Python语言

04 可视化Matplotlib

广东工业大学自动化学院　邢延

内容提纲

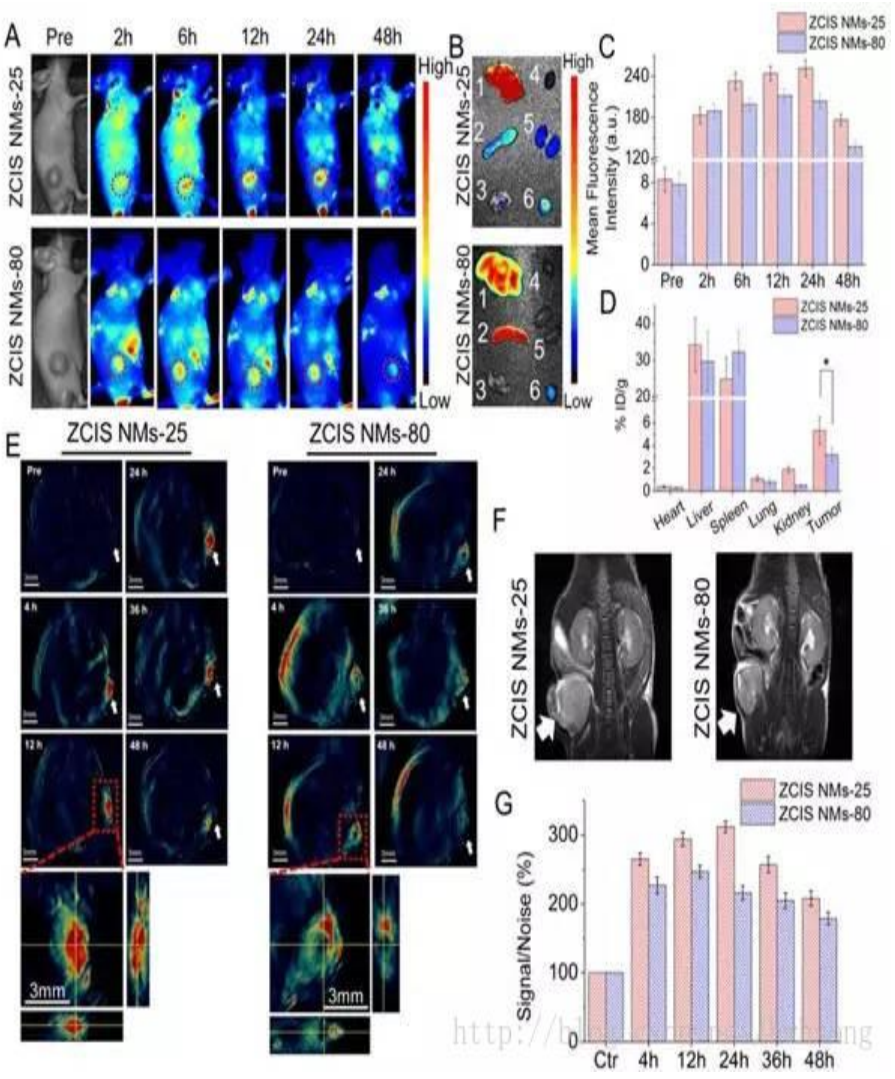

可视化
数据可视化概览
matplotlib
快速绘图
图像组成
基本类型
点线图、条形图、直方图、饼图
散点图、箱型图
复杂类型
多轴图
坐标共享轴
3D
图形属性设置
其他类库
Seaborn
Basemap（地图）
其他语言可视化工具
数据可视化使用小贴士

1. 可视化概览

- 科学可视化(Scientific Visualization)、信息可视化(Information Visualization)和可视分析学(Visual Analytics)三个学科方向通常被看成可视化的三个主要分支。而将这三个分支整合在一起形成的新学科“数据可视化”，这是可视化研究领域的新起点。——《数据可视化》

1. 可视化概览

- 科学可视化（Scientific Visualization）是科学之中的一个跨学科研究与应用领域；
- 主要关注三维现象的可视化，如建筑学、气象学、医学或生物学方面的各种系统；
- 科学可视化的目的是以图形方式说明科学数据，使科学家能够从数据中了解、说明和收集规律。



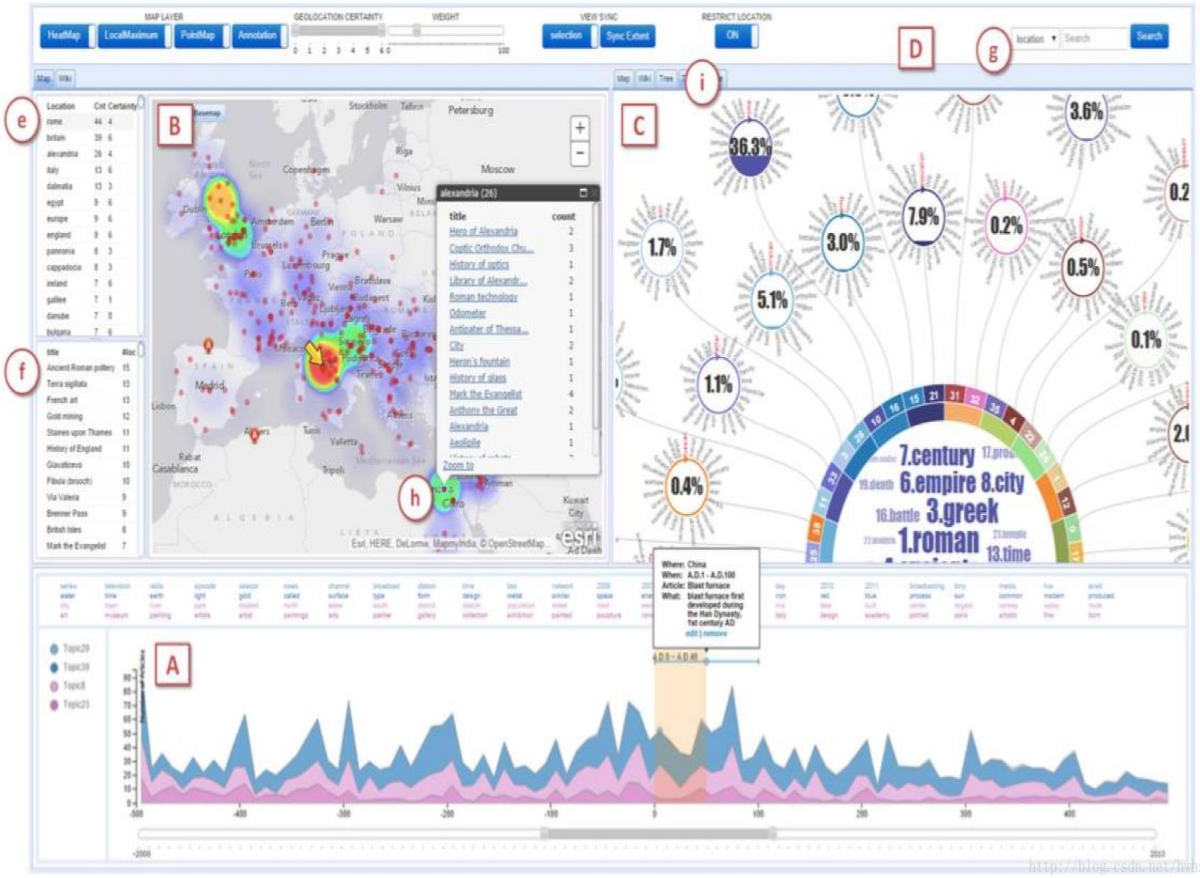
1. 可视化概览

- 信息可视化（Information Visualization）是研究抽象数据的交互式视觉表示以加强人类认知。

- 抽象数据包括数字和非数字数据，如地理信息与文本。柱状图、趋势图、流程图、树状图等都属于信息可视化，这些图形的设计都将抽象的概念转化成为可视化信息。

1. 可视化概览

- 可视分析学（Visual Analytics）是随着科学可视化和信息可视化发展而形成的新领域，重点是通过交互式视觉界面进行分析推理。



1. 可视化概览

❖ 为什么需要可视化

- **人类利用视觉获取的信息量，远远超出其他器官**
  - **人类的眼睛是一对高带宽巨量视觉信号输入的并行处理器，拥有超强模式识别能力，配合超过 50% 功能用于视觉感知相关处理的大脑，使得人类通过视觉获取数据比任何其他形式的获取方式更好，大量视觉信息在潜意识阶段就被处理完成，**
  - **人类对图像的处理速度比文本快 6 万倍。 数据可视化正是利用人类天生技能来增强数据处理和组织效率。**
- **可视化可以帮助人类处理更加复杂的信息并增强记忆**
  - **大多数人对统计数据了解甚少，基本统计方法（平均值、中位数、范围等）并不符合人类的认知天性。**

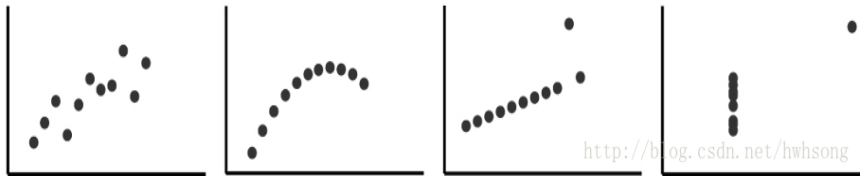
1. 可视化概览

- 一个例子是 Anscombe 的四重奏，根据统计方法看数据很难看出规律，但一可视化出来，规律就非常清楚。

a

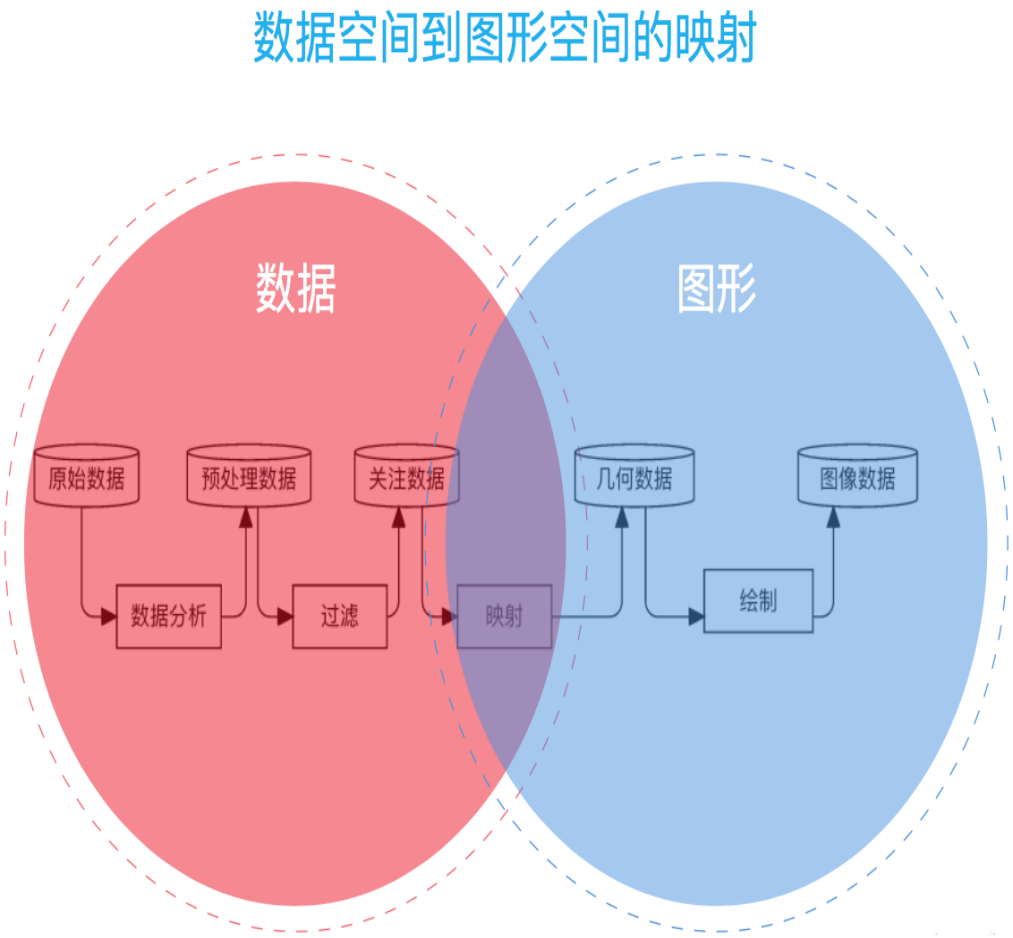
I		II		III		IV	
x	*y*	*x*	*y*	*x*	*y*	*x*	*y*
10	8.04	10	9.14	10	7.46	8	6.58
8	6.95	8	8.14	8	6.77	8	5.76
13	7.58	13	8.74	13	12.74	8	7.71
9	8.81	9	8.77	9	7.11	8	8.84
11	8.33	11	9.26	11	7.81	8	8.47
14	9.96	14	8.10	14	8.84	8	7.04
6	7.24	6	6.13	6	6.08	8	5.25
4	4.26	4	3.10	4	5.39	19	12.5
12	10.84	12	9.13	12	8.15	8	5.56
7	4.82	7	7.26	7	6.42	8	7.91
5	5.68	5	4.74	5	5.73	8	6.89

