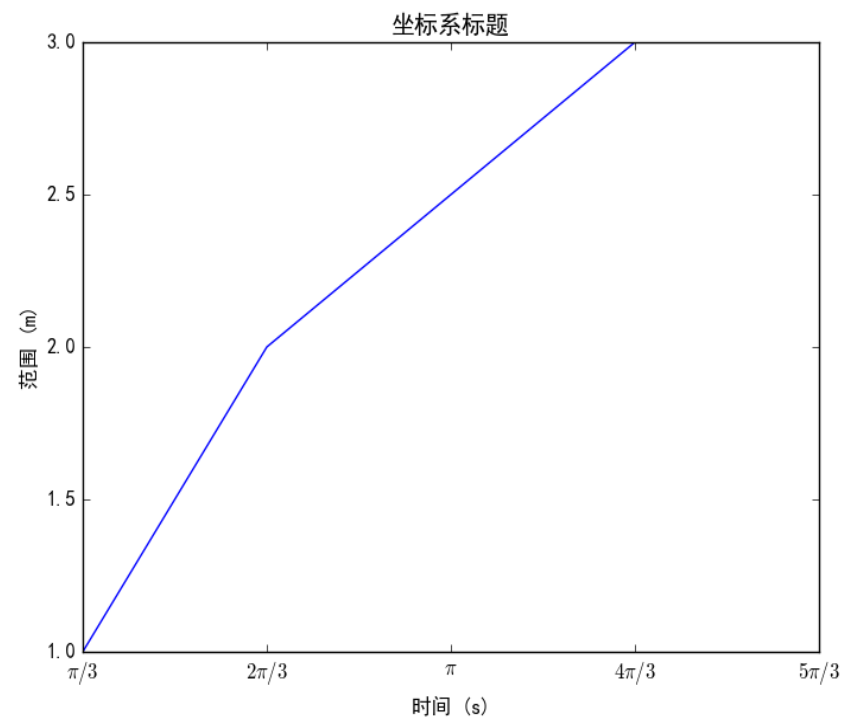
微实例 带标签的坐标系

plt库的坐标轴设置函数

微实例 10.2

m10.2PlotCoordinate.py

```
1 import matplotlib.pyplot as plt
2 import matplotlib
3 matplotlib.rcParams['font.family']='SimHei'
4 matplotlib.rcParams['font.sans-serif'] = ['SimHei']
5 plt.plot([1,2,4], [1,2,3])
6 plt.title("坐标系标题")
7 plt.xlabel('时间 (s)')
8 plt.ylabel('范围 (m)')
9 plt.xticks([1,2,3,4,5],[r'$\pi/3$', r'$2\pi/3$', r'$\pi$', \
10               r'$4\pi/3$', r'$5\pi/3$'])
11 plt.show()
```





