1 Python数据可视化基础

1.1 Python中的数据可视化工具

1.2 matplotlib的基本绘图框架

绘图命令的基本框架

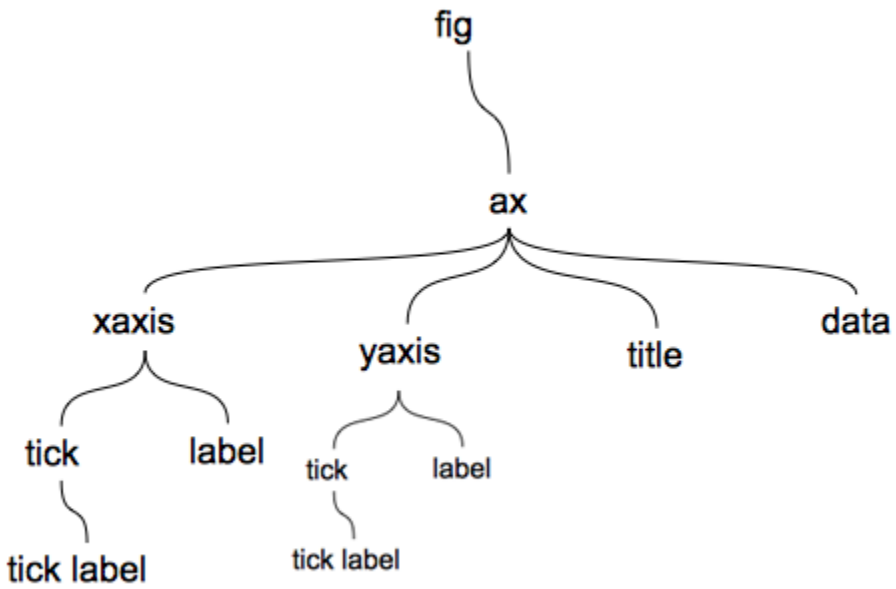
In [ ]:

```
# 最基本的绘图程序框架
from matplotlib import pyplot as plt

plt.plot([1,2,2,1])
plt.show()
```

In [ ]:

```
# 让图形在notebook中自动显示
% matplotlib inline

plt.plot([1,2,2,1])
```

图形对象的基本框架



1.2.1 创建Figure对象

Figure对象是后续绘图操作的总容器。

如果不创建Figure对象，直接调用接下来的plot()进行绘图，matplotlib会自动创建一个Figure对象。

但是事先创建Figure对象则可以对它做更精细的设定。

matplotlib.pyplot.figure(

```
num : 图形序号（ID），不指定时自动递增。如果该图形已存在，则激活相应图形
figsize : 图形的长宽，英寸，tuple
    缺省为matplotlib.rcParams["figure.figsize"]
dpi : 图形分辨率，即每英寸所表示的像素数，缺省为rc figure.dpi
    保存图像时的dpi参数为matplotlib.rcParams["savefig.dpi"]
facecolor : 图形背景色，缺省为rc figure.facecolor
edgecolor : 图形边框色，缺省为rc figure.edgecolor
frameon = True : 是否绘制图形外框架
FigureClass : 使用自定义的matplotlib.figure.Figure类
clear = False : 图形已存在时是否清除原有对象
```

)

In [ ]:

```
fig1 = plt.figure(figsize = (10, 5))
plt.plot([1,2,2,1])
```

In [ ]:

```
# 图形被显示出来之后，再次绘图又会生成一个新的Figure
plt.plot([1,2,2,1])
```

In [ ]:

```
# 使用Figure的名称调用所需的图形
fig1
```

1.2.2 调用plot()在Figure对象中绘图

实际上plot()是在Axes（子图）对象上绘图，如果当前Figure对象中没有Axes对象，则将会创建一个几乎充满整个图表的Axes对象，并且使此Axes对象成为当前Axes对象。