b



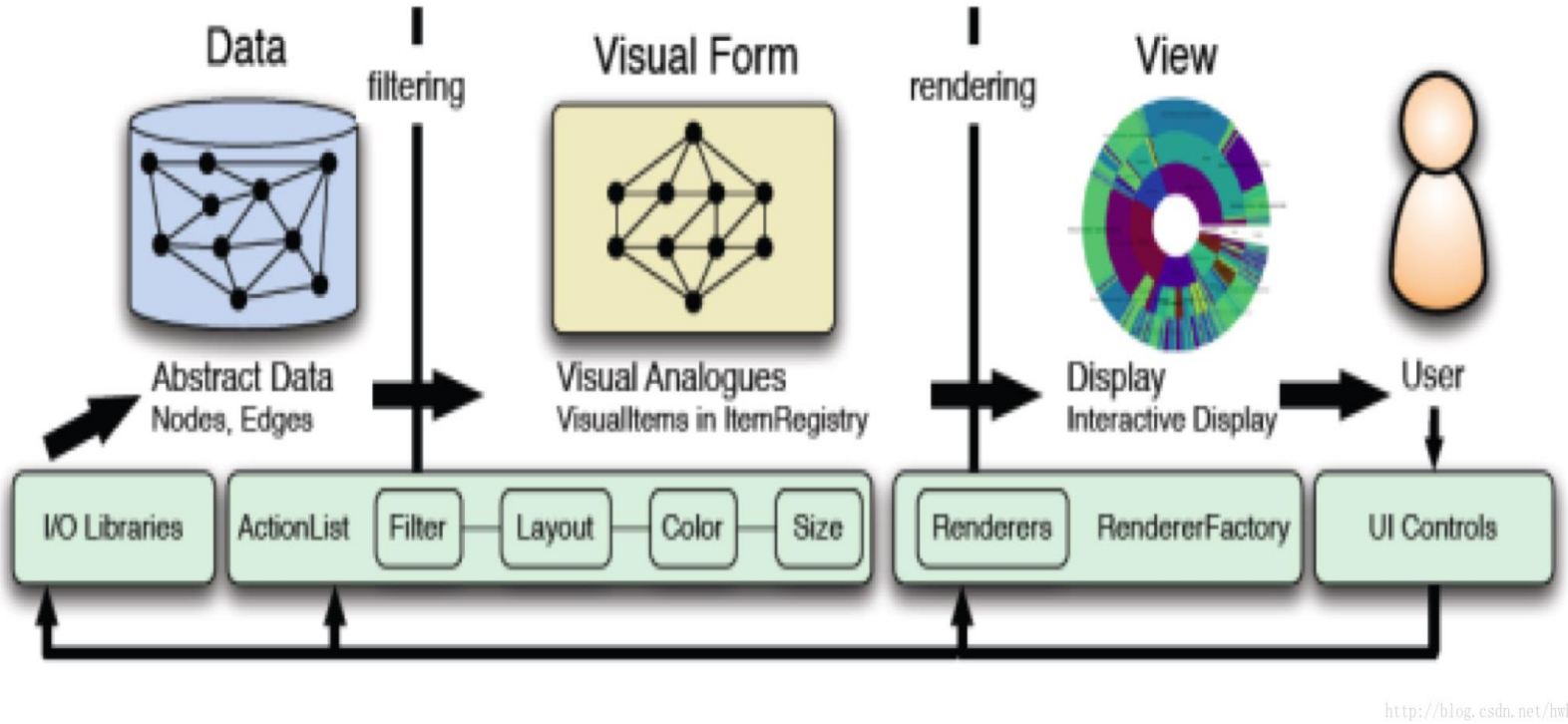
1. 可视化概览

- 数据可视化最简单的理解，就是数据空间到图形空间的映射



1. 可视化概览

- 可视化实现流程
  - 一个经典的可视化实现流程，是先对数据进行加工过滤，转变成视觉可表达的形式（Visual Form），然后再渲染成用户可见的视图（View）。

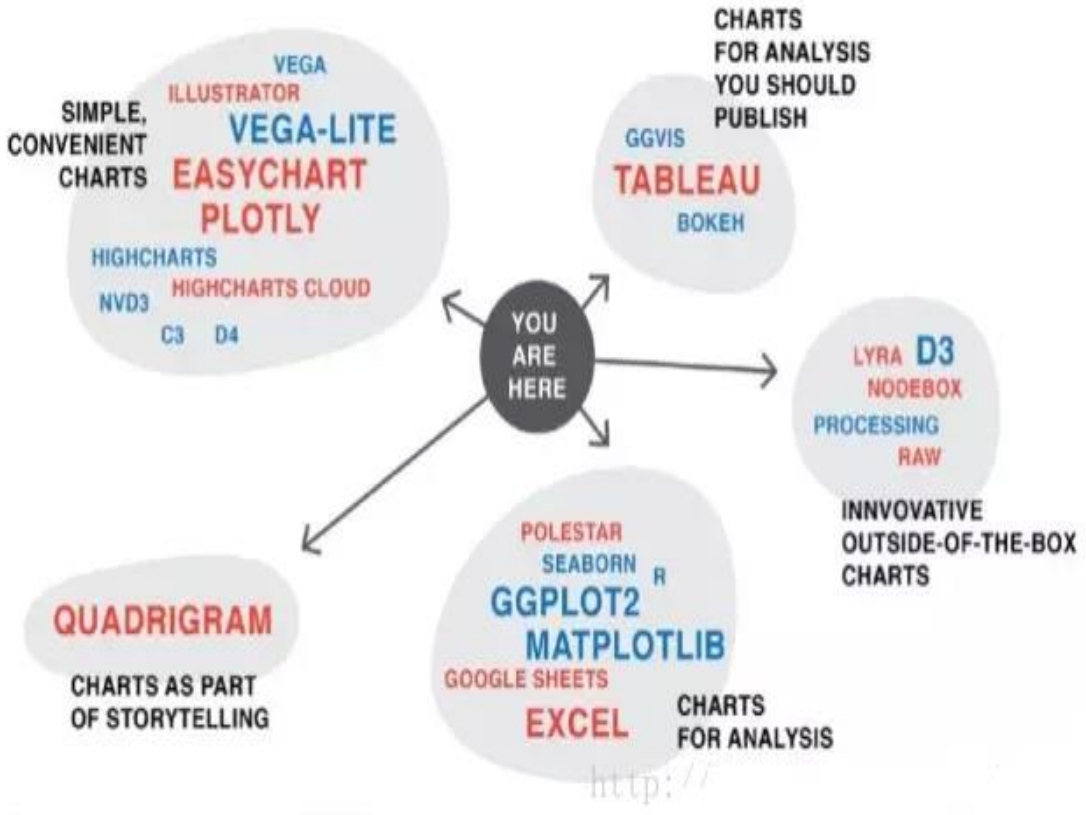


1. 可视化概览

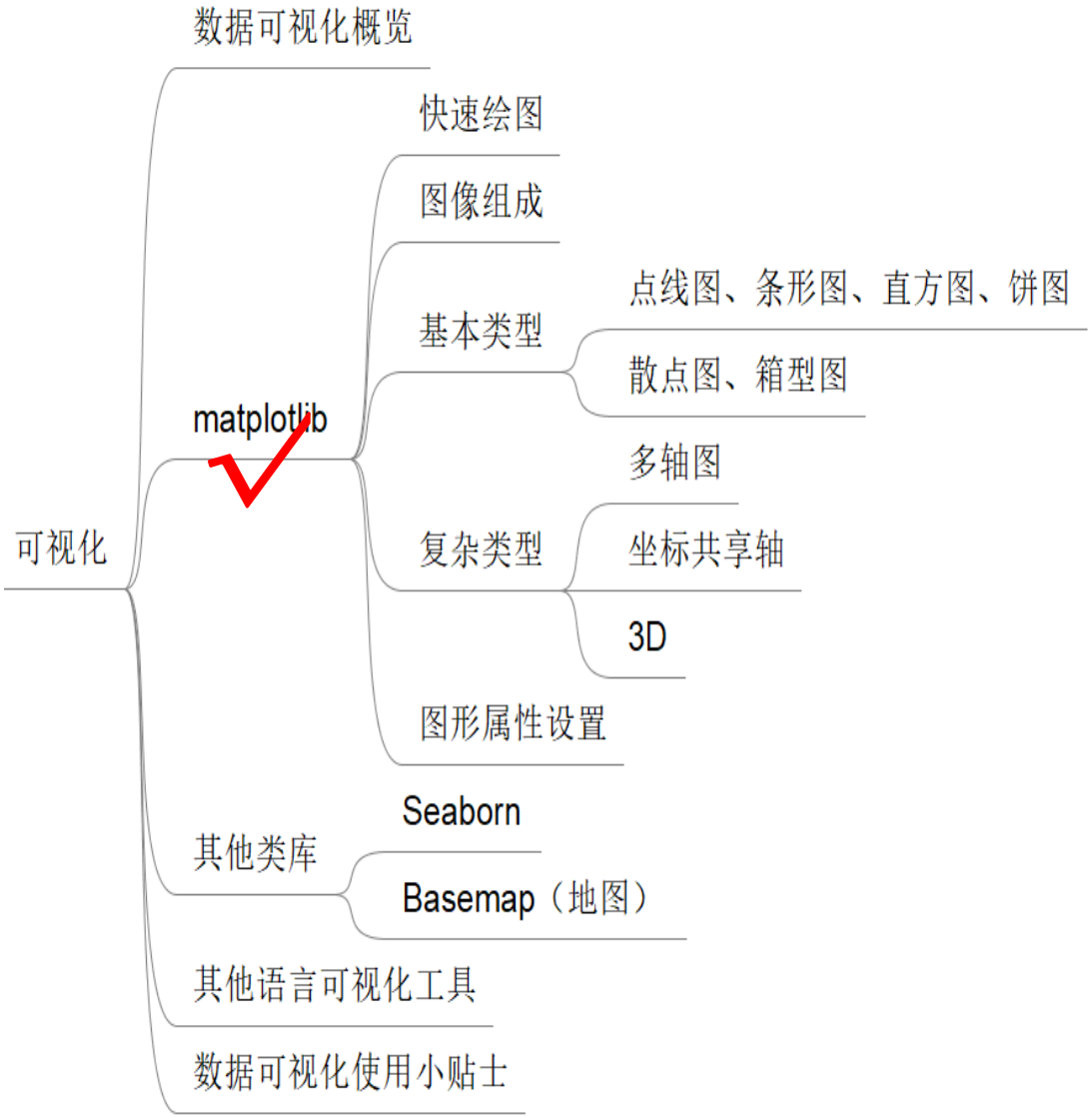
- 数据可视化大多指狭义的数据可视化以及部分信息可视化。根据数据类型和性质的差异，经常分为以下几种类型：
  - 统计数据可视化：用于对统计数据进行展示、分析。统计数据一般都是以数据库表的形式提供，常见的统计可视化类库有 HighCharts、ECharts、G2、Chart.js 等等，都是用于展示、分析统计数据。
  - 关系数据可视化：主要表现为节点和边的关系，比如流程图、网络图、UML 图、力导图等。常见的关系可视化类库有 mxGraph、JointJS、GoJS、G6 等。
  - 地理空间数据可视化：地理空间通常特指真实的人类生活空间，地理空间数据描述了一个对象在空间中的位置。在移动互联网时代，移动设备和传感器的广泛使用使得每时每刻都产生着海量的地理空间数据。常见类库如 Leaflet、Turf、Polymaps 等等，最近 Uber 开源的 deck.gl 也属于此类。
  - 时间序列数据可视化（如 timeline）、文本数据可视化（如 worldcloud）等等。

