

### 学习目标

- ★ 认识图表常用的辅助元素
- ★ 掌握坐标轴的定制方法，包括设置坐标轴的标签、刻度范围和刻度标签
- ★ 掌握标题与图例的定制方法，能够为图表添加标题和图例
- ★ 掌握网格的定制方法，包括显示网格及设置网格的样式
- ★ 掌握参考线和参考区域的定制方法，能够为图表添加参考线和参考区域
- ★ 掌握注释文本的定制方法，包括为图表添加指向型和无指向型的注释文本
- ★ 掌握表格的定制方法，能够为图表添加表格

第2章使用matplotlib绘制了一些简单的图表，并通过这些图表直观地展示了数据，但图表中的矩形条因缺少数值标注而无法知道准确的数据等。因此，需要添加一些辅助元素来速且正确地理解图表。本章将对图表辅助元素的定制进行详细介绍。

### 3.1 认识图表常用的辅助元素

图表的辅助元素是指除根据数据绘制的图形之外的元素，常用的辅助元素包括坐标轴、标题、图例、网格、参考线、参考区域、注释文本和表格，它们都可以对图形进行补充说明。为了便于理解，下面以折线图为例介绍图表常用的辅助元素，如图3-1所示。

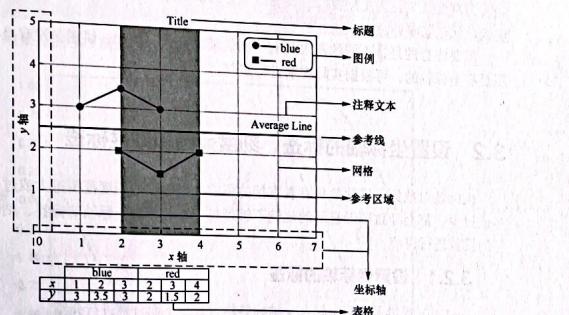


图3-1 图表常用的辅助元素

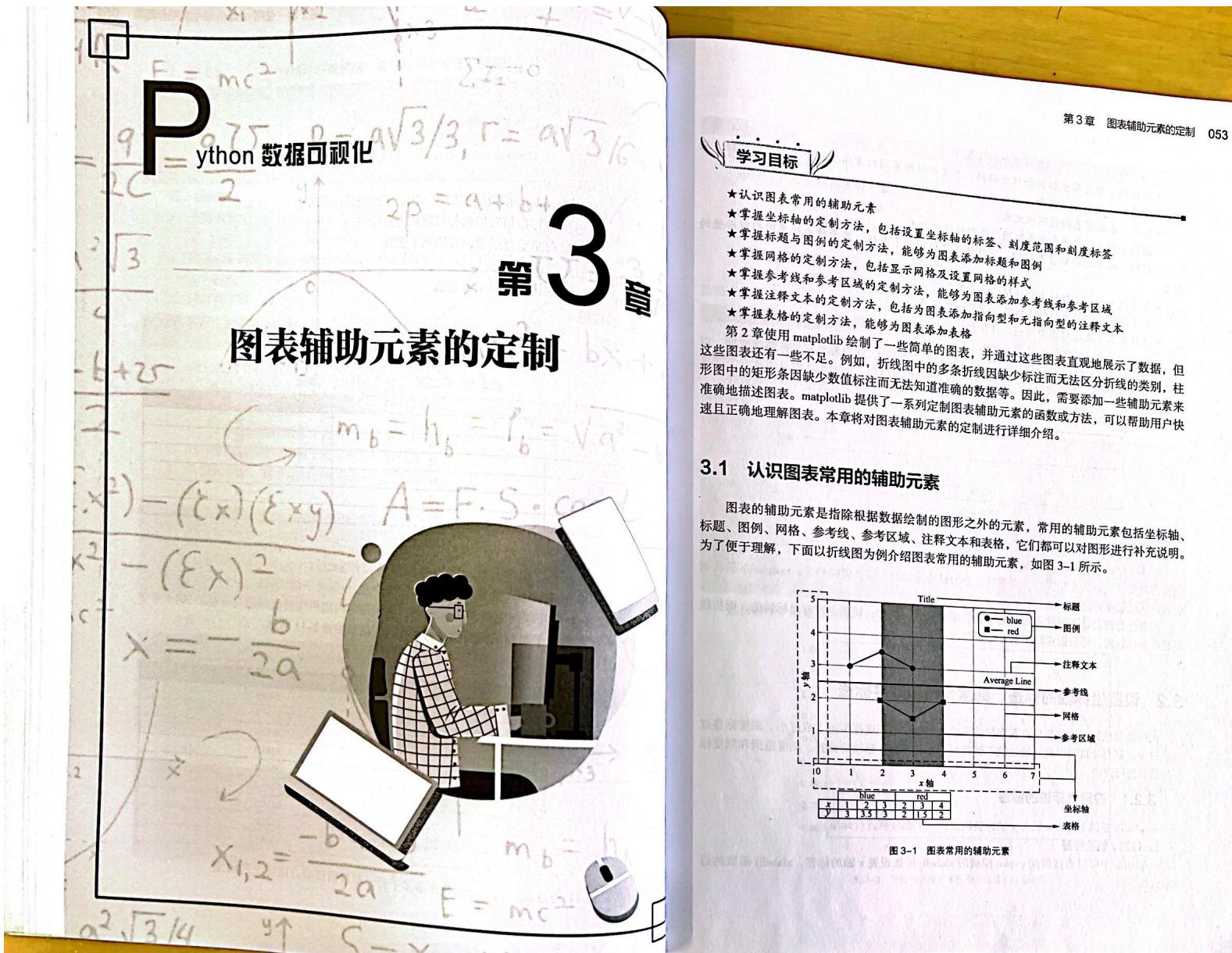


图 3-1 中的图表常用辅助元素的说明如下。

- 坐标轴：分为单坐标轴和双坐标轴，单坐标轴按不同的方向又可分为水平坐标轴（又称 x 轴）和垂直坐标轴（又称 y 轴）。
- 标题：表示图表的说明性文本。
- 图例：用于指出图表中各组图形采用的标识方式。
- 网格：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：标记坐标轴上特殊值的一条直线。
- 参考区域：标记坐标轴上特殊范围的一块区域。
- 注释文本：表示对图形的一些注释和说明。
- 表格：用于强调比较难理解数据的表格。

坐标轴是由刻度标签、刻度线（主刻度线和次刻度线）、轴脊和坐标轴标签组成的。以图 3-1 的 x 轴为例，下面通过一张图来描述坐标轴的完整结构，如图 3-2 所示。

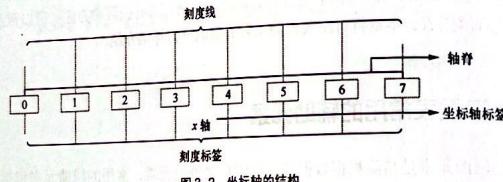


图 3-2 坐标轴的结构

图 3-2 中，“x 轴”为坐标轴的标签，“0”~“7”均为刻度标签，“0”~“7”对应的短竖线为刻度线，且为主刻度线，刻度线上方的横线为轴脊。需要注意的是，matplotlib 的次刻度线默认是隐藏的。

需要注意的是，不同的图表具有不同的辅助元素。例如，饼图是没有坐标轴的，而折线图是有坐标轴的，可根据实际需求进行定制。

## 3.2 设置坐标轴的标签、刻度范围和刻度标签

坐标轴对数据可视化效果有着直接的影响。坐标轴的刻度范围过大或过小、刻度标签过多或过少，都会导致图形显示的比例不够理想。本节将对坐标轴的标签、刻度范围和刻度标签的设置进行讲解。

