# 第1章 数据可视化与matplotlib

1. **填空题**
2. 图形
3. 图表
4. 箱形图
5. 2D
6. Anaconda
7. **判断题**
8. √
9. ×
10. ×
11. ×
12. **选择题**
13. C
14. C
15. D
16. D
17. A，B，C，D
18. **简答题**
19. 答：数据可视化旨在借助图形化的手段，将一组数据以图形的形式表示，并利用数据分析和开发工具发现其中未知信息的处理过程。
20. 答：折线图是将数据标注成点，并通过直线将这些点按某种顺序连接而成的图表，它以折线的方式反映事物沿某一维度的变化趋势，能够清晰地展示数据增减的趋势、速率、规律及峰值等特征；柱形图是由一系列宽度相等、高低不齐的纵向矩形条组成的图表，它使用矩形条的高度表示数据的多少，以此反映不同分类数据之间的差异；饼图是由若干个面积大小不一、颜色不同的扇形组成的圆形图表，它使用圆表示数据的总量，组成圆的每个扇形表示数据中各项占总量的比例大小，主要用于显示数据中各项大小与各项总和的比例。
21. 答：当使用pyplot API绘图时，用户需要先使用“import matplotlib.pyplot as plt”语句导入pyplot模块，之后使用该模块调用绘图函数即可在当前的画布和绘图区域中绘制图表；当使用object-oriented API绘图时，用户需要先创建画布（pyplot.Figure类对象），再在该画布上创建坐标系风格的绘图区域（pyplot.Axes类对象），之后调用绘图方法创建图形，将这些创建的对象组合到一起才完成一次完整的绘图。
22. **编程题**

答案：

# ————面向对象的方式

import numpy as np

import matplotlib.pyplot as plt

x\_data = np.linspace(-np.pi, np.pi, 256, endpoint=True)

y\_sin, y\_cos = np.sin(x\_data), np.cos(x\_data)

fig = plt.figure()

ax = fig.add\_subplot(111)

ax.plot(x\_data, y\_sin)

ax.plot(x\_data, y\_cos)

plt.show()

# ————面向函数的方式

import numpy as np

import matplotlib.pyplot as plt

x\_data = np.linspace(-np.pi, np.pi, 256, endpoint=True)

y\_sin, y\_cos = np.sin(x\_data), np.cos(x\_data)

plt.plot(x\_data, y\_sin)

plt.plot(x\_data, y\_cos)

plt.show()

# 第2章 使用matplotlib绘制简单图表

1. **填空题**
2. Line2D
3. 堆积图
4. 10
5. **判断题**
6. ×
7. √
8. ×
9. **选择题**
10. D
11. C
12. B
13. D
14. C
15. **编程题**
16. （1）答案：

import numpy as np

import matplotlib.pyplot as plt

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

plt.rcParams['axes.unicode\_minus'] = False

labels = np.array(['语文', '数学', '英语', '物理', '化学', '生物'])

x = np.arange(0, labels.size)

y\_men = np.array([85.5, 91, 72, 59, 66, 55])

y\_women = np.array([94, 82, 89.5, 62, 49, 53])

bar\_width = 0.35

# 绘制柱形图

plt.bar(x - bar\_width /2, y\_men, width=bar\_width, tick\_label=labels)

plt.bar(x + bar\_width /2, y\_women, width=bar\_width, tick\_label=labels)

plt.show()

（2）答案：

import numpy as np

import matplotlib.pyplot as plt

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

plt.rcParams['axes.unicode\_minus'] = False

labels = np.array(['语文', '数学', '英语', '物理', '化学', '生物'])

x = np.arange(0, labels.size)

y\_men = np.array([85.5, 91, 72, 59, 66, 55])

y\_women = np.array([94, 82, 89.5, 62, 49, 53])

bar\_width = 0.5

# 绘制柱形图

plt.bar(x, y\_men, width=bar\_width, tick\_label=labels)

plt.bar(x, y\_women, bottom=y\_men, width=bar\_width, tick\_label=labels)

plt.show()

1. 答案：

import numpy as np

import matplotlib.pyplot as plt

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

plt.rcParams['axes.unicode\_minus'] = False

# 饼图标签

kinds = ['童装', '奶粉辅食', '孕妈专区', '洗护喂养', '宝宝尿裤',

'春夏新品', '童车童床', '玩具文娱', '童鞋']