1. 可视化概览

- 数据可视化工具按目的分类，左上是简单快捷，左下是场景导向，右上是为了分享分析，右侧是创新型图表，右下是分析型工具。



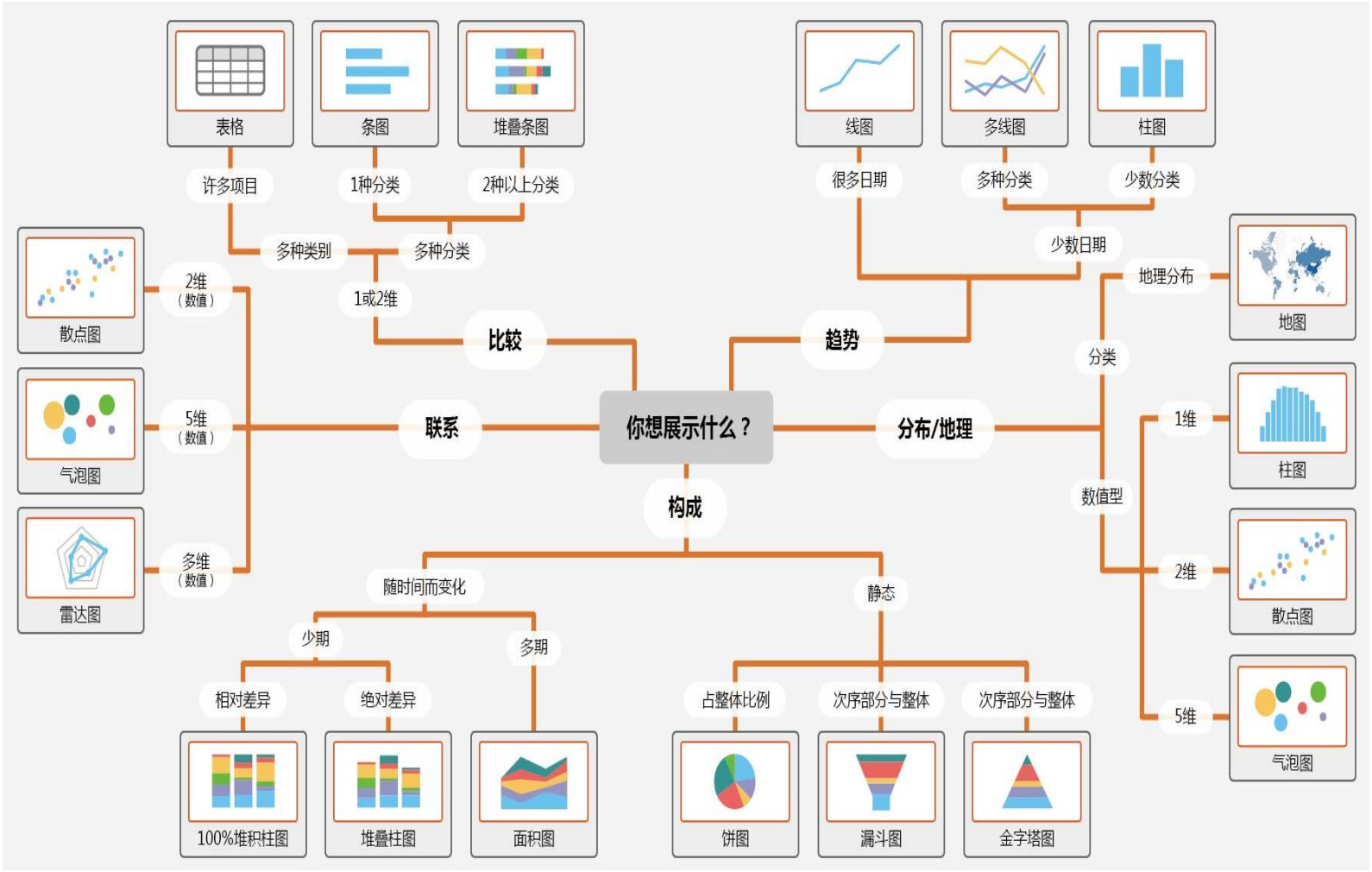
2. matplotlib



Matplotlab简介

- matplotlib 是python最著名的绘图库，它提供了一整套和matlab相似的命令API，十分适合交互式地进行制图。而且也可以方便地将它作为绘图控件，嵌入GUI应用程序中。
- 它的文档相当完备，并且Gallery页面中有上百幅缩略图，打开之后都有源程序。因此如果你需要绘制某种类型的图，只需要在这个页面中浏览/复制/粘贴一下，基本上都能搞定。
- 展示页面的地址：
  https://matplotlib.org/gallery/index.html

Matplotlab简介



2-1 快速绘图


可视化
数据可视化概览
matplotlib
快速绘图
图像组成
基本类型
点线图、条形图、直方图、饼图
散点图、箱型图
复杂类型
多轴图
坐标共享轴
3D
图形属性设置
其他类库
Seaborn
Basemap（地图）
其他语言可视化工具
数据可视化使用小贴士

2-1 快速绘图

- matplotlib的pyplot子库提供了和matlab类似的绘图API，方便用户快速绘制2D图表。
- matplotlib中的快速绘图的函数库可以通过如下语句载入：

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

- 接下来调用figure创建一个绘图对象，并且使它成为当前的绘图对象。

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

- 通过figsize参数可以指定绘图对象的宽度和高度，单位为英寸；dpi参数指定绘图对象的分辨率，即每英寸多少个像素，缺省值为80。因此本例中所创建的图表窗口的宽度为8*80 = 640像素。

2-1 快速绘图

- 下面的两行程序通过调用plot函数在当前的绘图对象中进行绘图

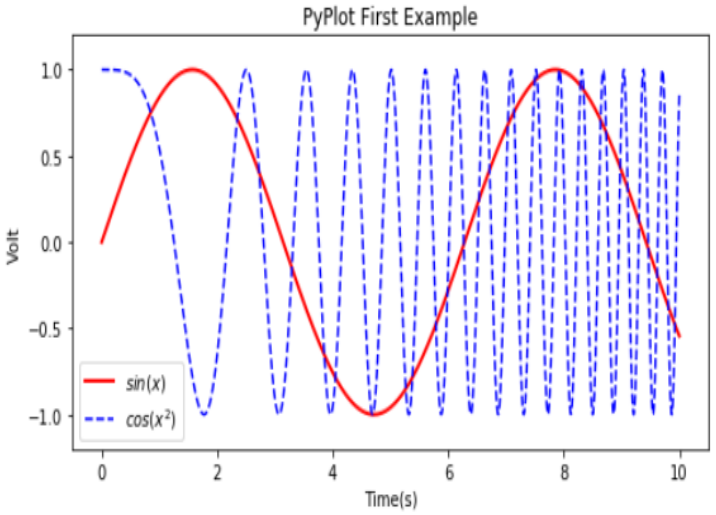
```
plt.plot(x,y,label="$sin(x)$",color="red",linewidth=2)
plt.plot(x,z,"b--",label="$cos(x^2)$")
```

- plot函数的调用方式很灵活，第一句将x,y数组传递给plot之后，用关键字参数指定各种属性：
  - label : 给所绘制的曲线一个名字，此名字在图示(legend)中显示。只要在字符串前后添加"$"符号，matplotlib就会使用其内嵌的latex引擎绘制的数学公式。
  - color : 指定曲线的颜色
  - linewidth : 指定曲线的宽度
  - "b--"指定曲线的颜色和线型

2-1 快速绘图

- 接下来通过一系列函数设置绘图对象的各个属性：

```
plt.xlabel("Time(s)")
plt.ylabel("Volt")
plt.title("PyPlot First Example")
plt.ylim(-1.2,1.2)
plt.legend()
```



- xlabel / ylabel : 设置X轴/Y轴的文字
- title : 设置图表的标题
- ylim : 设置Y轴的范围
- legend : 显示图示

- 最后调用plt.show()显示出创建的所有绘图对象。

2-1 快速绘图

- 还可以调用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。下面的程序将当前的图表保存为“test.png”，并且通过dpi参数指定图像的分辨率为 120，因此输出图像的宽度为“8X120 = 960”个像素。

```
run matplotlib_simple_plot.py
plt.savefig("test.png",dpi=120)
```

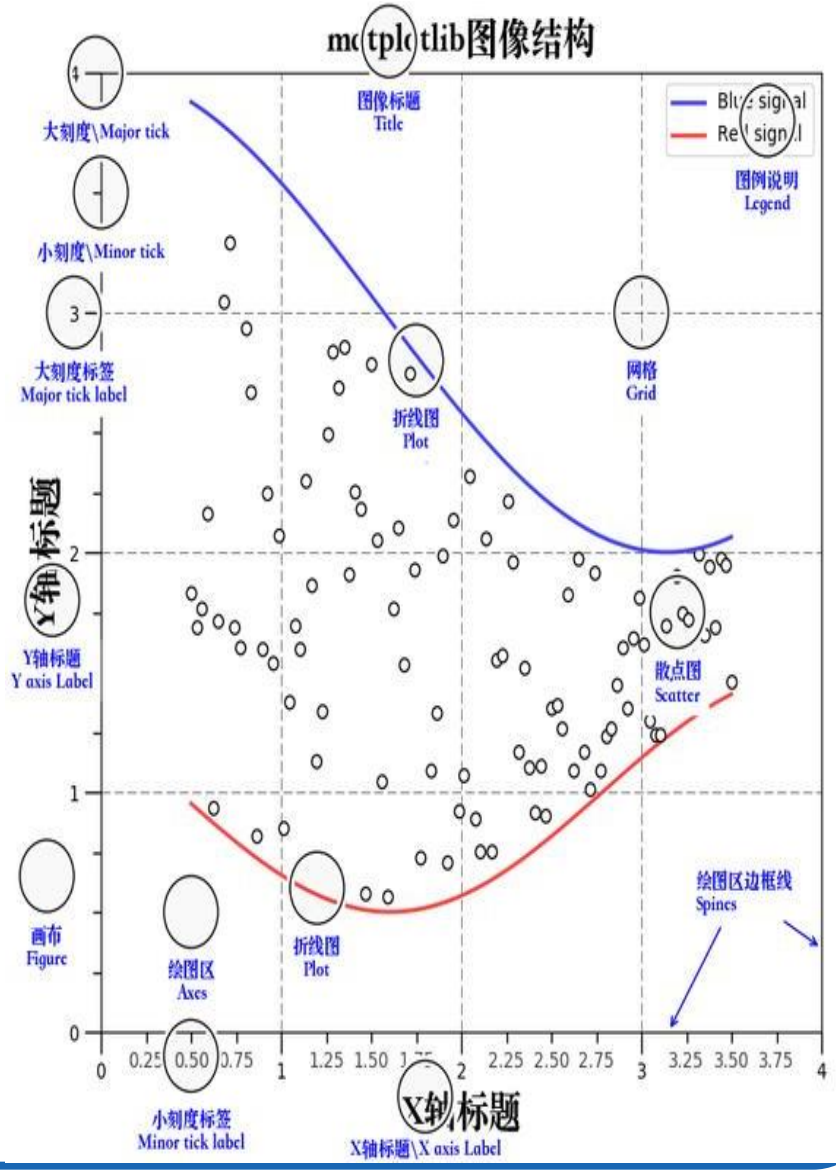
- 实际上不需要调用show()显示图表，可以直接用savefig()将图表保存成图像文件。使用这种方法可以很容易编写出 批量输出图表的程序。

2-2 图像组成


可视化
数据可视化概览
matplotlib
快速绘图
图像组成
基本类型
点线图、条形图、直方图、饼图
散点图、箱型图
复杂类型
多轴图
坐标共享轴
3D
图形属性设置
其他类库
Seaborn
Basemap（地图）
其他语言可视化工具
数据可视化使用小贴士

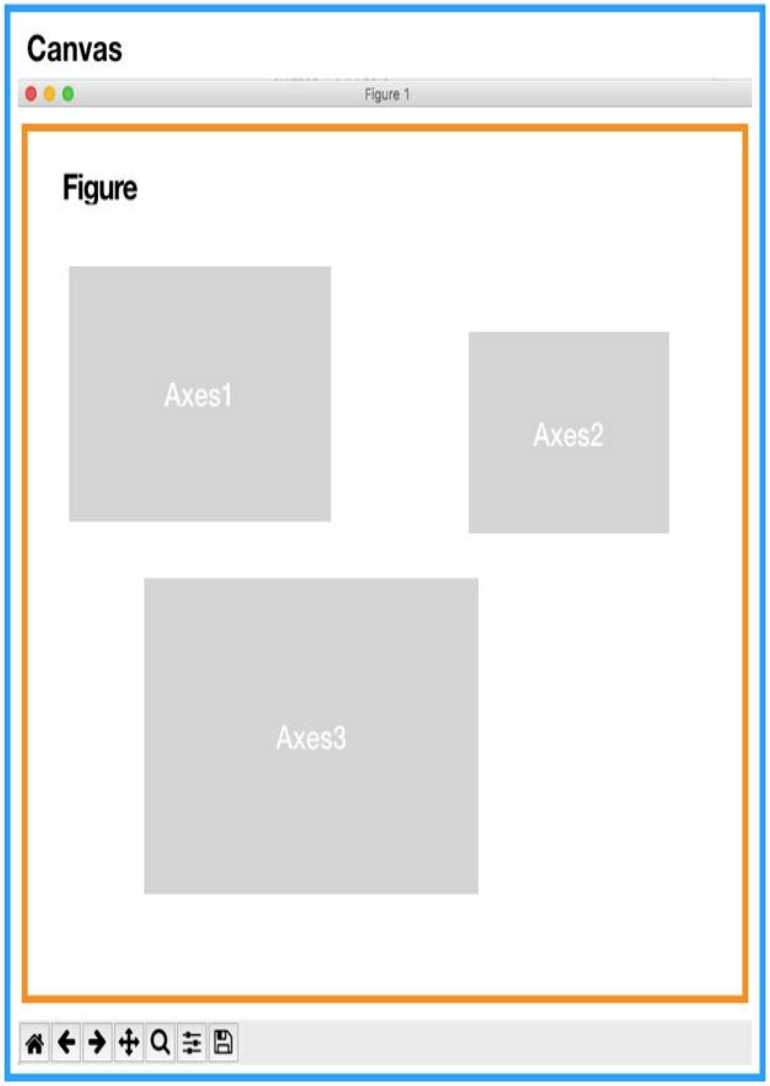
2-2 图像组成

- **容器层**，主要包括Canvas(画板)、Figure(画布、图片)、Axes(图表)；一个axes代表一个图表，包含一个plot，一个figure代表一张画布绘制一张图片，一张图片或一张画布可以有多个图表；canvas画板，摆放画布的工具。
- **辅助显示层**，主要包括Axis坐标轴、Spines、Tick、Grid、Legend、Title等，该层可通过set_axis_off()等方法设置不显示。
- **图像层**，即通过plot、contour、scatter等方法绘制的图像。