### 3.2.1 设置坐标轴的标签

matplotlib 提供了设置 x 轴和 y 轴标签的方式，下面分别进行介绍。

1. 设置 x 轴的标签  
matplotlib 可以直接使用 pyplot 模块的 xlabel() 函数设置 x 轴的标签， xlabel() 函数的语法格式如下所示：

```
xlabel(xlabel, fontdict=None, labelpad=None, **kwargs)
```

该函数各参数含义如下。

- xlabel：表示 x 轴标签的文本。
- fontdict：表示控制标签文本样式的字典。
- labelpad：表示标签与坐标轴边框（包括刻度和刻度标签）的距离。

#### 2. 设置 y 轴的标签

matplotlib 中可以直接使用 pyplot 模块的 ylabel() 函数设置 y 轴的标签， ylabel() 函数的语法格式如下所示：

```
ylabel(ylabel, fontdict=None, labelpad=None, **kwargs)
```

该函数的 ylabel 参数表示 y 轴标签的文本，其余参数与 xlabel() 函数的参数的含义相同，此处不再赘述。此外，Axes 对象使用 set\_ylabel() 方法也可以设置 y 轴的标签。

假设现在有一个包含正弦曲线和余弦曲线的图表，该图表中设置 x 轴和 y 轴的标签，具体代码如下。

```
In [1]:
import numpy as np
import matplotlib.pyplot as plt
plt.rcParams['font.sans-serif'] = ['SimHei']
plt.rcParams['axes.unicode_minus'] = False
x = np.linspace(-np.pi, np.pi, 256, endpoint=True)
y1, y2 = np.sin(x), np.cos(x)
plt.plot(x, y1, x, y2)
# 设置 x 轴和 y 轴的标签
plt.xlabel("x 轴")
plt.ylabel("y 轴")
plt.show()
```

运行程序，效果如图 3-3 所示。

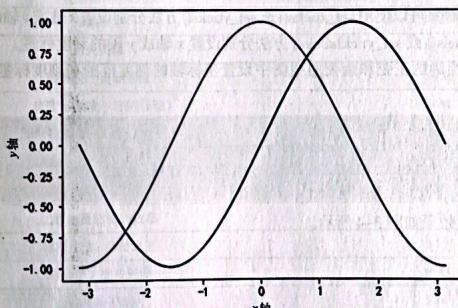


图 3-3 正弦和余弦曲线图——设置坐标轴标签

### 3.2.2 设置刻度范围和刻度标签

当绘制图表时，坐标轴的刻度范围和刻度标签都与数据的分布有着直接的联系，即坐标轴的刻度范围取决于数据的最大值和最小值。在使用 matplotlib 绘图时若没有指定任何数据，x 轴和 y 轴的范围均为  $0.05 \sim 1.05$ ，刻度标签均为  $[-0.2, 0.0, 0.2, 0.4, 0.6, 0.8, 1.0, 1.2]$ ；若指定了 x 轴和 y 轴的数据，刻度范围和刻度标签会随着数据的变化而变化。matplotlib 提供了重新设置坐标轴的刻度范围和刻度标签的方式，下面分别进行介绍。