# 饼图的数据

sales = np.array([29665, 3135.4, 4292.4, 5240.9, 5543.4,

5633.8, 6414.5, 9308.1, 10353])

sales\_total = np.sum(sales)

sales\_scale = [i/sales\_total for i in sales]

# 绘制饼图

plt.pie(sales\_scale, labels=kinds, autopct='%3.1f%%')

plt.show()

# 第3章 图表辅助元素的定制

1. **填空题**
2. 图形
3. 标识
4. 指示箭头
5. 参考线
6. 数学公式
7. **判断题**
8. ×
9. √
10. ×
11. √
12. √
13. **选择题**
14. B
15. C
16. B
17. A
18. D
19. **简答题**
20. 答：指向型注释文本是指通过指示箭头的注释方式对绘图区域的图形进行解释的文本，它一般使用线条连接说明点和箭头指向的注释文字；无指向型注解文本是指单纯地使用文字的注释方式对绘图区域的图形进行说明的文本。
21. 答：坐标轴是用于定义坐标系的一组直线或曲线；标题是图表的名称，可以迅速地让读者理解图表要说明的内容；图例是一个列举的各组图形标识方式的方框图，可以帮助用户明确每组图形代表的含义；网格是从坐标轴刻度开始的、贯穿绘图区域的若干条线，用于作为估算图形所示值的标准；参考线是标记坐标轴上特殊值的一条直线；参考区域是标记坐标轴上特殊范围的一块区域；注释文本是对图形的一些注释和说明；表格主要用于强调比较难理解的数据。
22. **编程题**
23. 答案：

import numpy as np

import matplotlib.pyplot as plt

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

plt.rcParams['axes.unicode\_minus'] = False

# 添加无指向型注释文本

def autolabel(rects):

"""在每个矩形条的上方附加一个文本标签，以显示其高度"""

for rect in rects:

height = rect.get\_height()

plt.text(rect.get\_x() + rect.get\_width() / 2, height + .5,

s='{}'.format(height),

ha='center', va='bottom')

labels = np.array(['语文', '数学', '英语', '物理', '化学', '生物'])

x = np.arange(0, labels.size)

y\_men = np.array([85.5, 91, 72, 59, 66, 55])

y\_women = np.array([94, 82, 89.5, 62, 49, 53])

bar\_width = 0.35

# 绘制柱形图

bars\_men = plt.bar(x - bar\_width /2, y\_men, width=bar\_width, label='男生')

bars\_women = plt.bar(x + bar\_width /2, y\_women,

width=bar\_width, label='女生')

plt.xticks(x, labels)

plt.title('高二男生、女生的平均成绩')

plt.ylabel('平均成绩（分）')

autolabel(bars\_men)

autolabel(bars\_women)

plt.legend()

plt.show()

1. 答案：

import numpy as np

import matplotlib.pyplot as plt

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

plt.rcParams['axes.unicode\_minus'] = False

# 饼图标签

kinds = ['童装', '奶粉辅食', '孕妈专区', '洗护喂养', '宝宝尿裤',

'春夏新品', '童车童床', '玩具文娱', '童鞋']

# 饼图的数据

sales = np.array([29665, 3135.4, 4292.4, 5240.9, 5543.4,

5633.8, 6414.5, 9308.1, 10353])

sales\_total = np.sum(sales)

sales\_scale = [i/sales\_total for i in sales]

# 表格数据

cell\_text = [29665, 3135.4, 4292.4, 5240.9, 5543.4,

5633.8, 6414.5, 9308.1, 10353]

# 绘制饼图

plt.pie(sales\_scale, autopct='%3.1f%%')

plt.title('拼多多平台子类目的销售额')

plt.legend(kinds, loc='upper right', bbox\_to\_anchor=[1.7, 1.0], ncol=2)

# 添加表格

plt.table(cellText=[cell\_text],

cellLoc='center',

rowLabels=['销售额(亿)'],

colLabels=kinds,

loc='lower center')

plt.show()