2-2 图像组成

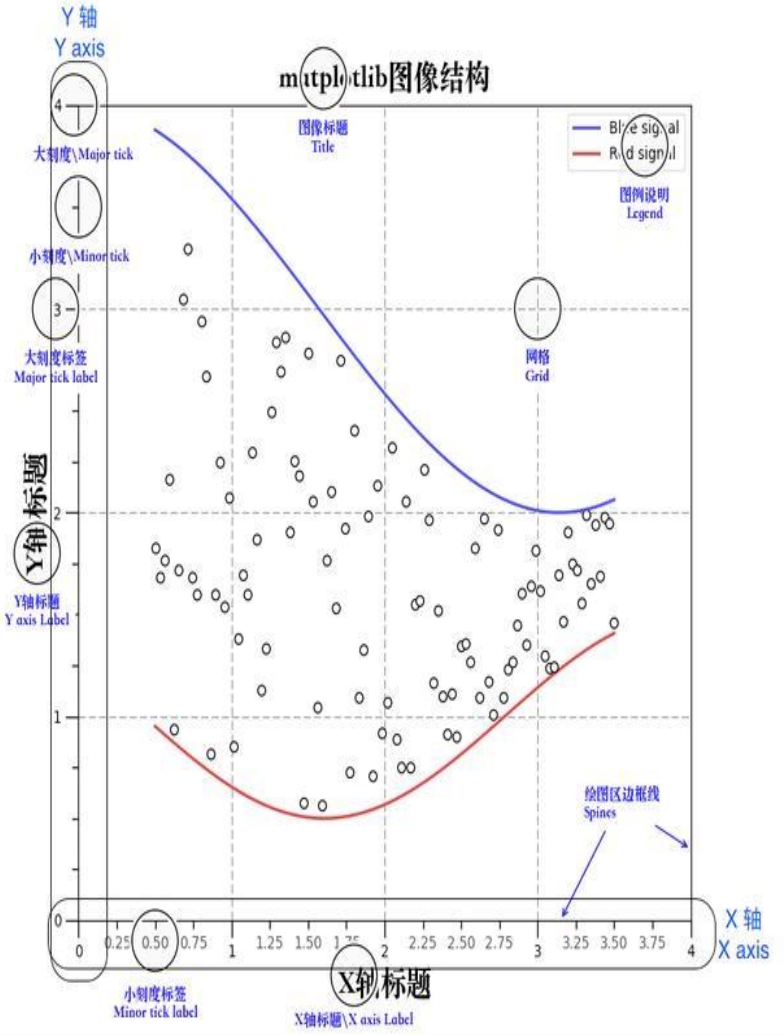
- 容器层包括：Canvas、Figure、Axes
  - Canvas是位于最底层的系统层，在绘图的过程中充当画板的角色，即放置画布的工具。
  - Figure是应用层的第一层，在绘图的过程中充当画布的角色。当对Figure大小、背景色彩等进行设置的时候，就相当于是选择画布大小、材质的过程。
  - Axes是应用层的第二层，在绘图的过程中相当于画布上的绘图区的角色。一个Figure对象可以包含多个Axes对象，每个Axes都是一个独立的坐标系，绘图过程中的所有图像都是基于坐标系绘制的。



2-2 图像组成

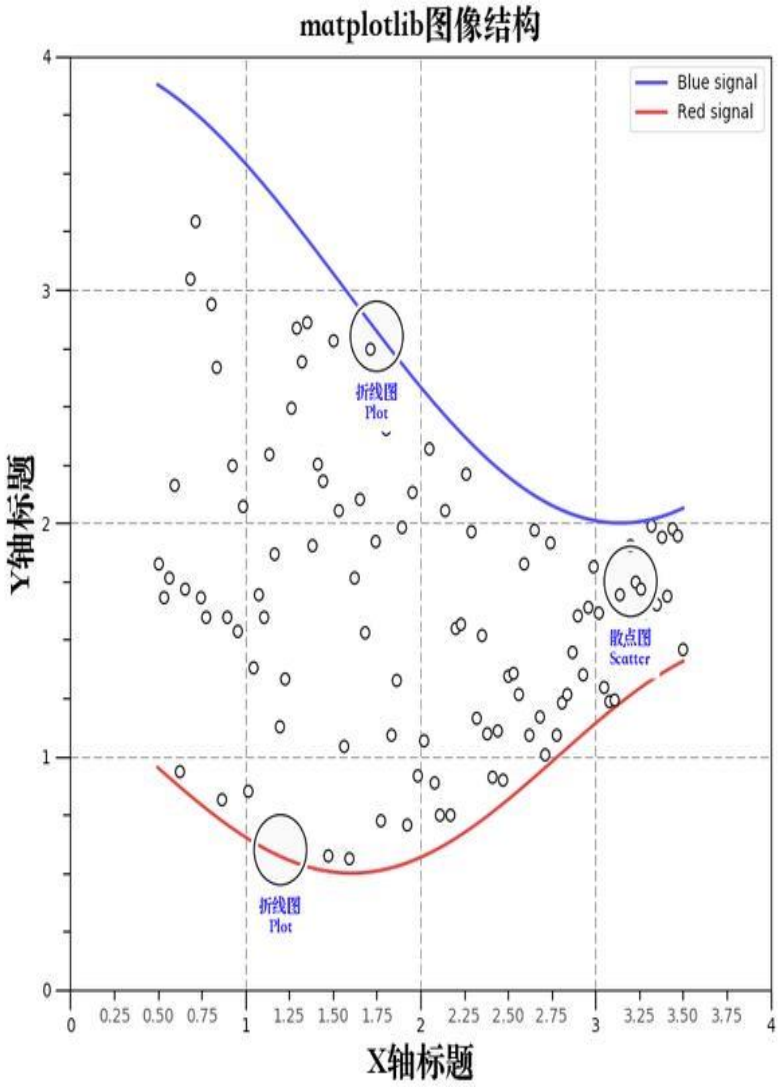
- **辅助显示层**为Axes内的除了根据数据绘制出的图像以外的内容，主要包括外观(facecolor)、边框线(spines)、坐标轴(axis)、坐标轴名称(axis label)、坐标轴刻度(tick)、坐标轴刻度标签(tick label)、网格线(grid)、图例(legend)、标题(title)等内容。

- 该层的设置可使图像显示更加直观更加容易被用户理解，但并不会对图像产生实质的影响。



2-2 图像组成

- **图像层**指Axes内通过plot折线图、scatter散点图、hist柱状图、contour轮廓图、bar柱状图、barbs、pie饼图等函数根据数据绘制出的图像。



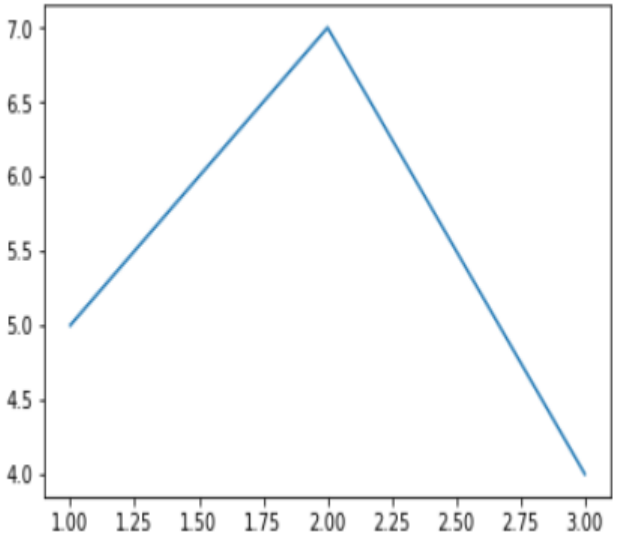
2-3 基本类型


可视化
数据可视化概览
matplotlib
快速绘图
图像组成
基本类型
点线图、条形图、直方图、饼图
散点图、箱型图
复杂类型
多轴图
坐标共享轴
3D
图形属性设置
其他类库
Seaborn
Basemap（地图）
其他语言可视化工具
数据可视化使用小贴士

2-3 基本绘图

- 点线图——plot
  - 适合用来呈现基于时间序列或有固定间隔的序列数据。横轴表示时间或者间隔，而纵轴则表示对应的数值。

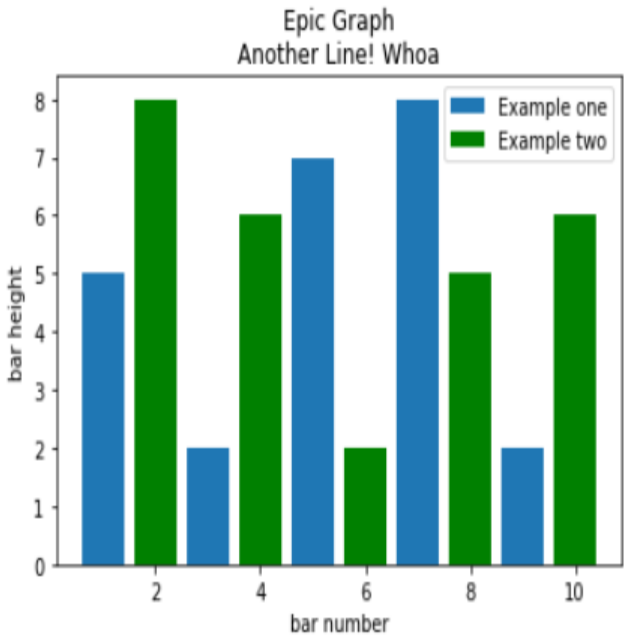
```
import matplotlib.pyplot as plt
plt.plot([1,2,3],[5,7,4])
plt.show()
```



2-3 基本绘图

- 条形图——bar
  - 主要是用来将数据分类显示，横轴表示数据的类型，而纵轴则表示对应类型的数值。

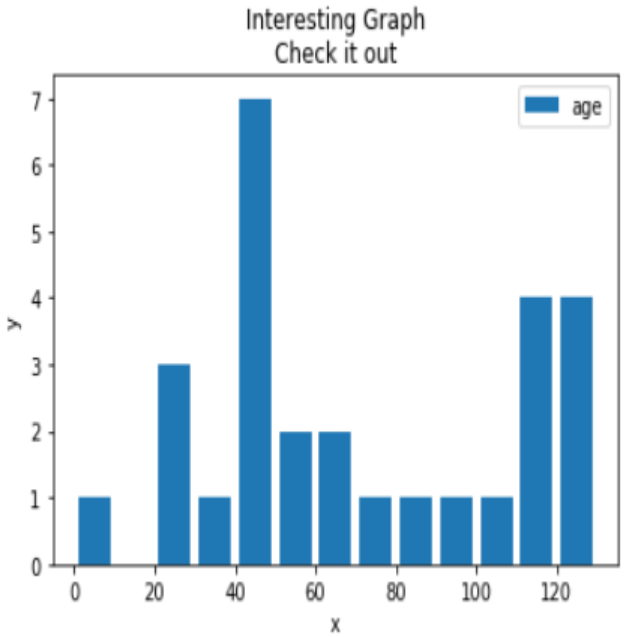
```
plt.bar([1,3,5,7,9],[5,2,7,8,2], label="Example one")
plt.bar([2,4,6,8,10],[8,6,2,5,6], label="Example two",
color='g')
```



2-3 基本绘图

- **直方图——hist**
  - **通常用来呈现变量的分布。它将数据按照一定的区间分组，而纵轴表示位于这一区间数据的个数。**

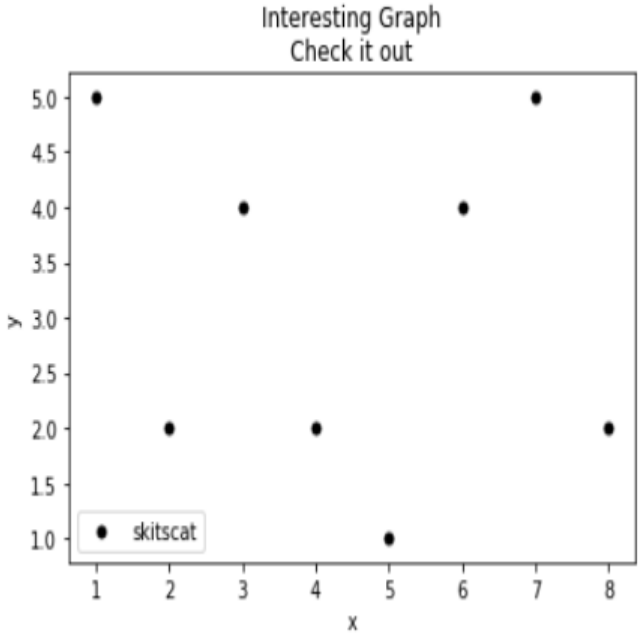
```
population_ages =
[22,55,62,45,21,22,34,42,42,4,99,102,1
10,120,121,122,130,111,115,112,80,75,
65,54,44,43,42,48]
bins =
[0,10,20,30,40,50,60,70,80,90,100,110,
120,130]
#直方图，rwidth为条的宽度
plt.hist(population_ages, bins,
histtype='bar', rwidth=0.8, label="age")
```