#### 1. 设置刻度范围

使用 pyplot 模块的 `xlim()` 和 `ylim()` 函数分别可以设置或获取 x 轴和 y 轴的刻度范围。

`xlim()` 函数的语法格式如下所示：

```
xlim(left=None, right=None, emit=True, auto=False, *, xmin=None,
     xmax=None)
```

该函数常用参数的含义如下。

- `left`：表示 x 轴刻度取值区间的左位数。
  - `right`：表示 x 轴刻度取值区间的右位数。
  - `emit`：表示是否通知限制变化的观察者，默认为 True。
  - `auto`：表示是否允许自动缩放 x 轴，默认为 True。
  - `xmin`：表示 x 轴刻度的最小值。
  - `xmax`：表示 x 轴刻度的最大值。
- 此外，`Axes` 对象可以使用 `set_xlim()` 和 `set_ylim()` 方法分别设置 x 轴和 y 轴的刻度范围。

#### 2. 设置刻度标签

使用 pyplot 模块的 `xticks()` 和 `yticks()` 函数分别可以设置或获取 x 轴和 y 轴的刻度线位置和刻度标签。`xticks()` 函数的语法格式如下所示：

```
xticks(ticks=None, labels=None, **kwargs)
```

该函数的 `ticks` 参数表示刻度显示的位置列表，它还可以设为空列表，以此禁用 x 轴的刻度；`labels` 表示指定位置刻度的标签列表。

此外，`Axes` 对象可以使用 `set_xticks()` 或 `set_yticks()` 方法分别设置 x 轴或 y 轴的刻度线位置，使用 `set_xticklabels()` 或 `set_yticklabels()` 方法分别设置 x 轴或 y 轴的刻度标签。

在 3.2.1 节绘制的正弦和余弦曲线图中设置坐标轴的刻度范围和刻度标签，增加的代码如下。

```
# 设置 x 轴的刻度范围和刻度标签
plt.xlim(x.min() * 1.5, x.max() * 1.5)
plt.xticks([-np.pi, -np.pi/2, 0, np.pi/2, np.pi], [r'$-\pi$', r'$-\pi/2$', r'$0$', r'$\pi/2$', r'$\pi$'])
```

运行程序，效果如图 3-4 所示。

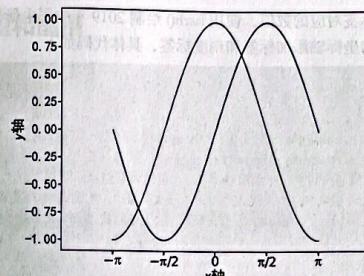


图 3-4 正弦和余弦曲线图——设置刻度范围和刻度标签

### 3.2.3 实例 1：2019 年中国电影票房排行榜

假如你有一段闲暇时间，到影院观影会是个不错的选项。如今，看电影已经成为人们休闲娱乐的方式之一，它不仅是一种视觉享受，而且是一场精神盛宴，使人们放松身心。2019 年中国上映了众多口碑不错的电影，对每部电影的总票房进行统计后，得出 2019 年中国电影票房排行榜 Top15，如表 3-1 所示。

表 3-1 2019 年中国电影票房排行榜 Top15

电影名称	总票房（亿元）
哪吒之魔童降世	48.57
流浪地球	46.18
复仇者联盟 4：终局之战	42.05
疯狂的外星人	21.83
飞驰人生	17.03
烈火英雄	16.70
蜘蛛侠：英雄远征	14.01
速度与激情：特别行动	13.84
扫毒 2：天地对决	12.85
大黄蜂	11.38
惊奇队长	10.25
比悲伤更悲伤的故事	9.46
哥斯拉 2：怪兽之王	9.27
阿丽塔：战斗天使	8.88
银河补习班	8.64

根据表 3-1 的数据，将“电影名称”一列的数据作为 y 轴的刻度标签，将“总票房（亿元）”

一列的数据作为刻度标签对应的数值，使用 barh() 绘制 2019 年中国电影票房排行榜 Top15 的条形图，并为条形图的坐标轴添加标签和刻度标签，具体代码如下。

```
In [2]:
# 01_film_rankings
import matplotlib.pyplot as plt
plt.rcParams['font.sans-serif'] = ['SimHei']
plt.rcParams['axes.unicode_minus'] = False
labels = ['哪吒之魔童降世', '流浪地球', '复仇者联盟4：终局之战',
          '疯狂的外星人', '飞驰人生', '烈火英雄', '蜘蛛侠：英雄远征',
          '速度与激情：特别行动', '扫毒2：天地对决', '大黄蜂', '惊奇队长',
          '比悲伤更悲伤的故事', '哥斯拉2：怪兽之王', '阿丽塔：战斗天使',
          '银河补习班']
bar_width = [48.57, 46.18, 42.05, 21.83, 17.03, 16.70, 14.01, 13.84,
             12.85, 11.38, 10.25, 9.46, 9.27, 8.88, 8.64]
y_data = range(len(labels))
fig = plt.figure()
ax = fig.add_subplot(111)
ax.barh(y_data, bar_width, height=0.2, color='orange')
# 设置x轴和y轴的标签
ax.set_xlabel("总票房(亿元)")
ax.set_ylabel("电影名称")
# 设置y轴的刻度线位置，刻度标签
ax.set_yticks(y_data)
ax.set_yticklabels(labels)
plt.show()
```