matplotlib库的使用

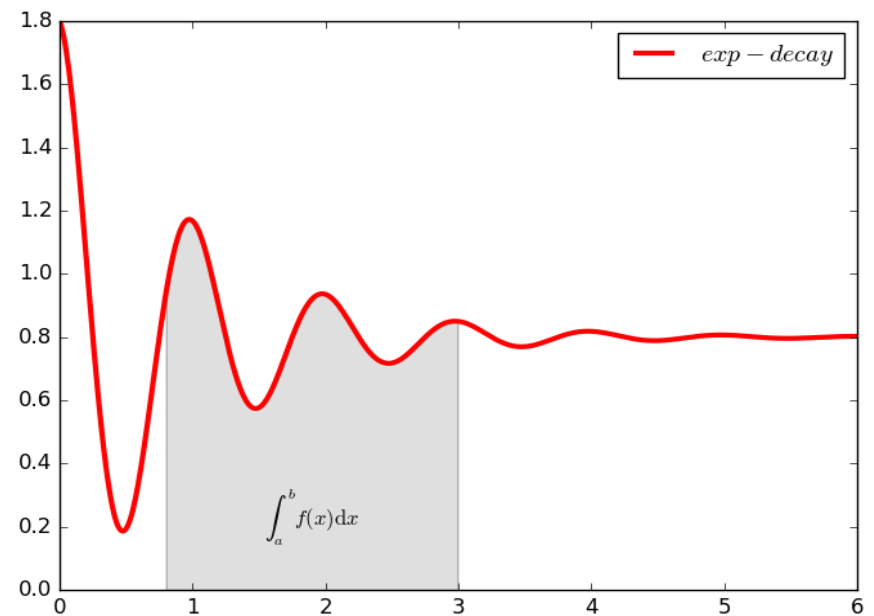
plt库的区域填充函数

微实例 10.3

m10.3PlotDarkCoordinate.py

```
1 import matplotlib.pyplot as plt
2 import numpy as np
3 x = np.linspace(0, 10, 1000)
4 y = np.cos(2*np.pi*x) * np.exp(-x)+0.8
5 plt.plot(x,y,'k',color='r',label="$exp-decay$",linewidth=3)
6 plt.axis([0,6,0,1.8])
7 ix = (x>0.8) & (x<3)
8 plt.fill_between(x, y ,0, where = ix,
9                  facecolor='grey', alpha=0.25)
10 plt.text(0.5*(0.8+3), 0.2, r"$\int_a^b f(x)\mathrm{d}x$",
11          horizontalalignment='center')
```

函数	描述
<code>fill(x,y,c,color)</code>	填充多边形
<code>fill_between(x,y1,y2,where,color)</code>	填充两条曲线围成的多边形
<code>fill_betweenx(y,x1,x2,where,hold)</code>	填充两条水平线之间的区域





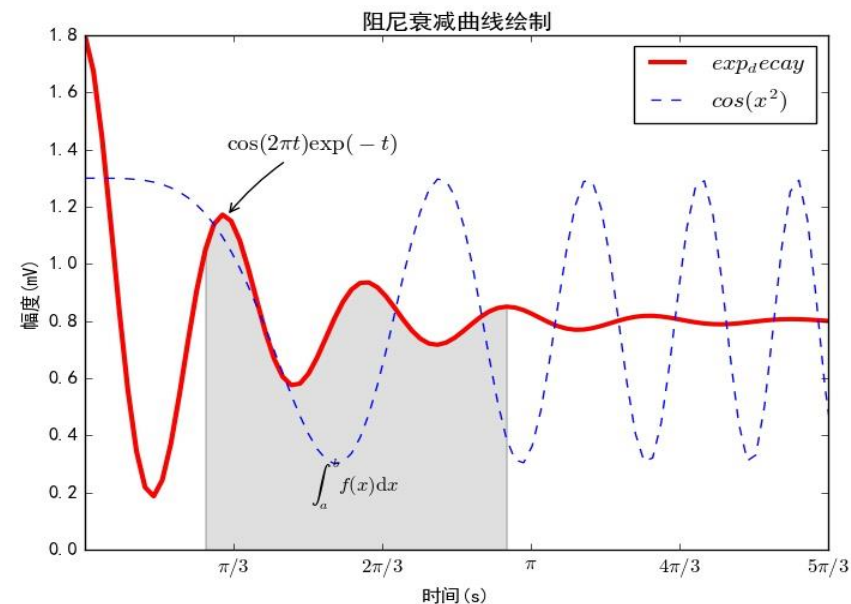
实例 科学坐标图绘制

plt库绘制科学坐标图

科学坐标图绘制有4个要素：坐标轴、数据曲线、标题和图注。这个实例以阻尼衰减曲线绘制来具体阐述科学坐标系的绘制方法。

本实例同时展示了在同一个区域用不同颜色和线条绘制两种曲线的方法，两条曲线分别为 (x, y) 和 (x, z)

```
x = np.linspace(0.0, 6.0, 100)
y = np.cos(2 * np.pi * x) * np.exp(-x) + 0.8
z = 0.5 * np.cos(x ** 2) + 0.8
```