2-3 基本绘图

- **散点图——scatter**
  - **数据在直角坐标系平面的分布图，显示两组变量之间的关系。**

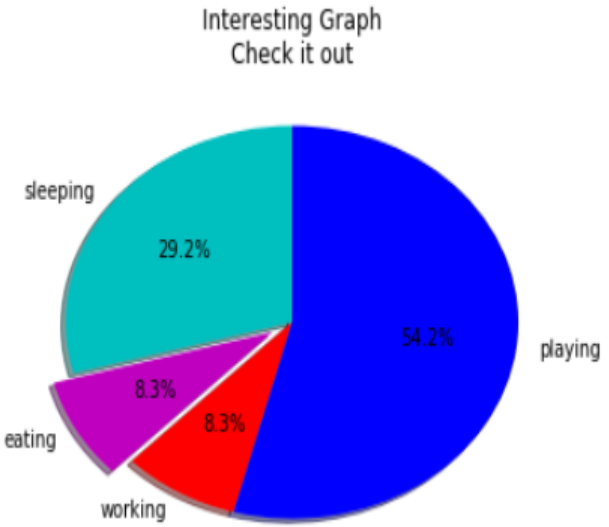
```
x = [1,2,3,4,5,6,7,8]
y = [5,2,4,2,1,4,5,2]
plt.scatter(x,y, label='skitscat', color='k', s=25, marker="o")
```



2-3 基本绘图

- 饼图——pie
  - 常用的统计学模块，可以显示每个部分大小与总和之间的比例。

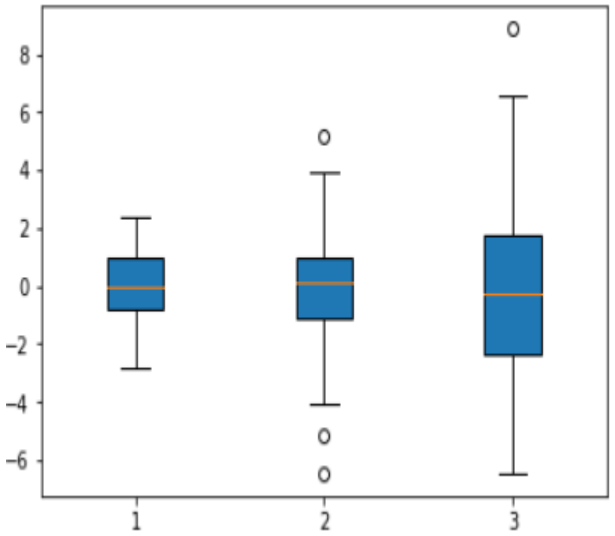
```
slices = [7,2,2,13]
activities =
['sleeping','eating','working','playing']
cols = ['c','m','r','b']

#slices是切片内容
plt.pie(slices,
        labels=activities,
        colors=cols,
        startangle=90,    #第一条线开始的角度
        shadow= True,
        explode=(0,0.1,0,0),    #一个切片拉出
        autopct='%1.1f%%')    #选择放置到图表
```



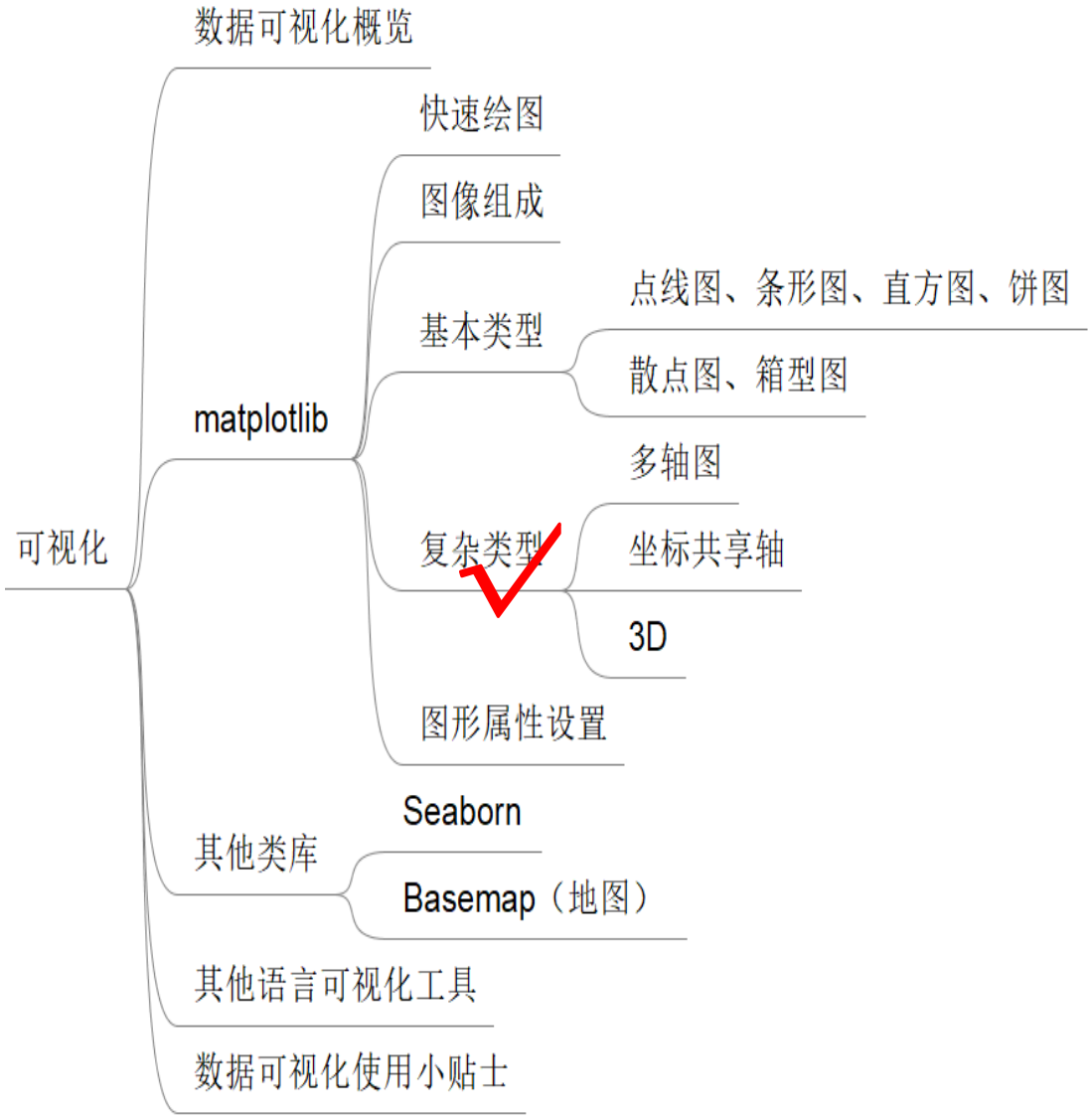
2-3 基本绘图

- **箱形图——boxplot**
  - **用来显示一组数据的分散情况，位于边缘线以外的数据点称为异常点。**

```
np.random.seed(123)
all_data = [np.random.normal(0, std, 100) for std in
range(1, 4)]
plt.boxplot(all_data, vert=True, patch_artist=True)
```



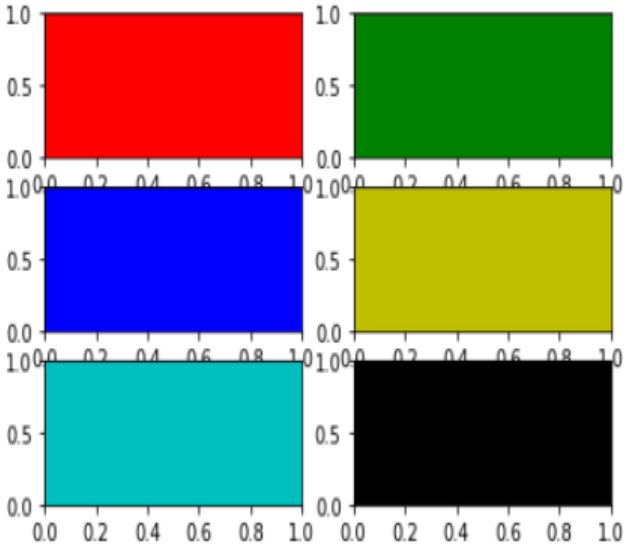
2-4 复杂绘图


可视化
数据可视化概览
matplotlib
快速绘图
图像组成
基本类型
点线图、条形图、直方图、饼图
散点图、箱型图
复杂类型
多轴图
坐标共享轴
3D
图形属性设置
其他类库
Seaborn
Basemap（地图）
其他语言可视化工具
数据可视化使用小贴士

2-4 复杂绘图——多轴图

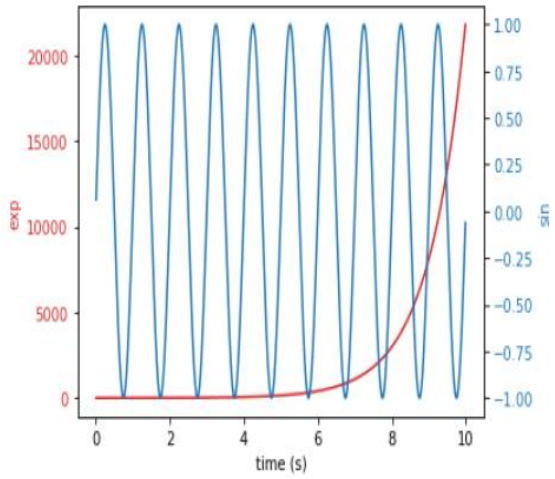
- 绘制多轴图
  - 一个绘图对象(figure)可以包含多个轴(axis)，在Matplotlib中用轴表示一个绘图区域，可以将其理解为子图。
  - 可以使用subplot函数快速绘制有多个轴的图表。

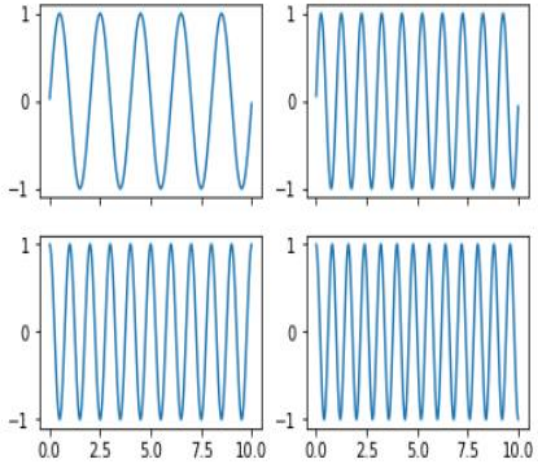
```
subplot(numRows, numCols, plotNum)
```



2-4 复杂绘图——共享坐标轴

- 共享坐标轴
  - 共享一个x轴坐标，y轴左右分别使用两个刻度来标识
  - Axes.twinx()，Axes.twiny()
    - 用来生成坐标轴实例以共享x或者y坐标轴
  - pyplot.subplots(nrows=1, ncols=1, sharex=False, sharey=False)
    - nrows,ncols: 用来确定绘制子图的行数和列数
    - sharex,sharey: 用来确定是否共享坐标轴刻度





2-4 复杂绘图——三维绘图

❖ 三维绘图

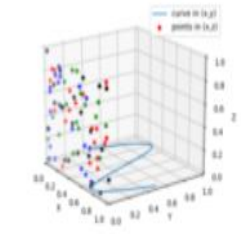
- **mpl_toolkits.mplot3d模块在matplotlib基础上提供了三维绘图的功能。由于它使用matplotlib的二维绘图功能来实现三维图形的绘制工作，因此绘图速度有限，不适合用于大规模数据的三维绘图。如果需要更复杂的三维数据可视化功能，可使用Mayavi。**

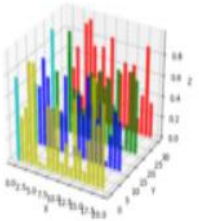
```
fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')
```

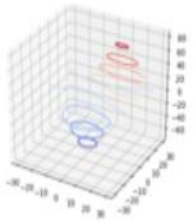
```
fig = plt.figure()
ax = p3d.Axes3D(fig)
```