运行程序，效果如图 3-5 所示。

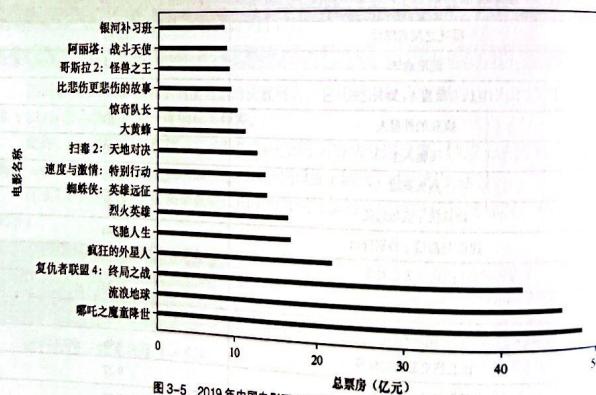


图 3-5 2019 年中国电影票房排行榜 Top15 的条形图

图 3-5 中，x 轴的标签位于底部，y 轴的标签位于左侧。由图 3-5 可知，电影《哪吒之魔童降世》的总票房最高，《流浪地球》的总票房排第二，《复仇者联盟4：终局之战》的总票房排第三。

### 3.3 添加标题和图例

图表的标题代表图表名称，一般位于图表的顶部且与图表居中对齐，可以迅速地让读者理解图表要说明的内容。matplotlib 中可以直接使用 pyplot 模块的 title() 函数添加图表标题，title() 函数的语法格式如下所示：

```
title(label, fontdict=None, loc='center', pad=None, **kwargs)
```

该函数常用参数的含义如下。

- label：表示标题的文本。
- fontdict：表示控制标题文本样式的字典。
- loc：表示标题的对齐样式，包括 'left'、'right' 和 'center' 三种取值，默认取值为 'center'，即居中显示标题。
- pad：表示标题与图表顶部的距离，默认为 None。

此外，Axes 对象还可以使用 set\_title() 方法添加图表的标题。

在 3.2.2 节绘制的正弦和余弦曲线图中添加标题“正弦曲线和余弦曲线”，增加的代码如下。

```
# 添加标题
plt.title("正弦曲线和余弦曲线")
```

运行程序，效果如图 3-6 所示。

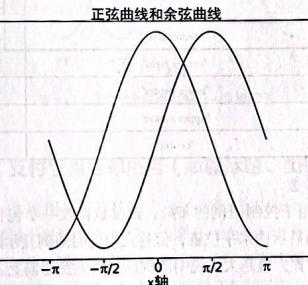


图 3-6 正弦和余弦曲线图——添加标题

### 3.3.2 添加图例

图例是一个列举各组图形数据标识方式的方框图，它由图例标识和图例项两个部分构成，其中图例标识是代表各组图形的图案；图例项是与图例标识对应的名称（说明文本）。当 matplotlib 绘制包含多组图形的图表时，可以在图表中添加图例，帮助用户明确每组图形代表

matplotlib 中可以直接使用 pyplot 模块的 legend() 函数添加图例，legend() 函数的语法格式

如下所示：

```
legend(handles, labels, loc, bbox_to_anchor, ncol, title, shadow,
       fancybox, *args, **kwargs)
```

该函数常用参数的介绍如下。

#### (1) handles 和 labels 参数

handles 参数表示由图形标识构成的列表，labels 参数表示由图例项构成的列表。需要注意的是，handles 和 labels 参数应接收相同长度的列表，若接收的列表长度不同，则会对较长的列表进行截断处理，使较长列表与较短列表长度相等。

#### (2) loc 参数

loc 参数用于控制图例在图表中的位置，该参数支持字符串和数值两种形式的取值，每种取值及其对应的图例位置的说明如表 3-2 所示。

表 3-2 loc 参数的取值及其对应的图例位置

位置编码	位置字符串	说明
0	'best'	自适应
1	'upper right'	右上方
2	'upper left'	左上方
3	'lower left'	左下方
4	'lower right'	右下方
5	'right'	右方
6	'center left'	中心偏左
7	'center right'	中心偏右
8	'lower center'	中心偏下
9	'upper center'	中心偏上
10	'center'	居中

#### (3) bbox\_to\_anchor 参数

bbox\_to\_anchor 参数用于控制图例的布局，该参数接收一个包含两个数值的元组，其中第一个数值用于控制图例显示的水平位置，值越大则说明图例显示的位置越偏右；第二个数值用于控制图例的垂直位置，值越大则说明图例显示的位置越偏上。

#### (4) ncol 参数

ncol 参数表示图例的列数，默认值为 1。

#### (5) title 参数

title 参数表示图例的标题，默认值为 None。

#### (6) shadow 参数

shadow 参数控制是否在图例后面显示阴影，默认值为 None。

#### (7) fancybox 参数

fancybox 参数控制是否为图例设置圆角边框，默认值为 None。

若使用 pyplot 绘图函数绘图时已经预先通过 label 参数指定了显示于图例的标签，则后续可以直接调用 legend() 函数添加图例；若未预先指定应用于图例的标签，则后续在调用 legend() 函数时为参数 handles 和 labels 传值即可，示例代码如下。

```
ax.plot([1, 2, 3], label='Inline label')
ax.legend()
# 或
ax.legend((line1, line2, line3), ('label1', 'label2', 'label3'))
```