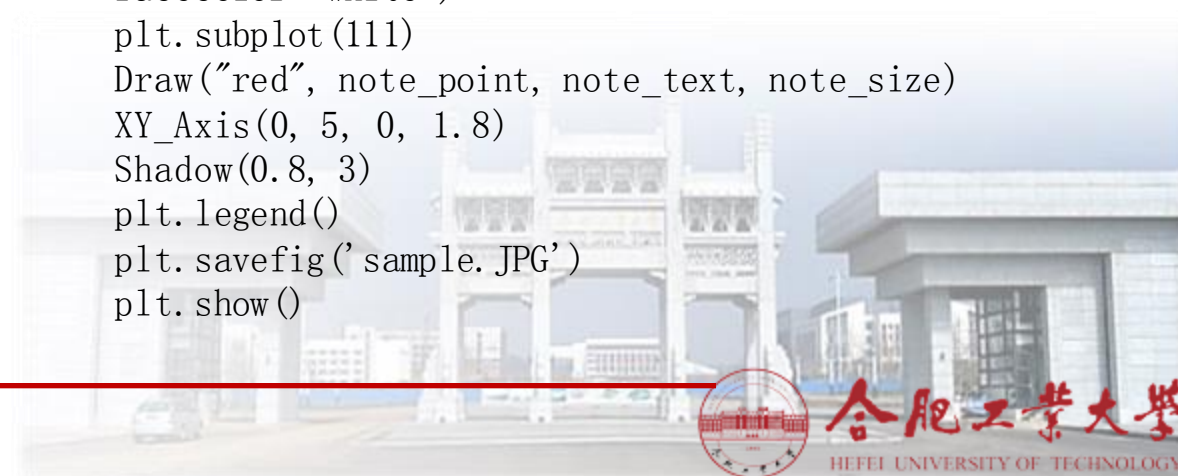


阻尼衰减曲线坐标图绘制

```
import numpy as np
import matplotlib.pyplot as plt
import matplotlib
matplotlib.rcParams['font.family']='SimHei'
matplotlib.rcParams['font.sans-serif'] = ['SimHei']
def Draw(pcolor, nt_point, nt_text, nt_size):
    plt.plot(x, y, 'k', label="$exp\_decay$", color=pcolor,
             linewidth=3, linestyle="--")
    plt.plot(x, z, "b--", label="$cos(x^2)$", linewidth=1)
    plt.xlabel('时间(s)')
    plt.ylabel('幅度(mV)')
    plt.title("阻尼衰减曲线绘制")
    plt.annotate('$\cos(2 \pi t) \exp(-t)$', xy=nt_point, \
               xytext=nt_text, fontsize=nt_size, \
               arrowprops=dict(arrowstyle='->', \
                               connectionstyle="arc3,rad=.1"))
def Shadow(a, b):
    ix = (x>a) & (x<b)
    plt.fill_between(x, y, 0, where=ix, facecolor='grey',
                    alpha=0.25)
    plt.text(0.5 * (a + b), 0.2, "$\int_a^b$")
    plt.text(0.5 * (a + b), 0.2, "f(x) \mathrm{d} x$", horizontalalignment='center')
```

```
def XY_Axis(x_start, x_end, y_start, y_end):
    plt.xlim(x_start, x_end)
    plt.ylim(y_start, y_end)
    plt.xticks([np.pi/3, 2 * np.pi/3, 1 * np.pi,
                4 * np.pi/3, 5 * np.pi/3], ['$\pi/3$',
                '$2\pi/3$', '$\pi$', '$4\pi/3$', '$5\pi/3$'])
```