一个Figure对象中可以有多个Axes对象。

matplotlib.pyplot.plot(

```
x, y : X、Y轴所对应的数据，均为NumPy数组
    x轴对应数据为可选，缺省为[0, ..., N-1]
data : 可选，变量列所对应的数据框名称

fmt : 用'[color][marker][line]'格式的字符串参数进行图像格式设定
```

) 返回：Line2D objects

In [ ]:

```
plt.plot([1,2,2,1])
```

In [ ]:

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

In [ ]:

```
# 在同一个Axes对象中叠加图形
plt.plot([1,2,2,1])
plt.plot([4,2,3,1], [1,2,2,1])
```

In [ ]:

```
# 来点复杂的图形
import numpy as np
import matplotlib.pyplot as plt

t = np.linspace(-3, 3, 100)

x = 16*(np.sin(t))**3
y = 13*np.cos(t)-5*np.cos(2*t)-2*np.cos(3*t)-np.cos(4*t)

a = plt.figure(figsize = (5, 5))
plt.plot(x, y)
```

常用图形对应的绘图命令

绘图命令	具体图形
matplotlib.pyplot.angle_spectrum	绘制电子波谱图
matplotlib.pyplot.bar	绘制柱状图
matplotlib.pyplot.barh	绘制直方图
matplotlib.pyplot.broken_barh	绘制水平直方图
matplotlib.pyplot.contour	绘制等高线图
matplotlib.pyplot.errorbar	绘制误差线
matplotlib.pyplot.hexbin	绘制六边形图案
matplotlib.pyplot.hist	绘制柱形图
matplotlib.pyplot.hist2d	绘制水平柱状图
matplotlib.pyplot.imshow	以图像显示
matplotlib.pyplot.pie	绘制饼状图
matplotlib.pyplot.quiver	绘制量场图
matplotlib.pyplot.scatter	散点图
matplotlib.pyplot.specgram	绘制光谱图
matplotlib.pyplot.subplot	绘制子图

1.2.3 对图形元素做格式设定

用fmt = '[color][marker][line]'参数做格式设定

fmt参数通过一些易记的符号指定曲线的样式。例如'b'表示蓝色，'–'表示线型为虚线。在IPython中输入"plt.plot?"可以查看格式化字符串以及各个参数的详细说明。

颜色字符	颜色描述	形状字符	形状描述	连线字符	连线描述
'b'	blue	'.'	point marker	'-'	solid line style
'g'	green	','	pixel marker	'--'	dashed line style
'r'	red	'o'	circle marker	'-.'	dash-dot line style
'c'	cyan	'v'	triangle_down marker	':'	dotted line style
'm'	magenta	'^'	triangle_up marker		
'y'	yellow	'<'	triangle_left marker		
'k'	black	'>'	triangle_right marker		
'w'	white	'1'	tri_down marker		
		'2'	tri_up marker		
		'3'	tri_left marker		
		'4'	tri_right marker		
		's'	square marker		
		'p'	pentagon marker		
		'*'	star marker		
		'h'	hexagon1 marker		
		'H'	hexagon2 marker		
		'+'	plus marker		
		'x'	x marker		
		'D'	diamond marker		
		'd'	thin*diamond marker*		
		'|'	vline marker		
		''	hline marker		

In [ ]:

```
plt.plot([4,2,3,1], [1,2,2,1], 'ro--')
```

使用其余参数进行格式设定

label：给曲线指定一个标签名称，此标签将在图示中显示。
　　如果标签字符串的前后有字符'$'，则会使用其内嵌的LaTex引擎将其显示为数学公式。
　　使用LaTex语法绘制数学公式会极大地降低图表的绘制速度。
color：指定曲线的颜色，颜色可以用英文单词。
　　或者以'#'字符开头的三个16进制数，例如'#ff0000'表示红色。
　　或者使用0到1范围之内的三个元素的元组表示，例如(1.0, 0.0, 0.0)也表示红色。
linewidth：指定曲线的宽度，可以不是整数，也可以使用缩写形式的参数名lw。
继承自matplotlib.lines.Line2D的各种属性。