在 3.3.1 节绘制的正弦和余弦曲线图中添加图例，增加的代码如下。

```
lines = plt.plot(x, y1, x, y2)
# 添加图例
plt.legend(lines, ['正弦', '余弦'], shadow=True, fancybox=True)
```

运行程序，效果如图 3-7 所示。

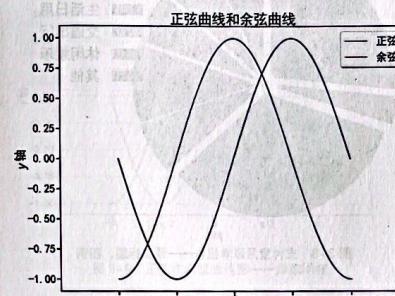


图 3-7 正弦和余弦曲线图——添加图例

### 3.3.3 实例 2：支付宝账单报告（添加标题、图例）

图例常见于饼图中，主要用于标注饼图中每个扇形代表的含义。2.6.2 节的用户 A 某月支付宝账单报告的饼图将每个扇形的含义标注到圆外，由于标注的文字长短不一且扇形数量偏多，导致图表显得比较杂乱，因此将饼图中全部的标注文字移到图例中，具体代码如下。

```
In [3]:
# 02_monthly_bills_of_alipay
import matplotlib.pyplot as plt
plt.rcParams['font.sans-serif'] = ['SimHei']
plt.rcParams['axes.unicode_minus'] = False
kinds = ['购物', '人情往来', '餐饮美食', '通信物流', '生活日用',
         '交通出行', '休闲娱乐', '其他']
money_scale = [800 / 3000, 100 / 3000, 1000 / 3000, 200 / 3000,
               300 / 3000, 200 / 3000, 200 / 3000, 200 / 3000]
dev_position = [0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1]
```

```

plt.pie(money_scale, autopct='%.1f%%', shadow=True,
        explode=dev_position, startangle=90)
# 添加标题
plt.title('支付宝月账单报告')
# 添加图例
plt.legend(kinds, loc='upper right', bbox_to_anchor=[1.3, 1.1])
plt.show()
    
```

运行程序，效果如图 3-8 所示。

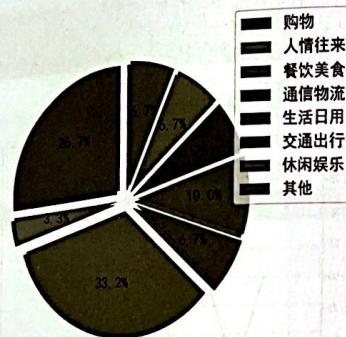


图 3-8 支付宝月账单报告——添加标题、图例

图 3-8 中，标题位于图表顶部且与图表居中对齐，图例位于图表的右上方。与图 2-11 相比，图 3-8 增加了标题和图例，有助于用户明确饼图及其每个颜色的扇形的含义。

## 3.4 显示网格

### 3.4.1 显示指定样式的网格

网格是从刻度线开始延伸，贯穿至整个绘图区域的辅助线条，它能帮助人们轻松地查看图形的数值。网格按不同的方向可以分为垂直网格和水平网格，这两种网格既可以单独使用，也可以同时使用，常见于添加图表精度、分辨图形细微差别的场景。

matplotlib 中可以直接使用 pyplot 模块的 grid() 函数显示网格，grid() 函数的语法格式如下所示：

```
grid(b=None, which='major', axis='both', **kwargs)
```

该函数常用参数的含义如下。

- b：表示是否显示网格。若提供其他关键字参数，则 b 参数设为 True。

- which：表示显示网格的类型，支持 major、minor、both 这 3 种类型，默认为 major。
- axis：表示显示哪个方向的网格，该参数支持 both、x 和 y 这 3 个选项，默认为 both。
- linewidth 或 lw：表示网格线的宽度。

此外，还可以使用 Axes 对象的 grid() 方法显示网格。需要说明的是，坐标轴若没有刻度，就无法显示网格。

在 3.3.2 节绘制的正弦和余弦曲线图中显示水平网格，增加的代码如下。

```
# 显示网格
plt.grid(b=True, axis='y', linewidth=0.3)
```

运行程序，效果如图 3-9 所示。

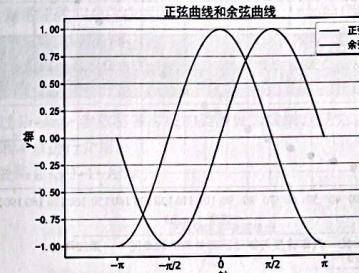


图 3-9 正弦和余弦曲线图——添加网格

### 3.4.2 实例 3：汽车速度与制动距离的关系（添加网格）

在 2.7.2 节的汽车速度与制动距离关系的散点图中，很多圆点因距离坐标轴较远而无法准确地看出数值。因此，本实例将在散点图中显示网格，并调整坐标轴的刻度，具体代码如下。

```
In [4]:
# 03_vehicle_speed_and_braking_distance
import numpy as np
import matplotlib.pyplot as plt
plt.rcParams['font.sans-serif'] = 'SimHei'
plt.rcParams['axes.unicode_minus'] = False
x_speed = np.arange(10, 210, 10)
y_distance = np.array([0.5, 2.0, 4.4, 7.9, 12.3,
                      17.7, 24.1, 31.5, 39.9, 49.2,
                      59.5, 70.8, 83.1, 96.4, 110.7,
                      126.0, 142.2, 159.4, 177.6, 196.8])
plt.scatter(x_speed, y_distance, s=50, alpha=0.9, linewidths=0.3)
# 设置 x 轴的标签、刻度标签
plt.xlabel('速度 (km/h)')
```

```
plt.ylabel('制动距离 (m)')
plt.xticks(x_speed)
# 显示网格
plt.grid(b=True, linewidth=0.3)
plt.show()
```