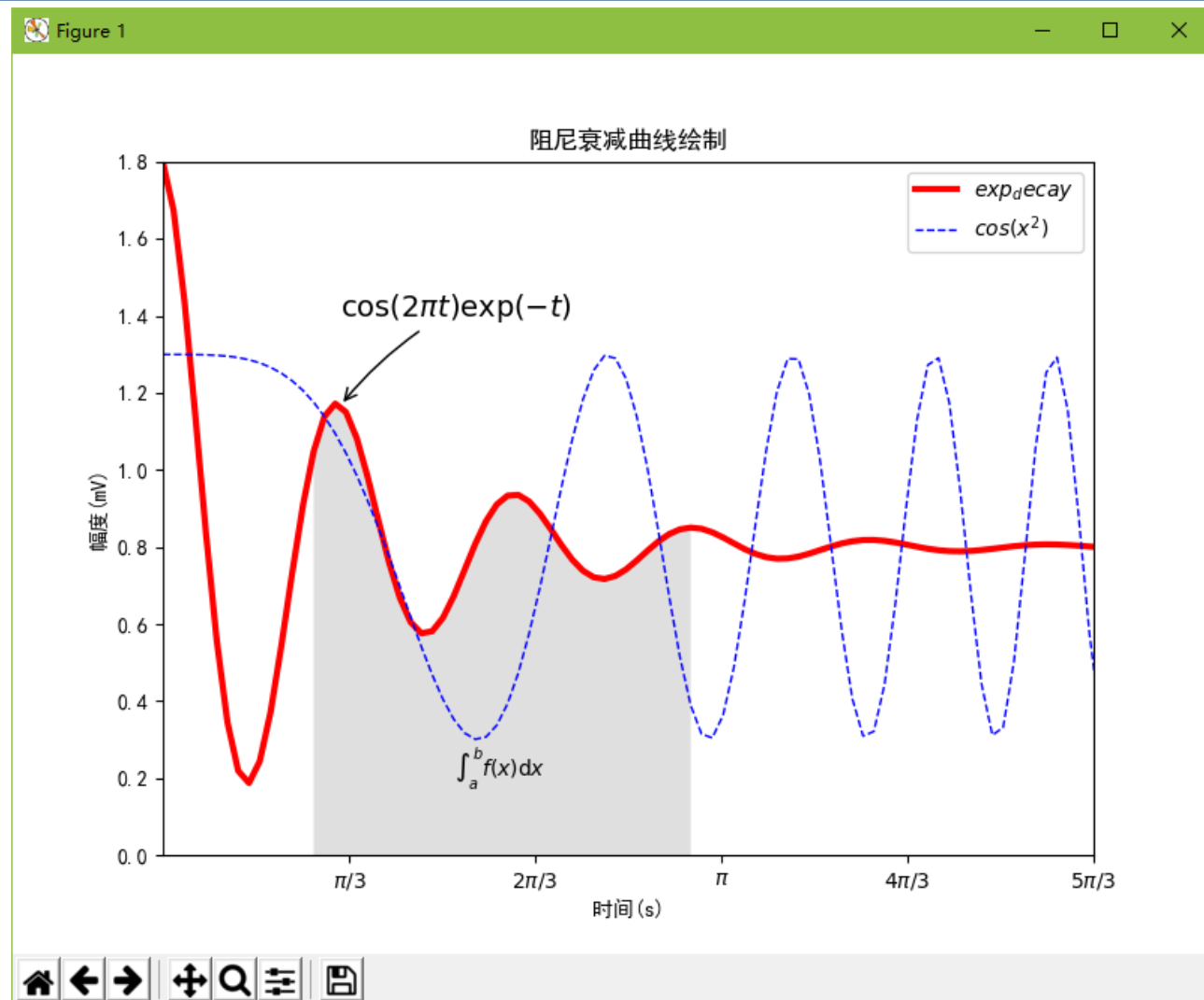
```
x = np.linspace(0.0, 6.0, 100)
y = np.cos(2 * np.pi * x) * np.exp(-x)+0.8
z = 0.5 * np.cos(x ** 2)+0.8
note_point, note_text, note_size = (1, np.cos(2 *
np.pi) * np.exp(-1)+0.8), (1, 1.4), 14
fig = plt.figure(figsize=(8, 6),
                    facecolor="white")
plt.subplot(111)
Draw("red", note_point, note_text, note_size)
XY_Axis(0, 5, 0, 1.8)
Shadow(0.8, 3)
plt.legend()
plt.savefig('sample.JPG')
plt.show()
```





阻尼衰减曲线坐标图绘制

为了便于阅读，程序设计了 Draw()、Shadow() 和 XY_Axis() 函数，分别用于绘制曲线、设置阴影和修改坐标轴。第13 行 annotate() 函数配合箭头在曲线绘图界面添加动态注释，箭头的线条和尖端都有多种样式和参数，可以通过 arrowprops 和 connectionstyle 选择，具体请参考官方文档。plt.savefig() 函数能够将产生的坐标图保存为文件。