In [ ]:

```
plt.plot([4,2,3,1], [1,2,2,1], 'ro--', linewidth = 3.5)
```

1.2.4 设定当前Axes对象的属性

xlim/ylim ： 分别设置X、Y轴的显示刻度范围。
xlabel/ylabel ： 分别设置X、Y轴的标题文字。
clabel ： 为轮廓线设置标签文字。
title ： 设置子图的标题。
legend ： 显示图示，即图中表示每条曲线的标签(label)和样式的矩形区域。
annotate ： 绘制图形标注。
axhspan ： 绘制垂直或水平色块。
fill ： 填充区域。

In [ ]:

```
plt.plot([4,2,3,1], [1,2,2,1], 'ro--', linewidth = 3.5)
plt.annotate("This is a point",  xy=(3, 2))
plt.xlabel('$X\ value$')
```

1.2.5 调用plt.show()显示图形

让图形在notebook中单独窗口显示

默认情况下show()将会阻塞程序的运行，直到用户关闭绘图窗口。

In [ ]:

```
% matplotlib qt5

plt.plot([1,2,2,1])
```

In [ ]:

```
# 让图形在notebook中自动显示
% matplotlib inline
```

保存图像文件

也可以调用plt.savefig()将当前的Figure对象保存成图像文件，图像格式由图像文件的扩展名决定。

In [ ]:

```
plt.savefig("test.png")
```

In [ ]:

```
fig1 = plt.figure(figsize = (10, 5))
plt.plot([1,2,2,1])
plt.figure()
plt.plot([1,2])

fig1.savefig("fig1.png")
```

1.3 matplotlib图形输出窗口的基本操作

1.4 准备matplotlib+seaborn绘图环境

检查相关包的版本

seaborn请务必升级至最新的0.9.0以上版本，否则经常会有警告出现，且有几个命令无法运行。

```
pip install --upgrade seaborn
```

升级过程中会同步升级相关的matplotlib, pandas, statsmodels等库

In [ ]:

```
import matplotlib
matplotlib.__version__ # 显示当前matplotlib版本号
```

In [ ]:

```
import seaborn
seaborn.__version__ # 显示当前seaborn版本号
```

加载相关库

In [ ]:

```
# 加载pandas库（用于数据管理）
import pandas as pd
```

In [ ]:

```
# 加载matplotlib.pyplot库
from matplotlib import pyplot as plt
```

In [ ]:

```
# 加载seaborn库
import seaborn as sns
```

格式修饰

In [ ]:

```
# 加载seaborn默认格式设定
sns.set() # matplotlib的初学者可暂缓使用该设定
```

解决中文显示问题

matplotlib环境默认无法显示中文，因此需要专门设定。

In [ ]:

```
# 黑体会出现负号无法显示的问题
plt.rcParams["font.family"] = "SimHei"
plt.plot([-1,2,2,1])
plt.xlabel("中文字体")
```

In [ ]:

```
# 华文细黑是相对而言显示效果较好的中文字体
plt.rcParams["font.family"] = "STXIHEI"
plt.plot([-1,2,2,1])
plt.xlabel("中文字体")
```

1.5 实战练习

尝试对前面绘制的心形图案进行如下编辑操作：

> 将曲线颜色改为红色，线形改为虚线。
> 在座标(0,5)处加绘反方向的蓝色小心形（对修改函数不熟悉的可以改为绘制一个放大的实心散点）。
> 在图案上叠加适当的中文文字。

自行尝试使用matplotlib绘制简单的线图，并尝试修改其格式。

2 数据可视化理论入门

2.1 统计图的基本信息维度

2.2 统计图的基本框架和格式需求

2.3 统计图的基本分类

2.3.1 单变量图

2.3.2 双变量图

2.3.3 多变量图

2.3.4 其他更复杂的图形