运行程序，效果如图 3-10 所示。

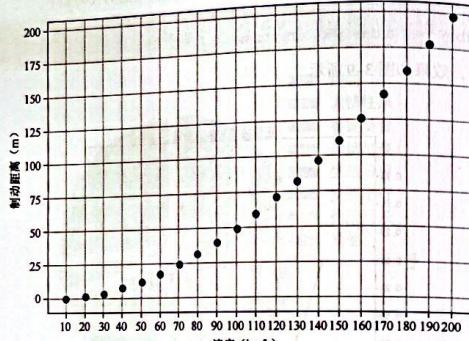


图 3-10 汽车速度与制动距离关系的散点图——添加网格

与图 2-19 相比，图 3-10 的散点图增加了坐标轴标签、浅灰色的网格，有助于用户大致了解各数据点对应的数值。

### 3.5 添加参考线和参考区域

#### 3.5.1 添加参考线

参考线是一条或多条贯穿绘图区域的线条，用于为绘图区域中图形数据之间的比较提供参考依据，常见于财务分析、经营分析中，例如目标线、平均线、预算线等。参考线按方向的不同可分为水平参考线和垂直参考线。matplotlib 中提供了 `axhline()` 和 `axvline()` 函数，分别用于添加水平参考线和垂直参考线，具体介绍如下。

##### 1. 使用 `axhline()` 绘制水平参考线

`axhline()` 函数的语法格式如下所示：

```
axhline(y=0, xmin=0, xmax=1, linestyle='--', **kwargs)
```

该函数常用参数的含义如下。

- `y`：表示水平参考线的纵坐标。

- `xmin`：表示水平参考线的起始位置，默认为 0。

- `xmax`：表示水平参考线的终止位置，默认为 1。
- `linestyle`：表示水平参考线的类型，默认为实线。

##### 2. 使用 `axvline()` 绘制垂直参考线

`axvline()` 函数的语法格式如下所示：

```
axvline(x=0, ymin=0, ymax=1, linestyle='--', **kwargs)
```

该函数常用参数的含义如下。

- `x`：表示垂直参考线的横坐标。

- `ymin`：表示垂直参考线的起始位置，默认为 0。

- `ymax`：表示垂直参考线的终止位置，默认为 1。

- `linestyle`：表示垂直参考线的类型，默认为实线。

在 3.4.1 节绘制的正弦和余弦曲线图中添加参考线，增加的代码如下。

```
# 添加参考线
plt.axhline(x=0, linestyle='--')
plt.axvline(y=0, linestyle='--')
```

上述代码通过 `linestyle` 参数将参考线的类型设为虚线，避免了参考线与曲线混淆，关于线条的类型会在第 4 章进行介绍。

运行程序，效果如图 3-11 所示。

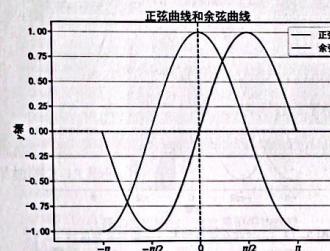


图 3-11 正弦和余弦曲线图——添加参考线

#### 3.5.2 添加参考区域

pyplot 模块中提供了 `axhspan()` 和 `axvspan()` 函数，分别用于为图表添加水平参考区域和垂直参考区域，具体介绍如下。

##### 1. 使用 `axhspan()` 绘制水平参考区域

`axhspan()` 函数的语法格式如下所示：

```
axhspan(ymin, ymax, xmin=0, xmax=1, **kwargs)
```

该函数常用参数的含义如下。

- `ymin`：表示水平跨度的下限，以数据为单位。
- `ymax`：表示水平跨度的上限，以数据为单位。
- `xmin`：表示垂直跨度的下限，以轴为单位，默认为 0。
- `xmax`：表示垂直跨度的上限，以轴为单位，默认为 1。

## 2. 使用 `axvspan()` 绘制垂直参考区域

`axvspan()` 函数的语法格式如下所示：

```
axvspan(xmin, xmax, ymin=0, ymax=1, **kwargs)
```

该函数常用参数的含义如下。

- `xmin`：表示垂直跨度的下限。
- `xmax`：表示垂直跨度的上限。

在 3.5.1 节绘制的正弦和余弦曲线图中添加参考区域，增加的代码如下。

```
# 添加参考区域
plt.axvspan(xmin=0.5, xmax=2.0, alpha=0.3)
plt.axvspan(ymin=0.5, ymax=1.0, alpha=0.3)
```