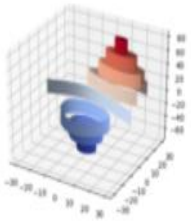
2-4 复杂绘图——三维绘图

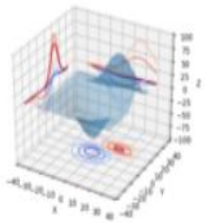
- 除了绘制三维曲面之外，Axes3D对象还提供了许多其他的三维绘图方法。可以通过下面的链接地址找到各种三维绘图的演示程序:
  - https://matplotlib.org/examples/mplot3d/index.html

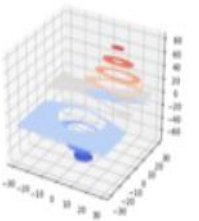

2dcollections3d_demo

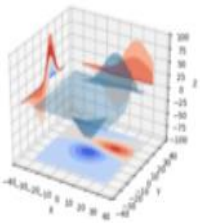

bars3d_demo

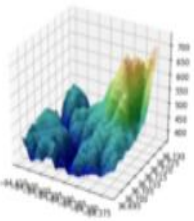

contour3d_demo


contour3d_demo2

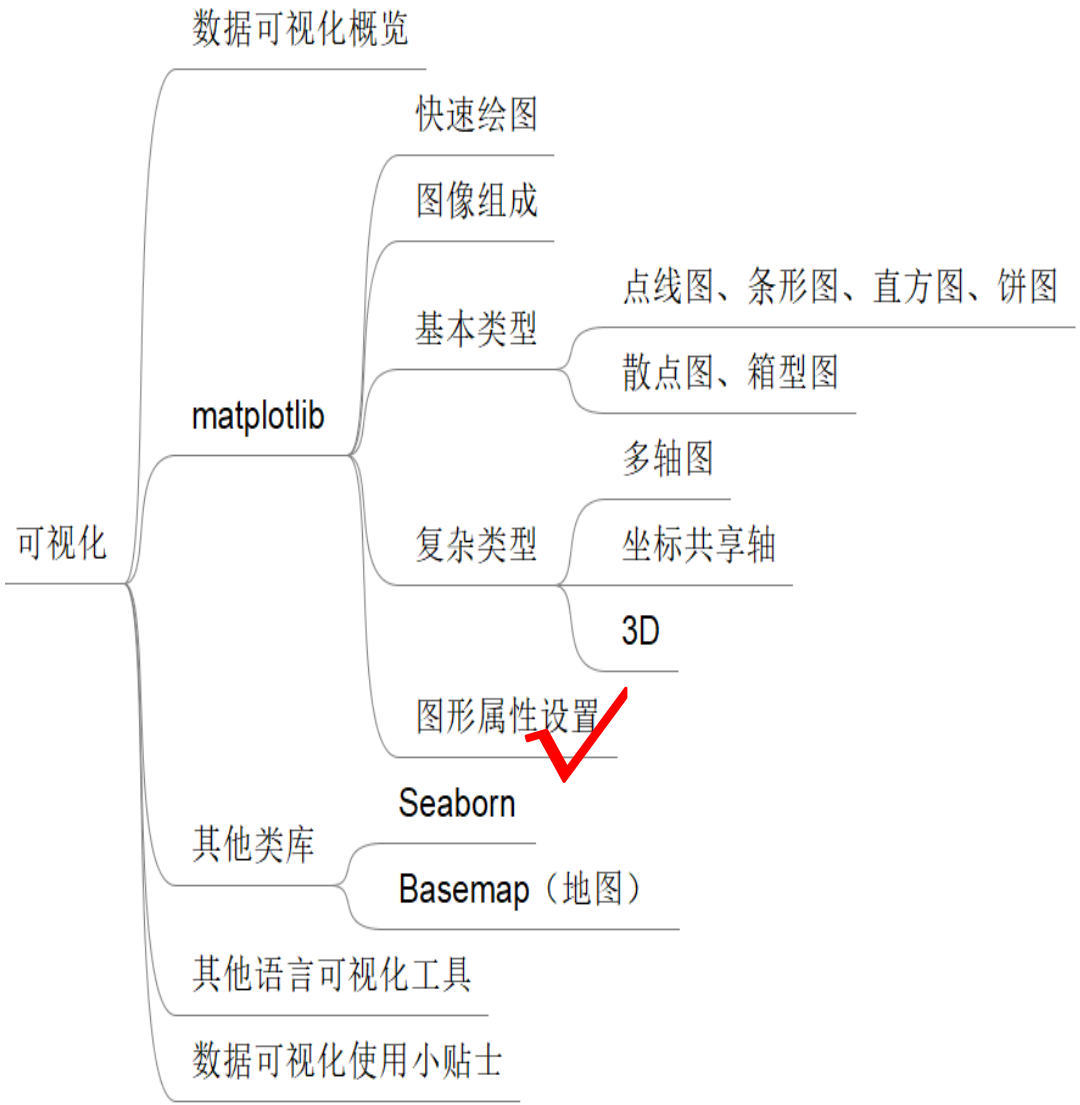

contour3d_demo3


contourf3d_demo


contourf3d_demo2


custom_shaded_3d_surface

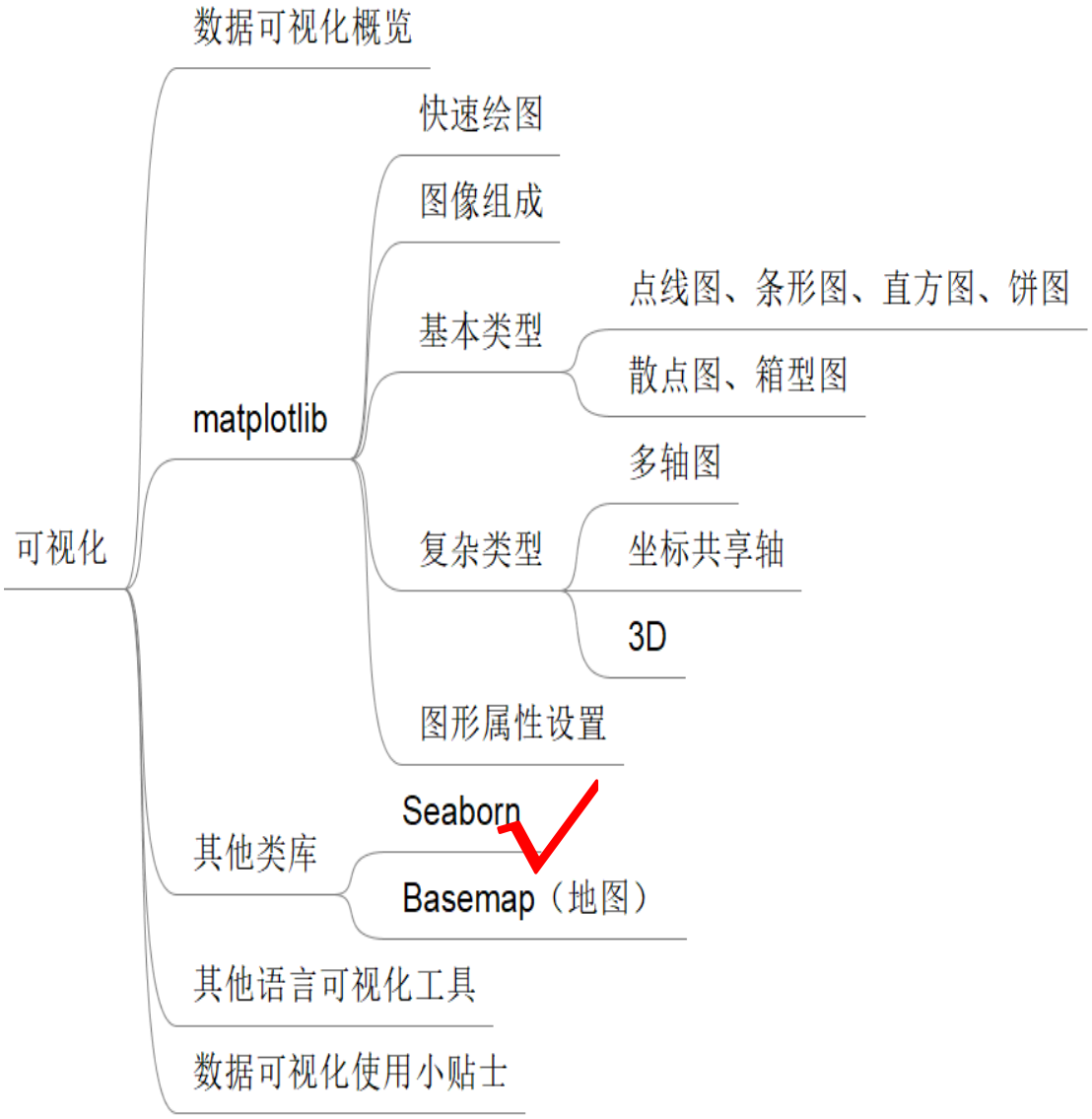
2-5 属性设置


可视化
数据可视化概览
matplotlib
快速绘图
图像组成
基本类型
点线图、条形图、直方图、饼图
散点图、箱型图
复杂类型
多轴图
坐标共享轴
3D
图形属性设置
其他类库
Seaborn
Basemap（地图）
其他语言可视化工具
数据可视化使用小贴士

2-5 属性设置

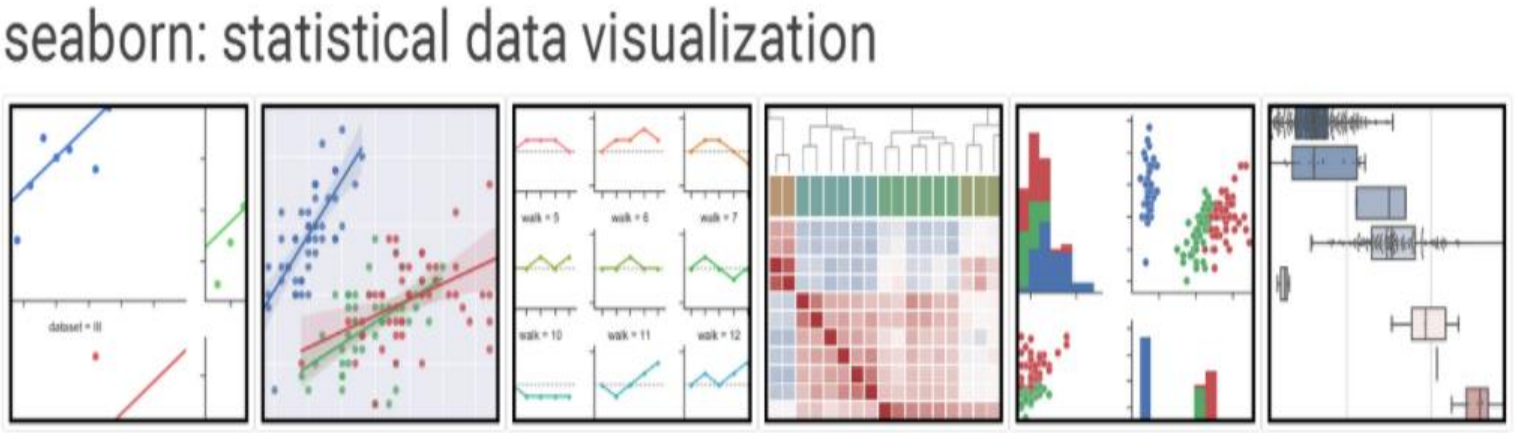
- axex: 设置坐标轴边界和表面的颜色、坐标刻度值大小和网格的显示
- backend: 设置目标暑促TkAgg和GTKAgg
- figure: 控制dpi、边界颜色、图形大小、和子区(subplot)设置
- font: 字体集（font family）、字体大小和样式设置
- grid: 设置网格颜色和线性
- legend: 设置图例和其中的文本的显示
- line: 设置线条（颜色、线型、宽度等）和标记
- patch: 是填充2D空间的图形对象，如多边形和圆。控制线宽、颜色等。
- savefig: 可以对保存的图形进行单独设置。例如设置渲染文件的背景为白色。
- verbose: 设置matplotlib在执行期间信息输出，如silent、helpful和debug等。
- xticks和yticks: 为x,y轴的主刻度和次刻度设置颜色、大小、方向等。

3-1 Seaborn


可视化
数据可视化概览
matplotlib
快速绘图
图像组成
基本类型
点线图、条形图、直方图、饼图
散点图、箱型图
复杂类型
多轴图
坐标共享轴
3D
图形属性设置
其他类库
Seaborn
Basemap（地图）
其他语言可视化工具
数据可视化使用小贴士

3-1 Seaborn简介

- Seaborn库相比较于matplotlib，它使得绘制较为复杂的图更加容易，并且不需要什么设置就能绘制出更为美观的图。也就是其官方语make a well-defined set of hard things easy，本质上，它就是matplotlib的高阶接口。



3-1 Seaborn特点

- 计算多变量间关系的面向数据集接口
- 可视化类别变量的观测与统计
- 可视化单变量或多变量分布并与其子数据集比较
- 控制线性回归的不同因变量并进行参数估计与作图
- 对复杂数据进行易行的整体结构可视化
- 对多表统计图的制作高度抽象并简化可视化过程
- 提供多个内建主题渲染 matplotlib 的图像样式
- 提供调色板工具生动再现数据