# 第4章 图表样式的美化

1. **填空题**
2. rcParams
3. 颜色
4. 数据标记
5. Text
6. use()
7. **判断题**
8. ×
9. √
10. ×
11. √
12. √
13. **选择题**
14. A
15. D
16. D
17. C
18. A
19. **简答题**
20. 答案：局部修改的方式是指通过代码动态地修改matplotlib配置项，此方式用于程序局部定制的需求；全局修改的方式是指直接修改matplotlibrc文件的配置项，此方式用于对程序全局定制的需求，可以将指定的图表样式进行统一修改，无需每次在具体的程序中进行单独修改。
21. 答案：fill()函数用于填充多边形，fill\_between()或fill\_betweenx()函数分别用于填充两条水平曲线或垂直曲线之间的区域。
22. **编程题**
23. 答案：

import numpy as np

import matplotlib.style as ms

import matplotlib.pyplot as plt

plt.rcParams["font.sans-serif"] = ["SimHei"]

plt.rcParams["axes.unicode\_minus"] = False

# 快递业务量

express\_2018 = np.array([39, 20, 40, 38, 42, 43, 41, 41, 45, 48, 52, 50])

express\_2019 = np.array([45, 28, 48, 49, 50, 51, 50, 50, 51, 52, 70, 65])

date\_x = np.arange(1, 13)

fig = plt.figure()

ax = fig.add\_subplot(111)

# 第1条折线

ax.plot(date\_x, express\_2018, color='#8B0000', marker='^',

linestyle='--', linewidth=1.5, label='2018年')

# 第2条折线

ax.plot(date\_x, express\_2019, color='#006374', marker='d',

linewidth=1.5, label='2019年')

ax.set\_ylabel('业务量（亿件）')

ax.legend()

ms.use('fivethirtyeight')

plt.show()

1. 答案：

import numpy as np

import matplotlib.pyplot as plt

x = np.linspace(-np.pi, np.pi, 256, endpoint=True)

y\_cos, y\_sin = np.cos(x), np.sin(x)

plt.plot(x, y\_cos, color='blue', linewidth=1.0, label='COS', alpha=0.5)

plt.plot(x, y\_sin, color='red', linewidth=1.0, label='SIN')

plt.title('Cos & Sin')

plt.xticks([-np.pi, -np.pi / 2.0, 0, np.pi / 2.0, np.pi],

[r'$-\pi$', r'$-\pi/2$', r'$0$', r'$\pi/2$', r'$\pi$'])

plt.yticks(np.linspace(-1, 1, 5, endpoint=True))

plt.legend(loc='upper left')

plt.grid()

# 填充两条线间的颜色

plt.fill\_between(x, np.abs(x) < 0.5, y\_cos, y\_cos > 0.5,

color='green', alpha=0.25)

# 加注释文本

plt.annotate('cos(1)', xy=(1, np.cos(1)), xycoords='data', xytext=(10, 30),

textcoords='offset points', arrowprops=dict(arrowstyle='->'))

plt.show()

# 第5章 子图的绘制及坐标轴共享

1. **填空题**
2. 等分
3. sharex
4. 约束布局
5. gridspec
6. **判断题**
7. √
8. ×
9. √
10. √
11. ×
12. **选择题**
13. D
14. C
15. A
16. C
17. D
18. **简答题**
19. 答案：pyplot使用subplot()函数可以在规划好的某个区域中绘制单个子图；pyplot使用subplots()函数可以在规划好的全部区域中绘制多个子图；pyplot使用subplot2grid()函数可以将整个画布规划成非等分布局的区域，之后在选中的某个区域中绘制单个子图。
20. 答案：约束布局是指通过一系列限制来确定画布中元素的位置的方式，它预先会确定一个元素的绝对定位，之后以该元素的为基点将其它元素进行绝对定位，从而灵活地定位和调整元素的位置。
21. **编程题**
22. 答案：

import numpy as np

import matplotlib.pyplot as plt

x = np.linspace(-np.pi, np.pi, 256, endpoint=True)

sin, cos = np.sin(x), np.cos(x)

ax\_one = plt.subplot(233)

ax\_one.plot(x, sin)

ax\_two = plt.subplot(236, sharex=ax\_one)

ax\_two.plot(x, cos)

plt.show()

1. 答案：

import matplotlib.pyplot as plt

fig = plt.figure(constrained\_layout=True)

gs = fig.add\_gridspec(3, 4)

fig.add\_subplot(gs[0, :])

fig.add\_subplot(gs[1, :-2])

fig.add\_subplot(gs[1, -2:])

fig.add\_subplot(gs[2, 0])

fig.add\_subplot(gs[2, 1:])

plt.show()

# 第6章 坐标轴的定制

1. **填空题**
2. 主刻度线
3. 轴脊
4. axis()
5. **判断题**
6. √
7. ×
8. ×
9. **选择题**
10. C
11. C
12. D
13. A
14. B
15. **简答题**