运行程序，效果如图 3-12 所示。

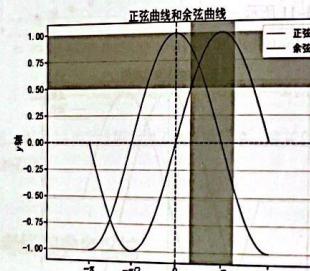


图 3-12 正弦和余弦曲线图——添加参考区域

### 3.5.3 实例 4：全校高二年级各班男女生英语成绩评估

某高中高二年级期中模拟考试后，学校对该年级各班各学科的平均成绩进行统计，计算出全体高二年级的英语平均成绩为 88.5，其中高二各班男生、女生的英语平均成绩如表 3-3 所示。

表 3-3 高二各班男生、女生英语平均成绩

班级名称	平均成绩（男生）	平均成绩（女生）
高二 1 班	90.5	92.7
高二 2 班	89.5	87.0
高二 3 班	88.7	90.5
高二 4 班	88.5	85.0

续表

班级名称	平均成绩（男生）	平均成绩（女生）
高二 5 班	85.2	89.5
高二 6 班	86.6	89.8

根据表 3-3 的数据，将“班级名称”一列的数据作为 x 轴的刻度标签，将“男生”和“女生”两列的数据作为刻度标签对应的数值，使用 `bar()` 绘制各班级男生、女生英语平均成绩的柱形图，并将高二年级的英语平均成绩作为参考线，比较哪些班级的英语成绩有待提高，具体代码如下。

```
In [5]:
# 04_average_score_of_english
import numpy as np
import matplotlib.pyplot as plt
plt.rcParams['font.sans-serif'] = ['SimHei']
plt.rcParams['axes.unicode_minus'] = False
men_means = (90.5, 89.5, 88.7, 88.5, 85.2, 86.6)
women_means = (92.7, 87.0, 90.5, 85.0, 89.5, 89.8)
ind = np.arange(len(men_means)) # 每组柱形的 x 位置
width = 0.2 # 各柱形的宽度
fig = plt.figure()
ax = fig.add_subplot(111)
ax.bar(ind - width / 2, men_means, width, label='男生平均成绩')
ax.bar(ind + 0.2, women_means, width, label='女生平均成绩')
ax.set_title('高二各班男生、女生英语平均成绩')
ax.set_ylabel('分数')
ax.set_xticks(ind)
ax.set_xticklabels(['高二 1 班', '高二 2 班', '高二 3 班', '高二 4 班',
                   '高二 5 班', '高二 6 班'])
# 添加参考线
ax.axhline(88.5, ls='--', linewidth=1.0, label='全体平均成绩')
ax.legend(loc="lower right")
plt.show()
```

运行程序，效果如图 3-13 所示。

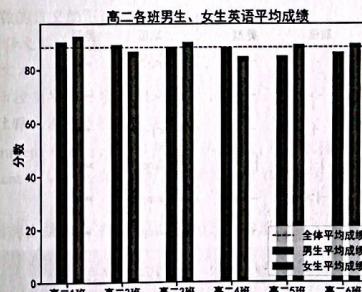


图 3-13 高二各班男生、女生英语平均成绩的柱形图

图 3-13 中, 蓝色的虚线代表高二年级的英语平均成绩。由图 3-13 可知, 高二 2 班、4 班女生和 5 班、6 班男生的平均成绩均低于高二年级的英语平均成绩。

### 3.6 添加注释文本

注释文本是图表的重要组成部分, 它能够对图形进行简短描述, 有助于用户理解图表。注释文本按注释对象的不同主要分为指向型注释文本和无指向型注释文本, 其中指向型注释文本一般是针对图表某一部分的特定说明, 无指向型注释文本一般是针对图表整体的特定说明。下面将介绍添加指向型注释文本和无指向型注释文本的方法。

#### 3.6.1 添加指向型注释文本

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。pyplot 模块中提供了 `annotate()` 函数为图表添加指向型注释文本, 该函数的语法格式如下所示:

```
annotate(s, xy, *args, **kwargs)
```

该函数常用参数的含义如下。

- `s`: 表示注释文本的内容。
- `xy`: 表示被注释的点的坐标位置, 接收元组 (`x,y`)。
- `xytext`: 表示注释文本所在的坐标位置, 接收元组 (`x,y`)。
- `xycoords`: 表示 `xy` 的坐标系统, 默认值为 “`data`”, 代表与折线图使用相同的坐标系统。
- `arrowprops`: 表示指示箭头的属性字典。
- `bbox`: 表示注释文本的边框属性字典。

`arrowprops` 参数接收一个包含若干键的字典, 通过向字典中添加键值来控制箭头的显示。常见的控制箭头的键包括 `width`、`headwidth`、`headlength`、`shrink`、`arrowstyle` 等, 其中键 `arrowstyle` 代表箭头的类型, 该键对应的值及对应的类型如图 3-14 所示。

取值	类型	取值	类型
-	—	<>	↔
->	→	< -	←
	—	< ->	↔
-	— -	fancy	→
->	→	simple	→
<	←	wedge	—

图 3-14 键 `arrowstyle` 的取值及对应的类型

在 3.5.2 节绘制的正弦和余弦曲线图中添加指向型注释文本, 增加的代码如下。

```
# 添加指向型注释文本
plt.annotate("最小值",
             xy=(-np.pi / 2, -1.0),
             xytext=(-np.pi / 2, -0.5),
             arrowprops=dict(arrowstyle="->"))
```