3-1 Seaborn使用

- Seaborn 将 matplotlib 的参数划分为两个独立的组合。
  - 设置绘图的外观风格的
  - 将绘图的各种元素按比例缩放的
- 操控这些参数的接口主要有两对方法：
  - 控制风格：axes_style(), set_style()
    - darkgrid, whitegrid, dark, white, ticks
  - 缩放绘图：plotting_context(), set_context()
    - paper, notebook, talk, poster

3-2 Basemap

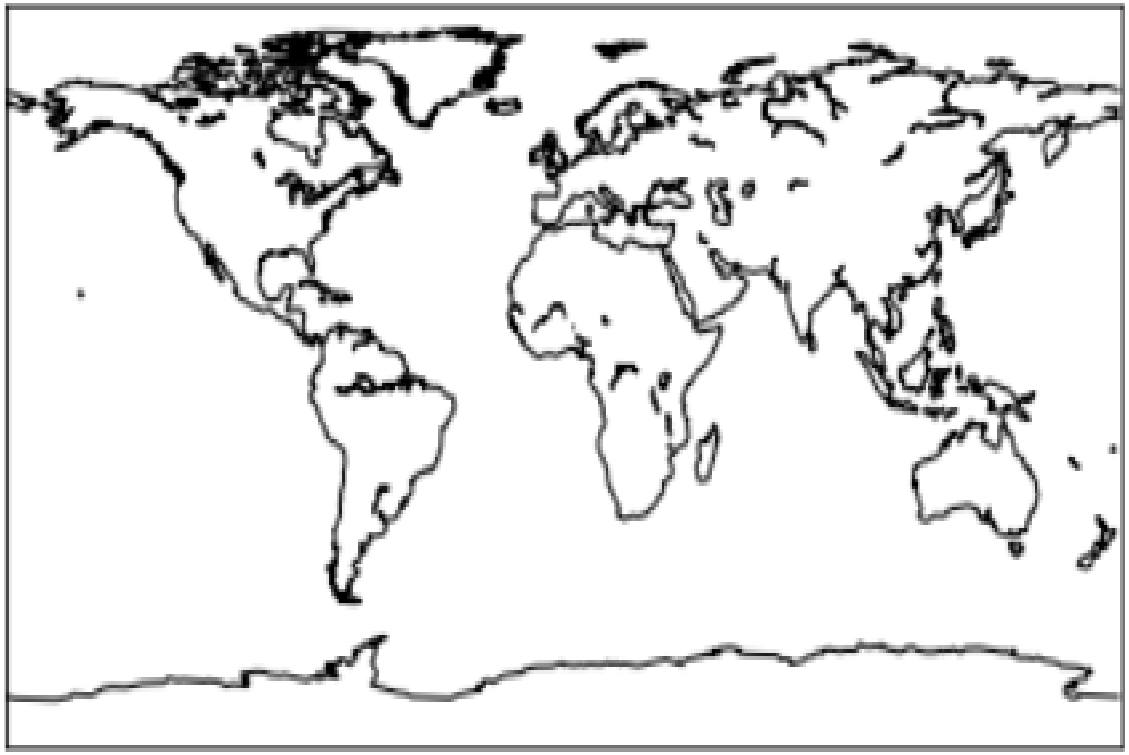

可视化
数据可视化概览
matplotlib
快速绘图
图像组成
基本类型
点线图、条形图、直方图、饼图
散点图、箱型图
复杂类型
多轴图
坐标共享轴
3D
图形属性设置
其他类库
Seaborn
Basemap（地图）
其他语言可视化工具
数据可视化使用小贴士

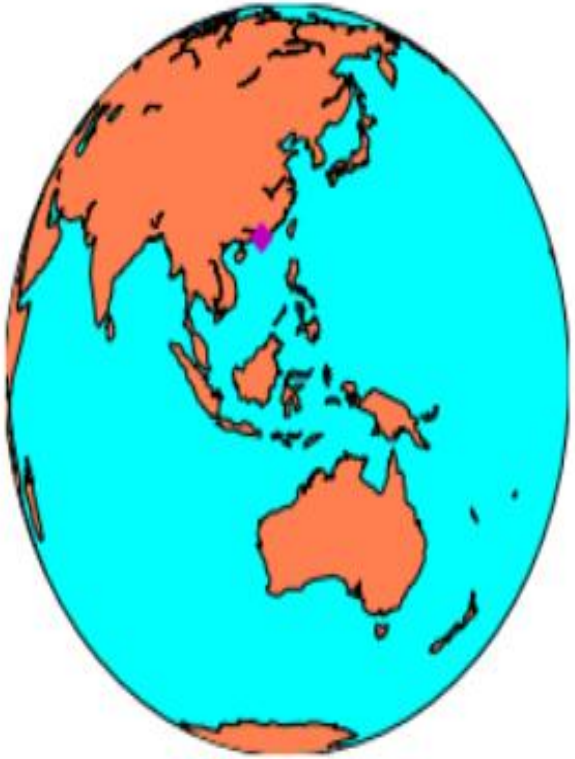
3-2 Basemap简介

- Basemap 是Matplotlib 的一个扩展，使得Matplotlib更方便处理地理数据。
- 在使用Basemap进行地理数据分析时分几个步骤：
  - 对特定的地图投影创建一个新的Basemap实例
  - 利用Basemap将球面坐标系转换为笛卡儿坐标
  - 利用 Matplotlib和Basemap来个性化这个地图
  - Show()函数显示这个地图

https://basemaptutorial.readthedocs.io/en/latest/

3-2 地理信息可视化





4 小贴士


可视化
数据可视化概览
matplotlib
快速绘图
图像组成
基本类型
点线图、条形图、直方图、饼图
散点图、箱型图
复杂类型
多轴图
坐标共享轴
3D
图形属性设置
其他类库
Seaborn
Basemap（地图）
其他语言可视化工具
数据可视化使用小贴士