答案：Locator是刻度定位器的基类，它派生了很多子类，可以自动地调整刻度的间隔、选择刻度的位置；Formatter是刻度格式器的基类, 它派生了很多子类，可以自动地调整刻度标签的格式。

1. **编程题**

答案：

import numpy as np

import matplotlib.pyplot as plt

from calendar import month\_name, day\_name

from matplotlib.ticker import FormatStrFormatter

plt.rcParams["font.sans-serif"] = ["SimHei"]

plt.rcParams["axes.unicode\_minus"] = False

x\_data = np.arange(1,8,1)

y\_data = [44.98, 45.02, 44.32, 41.05, 42.08, 42.08, 42.08]

fig = plt.figure()

ax = fig.add\_axes((0.2, 0.2, 0.5, 0.5))

ax.plot(x\_data, y\_data, '-o', ms=8)

ax.yaxis.set\_major\_formatter(FormatStrFormatter(r'$\yen%1.1f$'))

plt.xticks(x\_data, day\_name[0:7], rotation=20)

plt.tick\_params(direction='in', width=2)

# 隐藏上轴脊和右轴脊

ax.spines['top'].set\_color('none')

ax.spines['right'].set\_color('none')

plt.show()

# 第7章 绘制3D图表和统计地图

1. **填空题**
2. 3D图表
3. 函数
4. cyl
5. Basemap
6. **判断题**
7. ×
8. ×
9. √
10. **选择题**
11. B
12. C
13. D
14. B
15. A
16. **简答题**
17. 答案：FuncAnimation是基于函数的动画类，它通过重复地调用同一函数来制作动画；ArtistAnimation是基于一组Artist对象的动画类，它通过一帧一帧的数据制作动画。
18. 答案：（1）创建Basemap类的对象，指定地图投影的类型和要处理的地球区域；（2）通过Basemap类的方法绘制地图背景；（3）通过Basemap类的方法在地图上绘制数据。
19. **编程题**
20. 答案：

import numpy as np

import matplotlib.pyplot as plt

import matplotlib.animation as animation

# 指定渲染环境

%matplotlib auto

def update\_points(num):

'''更新数据点'''

point\_ani.set\_data(x[num], y[num])

text\_pt.set\_text("x=%.3f, y=%.3f"%(x[num], y[num]))

return point\_ani, text\_pt

x = np.linspace(0, 2\*np.pi, 100)

y = np.sin(x)

fig = plt.figure(tight\_layout=True)

plt.plot(x, y)

point\_ani, = plt.plot(x[0], y[0], "ro")

text\_pt = plt.text(4, 0.9, '', fontsize=16)

ani = animation.FuncAnimation(fig, update\_points, np.arange(0, 100),

interval=100, blit=True)

plt.show()

# 第8章 使用matplotlib绘制高级图表

1. **填空题**
2. 变化
3. 树状图
4. 年龄
5. contour()
6. 标记符号
7. **判断题**
8. √
9. ×
10. ×
11. √
12. √
13. **选择题**
14. C
15. C
16. B
17. A
18. A
19. **简答题**

答：（1）使用构造方法Sankey()创建桑基图；（2）调用add()方法为桑基图添加诸如数据流量、标签等选项；（3）调用finish()方法完成绘制。

1. **简答题**
2. 答案：

import numpy as np

import matplotlib.pyplot as plt

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

plt.rcParams['axes.unicode\_minus'] = False

x = np.arange(1, 11)

y = np.array([62253, 51255, 34541, 28733, 17073,

9000, 5963, 2041, 1879, 1681])

labels = np.array(['武磊登上电影频道', '京奥运会海报', '浓眉哥受伤',

'安东尼准绝杀', '湖人单场20记盖帽', '第77届金球奖红毯',

'孟非大赞武磊', '李铁上任', '活塞vs湖人', '英超'])

fig = plt.figure(figsize=(10, 6), dpi= 80)

ax = fig.add\_subplot(111)

# 绘制棉棒图

markerline, stemlines, baseline = ax.stem(x, y, linefmt='--',

markerfmt='o', label='TestStem', use\_line\_collection=True)

# 设置棉棒图线段的属性

plt.setp(stemlines, lw=1)

ax.set\_ylabel('搜索指数')

ax.set\_xticks(x)

ax.set\_xticklabels(labels, rotation=60)

for temp\_x, temp\_y in zip(x, y):

ax.text(temp\_x, temp\_y+0.5, s='{}'.format(temp\_y), ha= 'center',

va='bottom', fontsize=14)

plt.show()