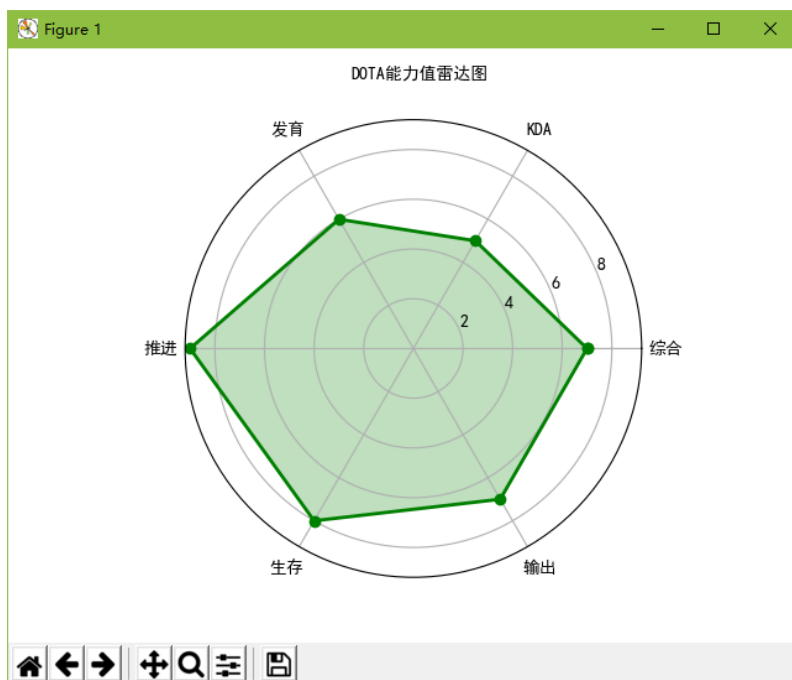




实例 多级雷达图绘制

DOTA人物能力值雷达图

本实例使用Python 来绘制这样的多级雷达图，即在一组同心圆上填充不规则六边形，其每个顶点到圆心的距离代表人物某个属性的数据。



```
import numpy as np
import matplotlib.pyplot as plt
import matplotlib
matplotlib.rcParams['font.family']='SimHei'
matplotlib.rcParams['font.sans-serif'] = ['SimHei']
labels=np.array(['综合','KDA','发育','推进','生存','输出'])
nAttr = 6
data = np.array([7, 5, 6, 9, 8, 7]) #数据值
angles = np.linspace(0, 2*np.pi, nAttr, endpoint=False)
data = np.concatenate((data, [data[0]]))
angles = np.concatenate((angles, [angles[0]]))
fig = plt.figure(facecolor="white")
plt.subplot(111, polar=True)
plt.plot(angles,data,'bo-',color='g',linewidth=2)
plt.fill(angles,data,facecolor='g',alpha=0.25)
plt.thetagrids(angles*180/np.pi, labels)
plt.figtext(0.52, 0.95, 'DOTA能力值雷达图', ha='center')
plt.grid(True)
plt.show()
```





霍兰德人格分析雷达图



霍兰德人格分析雷达图



```
import numpy as np
import matplotlib.pyplot as plt
import matplotlib
plt.rcParams['font.sans-serif'] = ['SimHei']
radar_labels = np.array(['研究型(I)', '艺术型(A)',
                          '社会型(S)', '企业型(E)',
                          '常规型(C)', '现实型(R)'])
```

```
nAttr = 6
```

```
data = np.array([[0.40, 0.32, 0.35, 0.30, 0.30, 0.88],
                 [0.85, 0.35, 0.30, 0.40, 0.40, 0.30],
                 [0.43, 0.89, 0.30, 0.28, 0.22, 0.30],
                 [0.30, 0.25, 0.48, 0.85, 0.45, 0.40],
                 [0.20, 0.38, 0.87, 0.45, 0.32, 0.28],
                 [0.34, 0.31, 0.38, 0.40, 0.92, 0.28]]) #数据值
```





霍兰德人格分析雷达图



霍兰德人格分析雷达图

```
data_labels = ('工程师', '实验员', '艺术家', '推销员', '社会工作者', '记者')
angles = np.linspace(0, 2*np.pi, nAttr, endpoint=False)
data = np.concatenate((data, [data[0]]))
angles = np.concatenate((angles, [angles[0]]))
fig = plt.figure(facecolor="white")
plt.subplot(111, polar=True)
plt.plot(angles, data, 'bo-', color='gray', linewidth=1, alpha=0.2)
plt.plot(angles, data, 'o-', linewidth=1.5, alpha=0.2)
plt.fill(angles, data, alpha=0.25)
plt.thetagrids(angles*180/np.pi, radar_labels)
plt.figtext(0.52, 0.95, '霍兰德人格分析', ha='center', size=20)
legend = plt.legend(data_labels, loc=(0.94, 0.80), labelspacing=0.1)
plt.setp(legend.get_texts(), fontsize='small')
plt.grid(True)
plt.show()
```





祝学习进步!