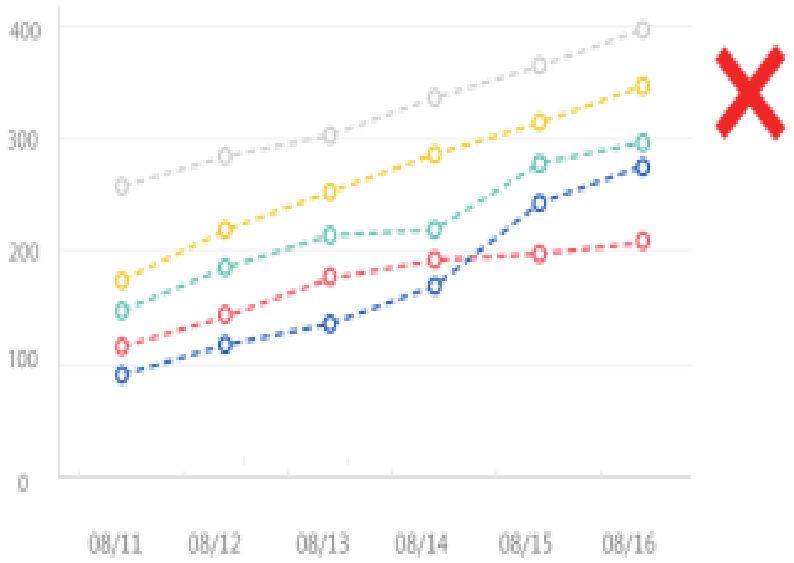
4 数据可视化使用小贴士

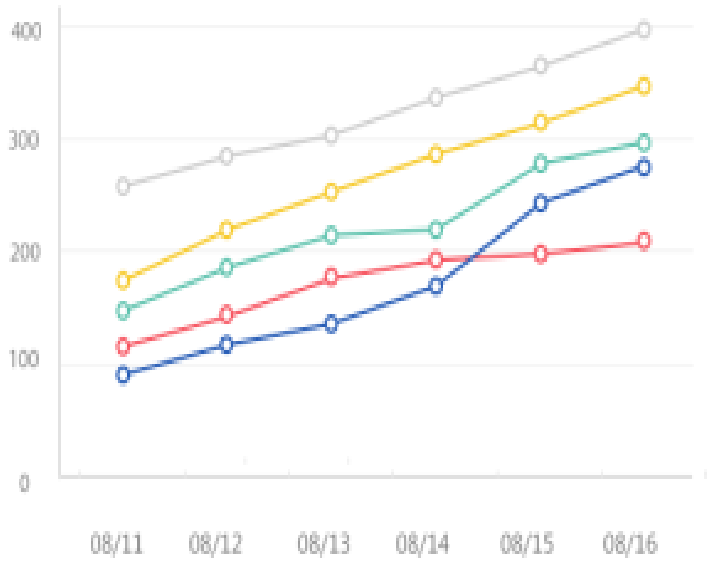
- **饼图顺序不当**

将份额最大的放在12点方向，顺时针放置第二大份额的部分，以此类推。

4 数据可视化使用小贴士

- 在线状图中使用虚线

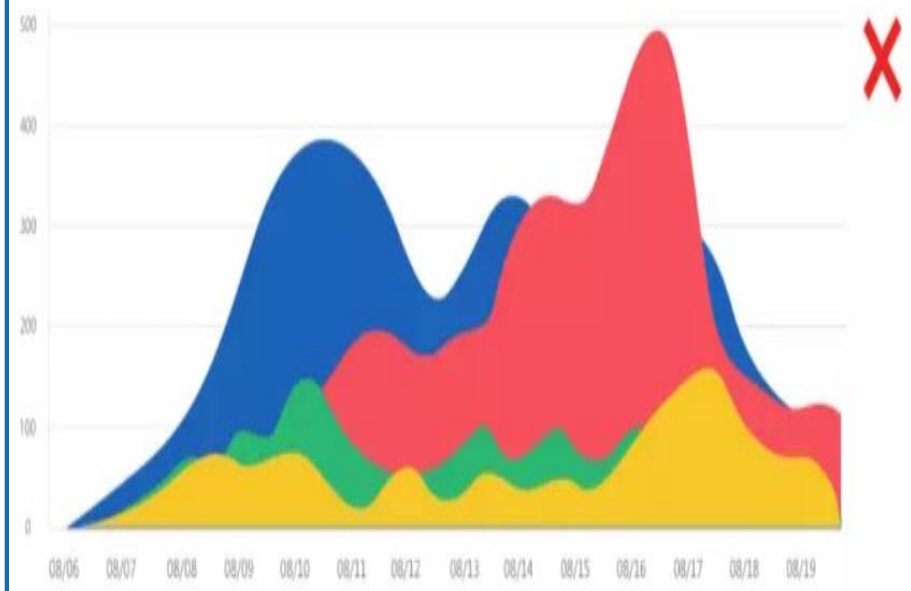


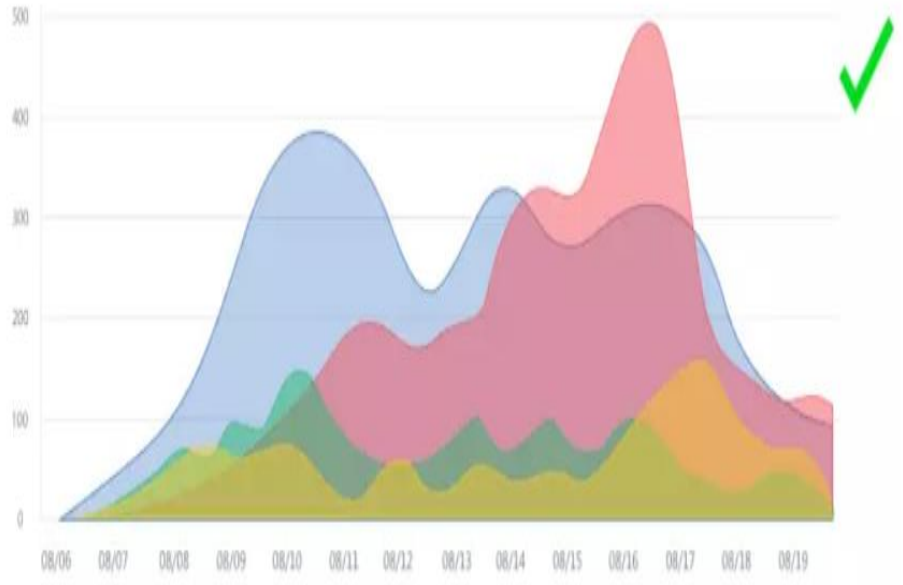


虚线会让人分心，用实线搭配合适的颜色更容易区分。

4 数据可视化使用小贴士

- **数据被遮盖**

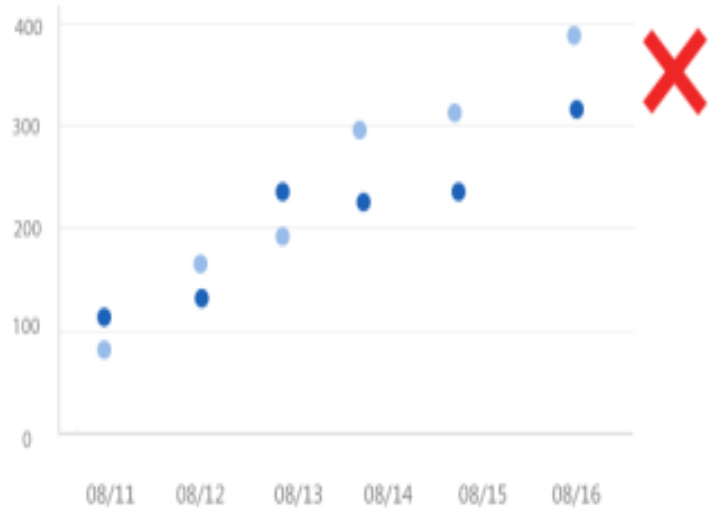


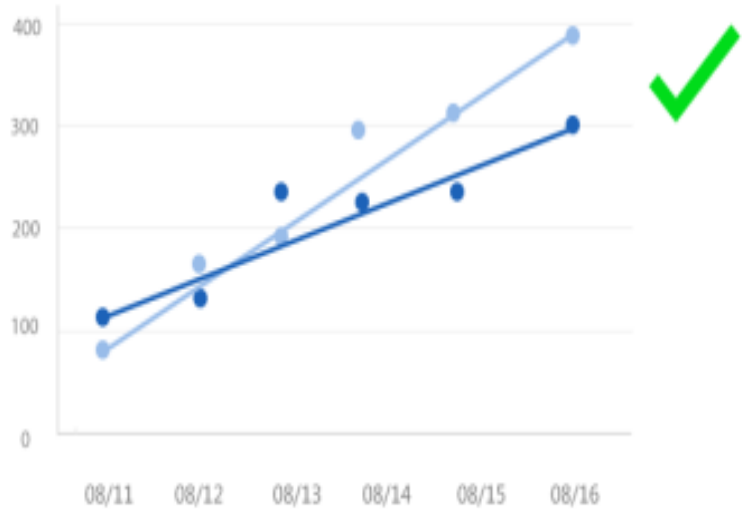


确保数据不会因为设计而丢失或被覆盖。
例如在面积图中使用透明效果来确保用户可以看到全部数据。

4 数据可视化使用小贴士

- **耗费用户更多的精力**





通过辅助的图形元素来使数据更易于理解，比如在散点图中增加趋势线。

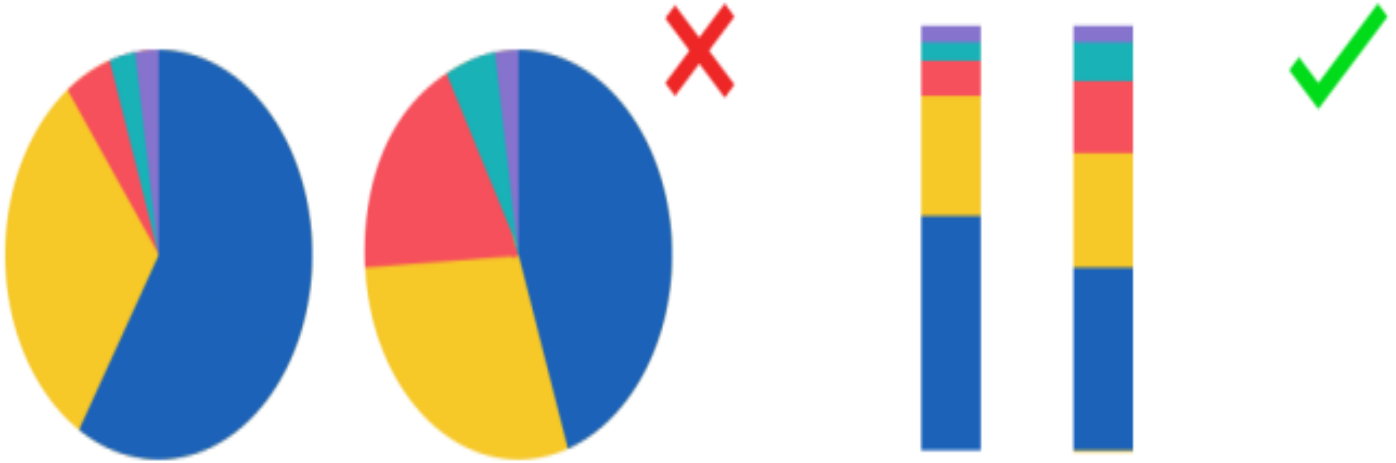
4 数据可视化使用小贴士

- **柱状过宽或过窄**

经过调研，柱子的间隔最好调整为宽的1/2。

4 数据可视化使用小贴士

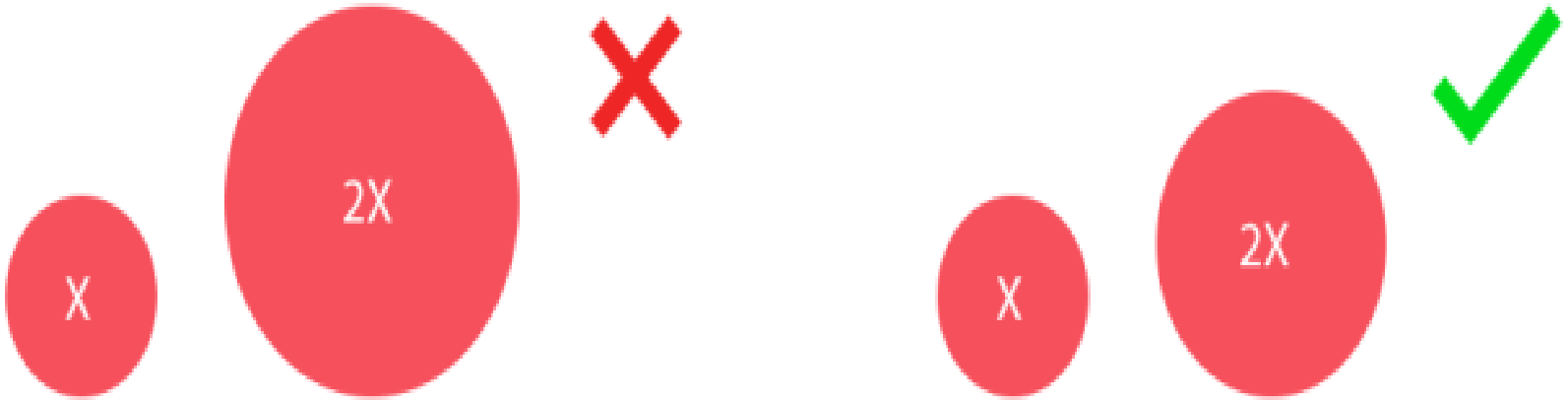
- 数据对比困难



选择合适的图表，让数据对比更明显直接。
柱状图比饼图在视觉上更易于比较。

4 数据可视化使用小贴士

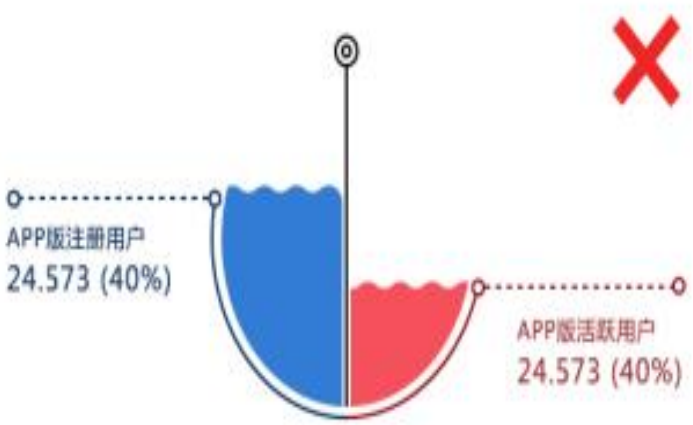
- 错误呈现数据

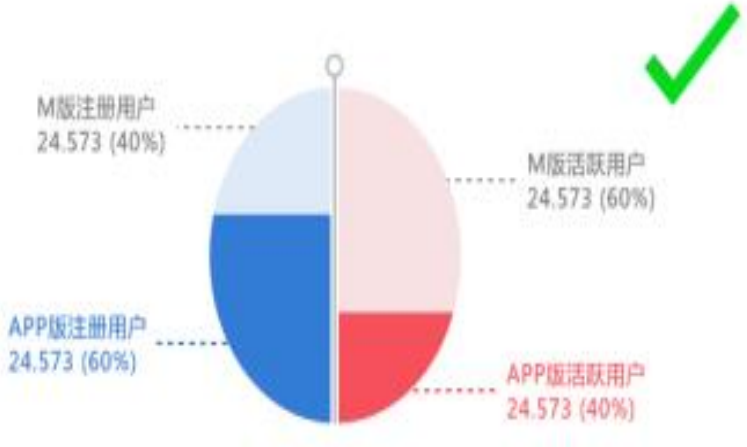


确保任何呈现都是准确的，比如，上图气泡图的面积大小应该跟数值一样。

4 数据可视化使用小贴士

- 不要过分设计

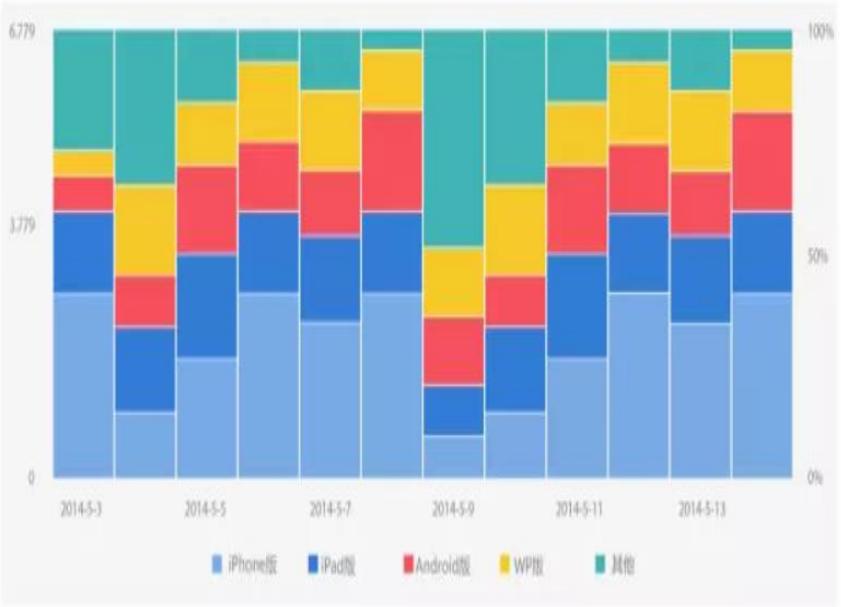


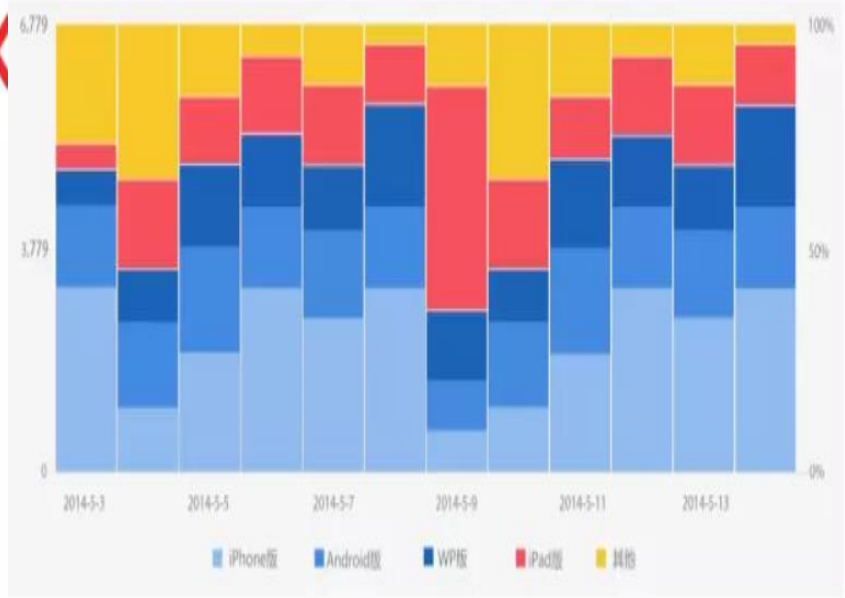


清楚标明各个图形表示的数据，避免用与主要数据不相关的颜色，形状干扰视觉。

4 数据可视化使用小贴士

- 数据没有很好归类，没有重点区分





将同类数据归类，简化色彩，帮助用户更快理解数据。上图的第一张没有属于同类型手机中不同系统进行颜色上的归类，从而减少了比较的作用。通过蓝色系很好的把iPhone,Android,WP版归为一类，很好的与iPad版、其他比较。

数据可视化使用小贴士

- 误导用户的图表

要客观反映真实数据，纵坐标不能被截断，否则视觉感受和实际数据相差很大。
上图的数据起始点被截断从50开始。