1. 答案：

import matplotlib.pyplot as plt

from matplotlib.sankey import Sankey

plt.rcParams["font.sans-serif"] = ["SimHei"]

plt.rcParams["axes.unicode\_minus"] = False

# 消费收入与支出数据

flows = [-0.1, -0.25, -0.2, -0.05, -0.025, 1, 0.025, -0.01]

# 流的标签列表

labels = ["旅行", "深造", "生活", "购物", "聚餐", "收入", "人情往来", "其它"]

# 流的方向

orientations = [1, 1, -1, -1, 1, 0, 1, -1]

# 创建Sankey类对象

sankey = Sankey()

# 为桑基图添加数据

sankey.add(flows=flows, # 收入与支出数据

labels=labels, # 数据标签

orientations=orientations, # 标签显示的位置

color="black", # 边缘线条颜色

fc="lightgreen", # 填充颜色

patchlabel="生活消费", # 图表中心的标签

alpha=0.7) # 透明度

# 桑基图绘制完成的对象

diagrams = sankey.finish()

diagrams[0].texts[4].set\_color("r") # 将下标为4的数据标签设为红色

diagrams[0].texts[4].set\_weight("bold") # 将下标为4的数据标签设为字体加粗

diagrams[0].text.set\_fontsize(20) # 将中心标签的字体大小设为20

diagrams[0].text.set\_fontweight("bold") # 将中心标签的字体设为加粗

plt.title("日常生活的开支流量图")

plt.show()

# 第9章 可视化后起之秀——pyecharts

1. **填空题**
2. Echarts
3. 同一对象
4. 渲染
5. 选项卡
6. ThemeType
7. **判断题**
8. √
9. ×
10. ×
11. √
12. √
13. **选择题**
14. C
15. D
16. A
17. B
18. A
19. **简答题**
20. 答案：（1）简洁的API令开发者使用起来如丝滑般流畅，且支持链式调用；（2）程序可运行到主流的Jupyter Notebook或JupyterLab工具；（3）程序可以轻松地集成至 Flask、Sanic、Django 等主流的Web框架中；（4）灵活的配置项可以轻松地搭配出精美的图表；（5）详细的文档和示例可以帮助开发者快速地上手项目；（6）多达400多个地图文件、原生百度地图为地理数据可视化提供强有力的支撑。
21. （1）创建与图表对应类的对象；（2）添加图表数据；（3）添加图表系列配置项；（4）添加图表全局配置项；（5）渲染图表。
22. **编程题**
23. 答案：

from pyecharts import options as opts

from pyecharts.charts import Bar

from pyecharts.globals import ThemeType

times = list(range(2009, 2019))

bar\_demo = (

Bar(init\_opts=opts.InitOpts(theme=ThemeType.ROMANTIC))

.add\_xaxis(times)

.add\_yaxis("", [3095, 4245, 6673, 10701, 13642, 31368,

40949, 41776, 56213, 64143])

.set\_global\_opts(title\_opts=opts.TitleOpts(

title="虎扑社区用户注册时间分布"),

yaxis\_opts=opts.AxisOpts(name="注册人数（个）",

name\_location="center", name\_gap=60),

xaxis\_opts=opts.AxisOpts(name="注册时间（年）",

name\_location="center", name\_gap=30))

)

bar\_demo.render\_notebook()

1. 答案：

from pyecharts import options as opts

from pyecharts.charts import PictorialBar

from pyecharts.globals import SymbolType

location = ["男", "女"]

values = [4.5, 95.4]

pict\_demo = (

PictorialBar()

.add\_xaxis(location)

.add\_yaxis(

"",

values,

label\_opts=opts.LabelOpts(is\_show=False),

symbol\_size=18,

symbol\_repeat="fixed",

symbol\_offset=[0, 0],

is\_symbol\_clip=True,

symbol=SymbolType.ROUND\_RECT,

)

.reversal\_axis()

.set\_global\_opts(

title\_opts=opts.TitleOpts(title="虎扑社区男用户与女用户的比例"),

xaxis\_opts=opts.AxisOpts(is\_show=False),

yaxis\_opts=opts.AxisOpts(

axistick\_opts=opts.AxisTickOpts(is\_show=False),

axisline\_opts=opts.AxisLineOpts(

linestyle\_opts=opts.LineStyleOpts(opacity=0)

),

),

)

)

pict\_demo.render\_notebook()