运行程序, 效果如图 3-15 所示。

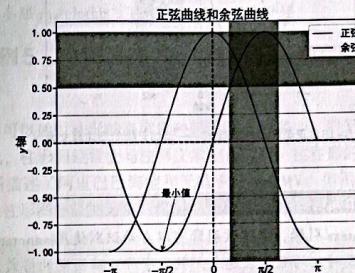


图 3-15 正弦和余弦曲线图——添加指向型注释文本

#### 3.6.2 添加无指向型注释文本

无指向型注释文本是指仅使用文字的注释方式对绘图区域的图形进行说明的文本。pyplot 模块中提供了 `text()` 函数为图表添加无指向型注释文本, 该函数的语法格式如下所示:

```
text(x, y, s, fontdict=None, withdash=<deprecated parameter>, **kwargs)
```

该函数常用参数的含义如下。

- `x, y`: 表示注释文本的位置。
- `s`: 表示注释文本的内容。
- `fontdict`: 表示控制字体的字典。
- `bbox`: 表示注释文本的边框属性字典。
- `horizontalalignment` 或 `ha`: 表示水平对齐的方式, 可以取值为 `center`、`right` 或 `left`。
- `verticalalignment` 或 `va`: 表示垂直对齐的方式, 可以取值为 `center`、`top`、`bottom`、`baseline` 或 `center_baseline`。

在 3.6.1 节绘制的正弦和余弦曲线图中添加无指向型注释文本, 增加的代码如下。

```
# 添加无指向型注释文本
plt.text(3.10, 0.10, "y=sin(x)", bbox=dict(alpha=0.2))
```

运行程序, 效果如图 3-16 所示。

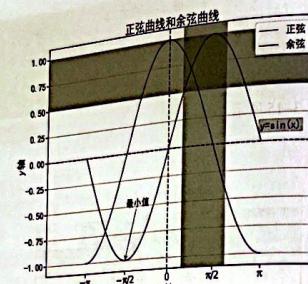


图 3-16 正弦和余弦曲线图——添加无指向型注释文本

**三多学一招：matplotlib 编写数学表达式**

matplotlib 中自带 mathtext 引擎，通过该引擎可以自动识别使用 `annotate()` 或 `text()` 函数传入的数学字符串，并解析成对应的数学表达式。数学字符串有固定的格式，它要求字符串以美元符号“\$”为首尾字符，且首尾字符中间为数学表达式，基本格式如下所示：

`r'$数学表达式$'`

为保证字符串中的所有字符能以字面的形式显示，数学字符串需要配合“`r`”使用。下面是使用 matplotlib 编写的一个简单的数学字符串：

`r'$\alpha > \beta$'`

以上字符串中“`\alpha`”和“`\beta`”对应常见的小写希腊字母  $\alpha$  和  $\beta$ ，其对应的数学表达式如下：

$\alpha > \beta$

此外，“`\alpha`”和“`\beta`”的后面还可以增加上标和下标，其中上标使用符号“`^`”表示，下标使用符号“`_`”表示。例如，将  $\alpha$  的下标设为  $i$ ，将  $\beta$  的下标设为  $j$  的示例如下：

`r'$\alpha_i > \beta_j$'`

以上示例对应的数学表达式如下：

matplotlib 中使用 “`\frac{1}{2}`” 可以编写分数形式的数字字符串，“`\frac`”后面的两个大括号分别代表分数的分子和分母，示例代码如下：

`r'$\frac{3}{4}$'`

以上示例对应的数学表达式如下：

$\frac{3}{4}$

此外，还可以编写分数嵌套的数学字符串，代码如下：

`r'$\frac{5 - \frac{1}{x}}{4}$'`

以上示例对应的数学表达式如下：

$$\frac{5 - \frac{1}{x}}{4}$$

更多内容可以参照 matplotlib 官网自学，此处不再赘述。

### 3.6.3 实例 5：2013—2019 财年阿里巴巴淘宝和天猫平台的 GMV(添加注释文本)

虽然柱形图中可以通过柱形的高度反映每组数据的多少，但是仍然无法让用户精准地知道具体数值。因此，柱形图经常会与注释文本配合使用，在柱形的顶部标注具体数值。2.2.2 节实例中的柱形图描述了阿里巴巴淘宝和天猫平台的 GMV，但图中的矩形条缺少具体的数值，因此这里将在柱形图中添加无指向型注释文本，代码如下。

```
In [6]:
# 05_taobao_and_tianmao_GMV
import matplotlib.pyplot as plt
import numpy as np
plt.rcParams['font.sans-serif'] = ['SimHei']
plt.rcParams['axes.unicode_minus'] = False
x = np.arange(1, 8)
y = np.array([10770, 16780, 24440, 30920, 37670, 48200, 57270])
bar_rects = plt.bar(x, y, tick_label=["FY2013", "FY2014", "FY2015",
                                      "FY2016", "FY2017", "FY2018", "FY2019"], width=0.5)

# 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() * 1.05 # 获取每个矩形条的高度
        plt.text(rect.get_x() + rect.get_width() / 2, height + 300,
                 s='{}'.format(height),
                 ha='center', va='bottom')
autolabel(bar_rects)
plt.ylabel('GMV(亿元)')
plt.show()
```

运行程序，效果如图 3-17 所示。

与图 2-6 相比，图 3-17 的柱形图增加了  $y$  轴的标签和注释文本，帮助用户准确地知道各柱形对应的数值。

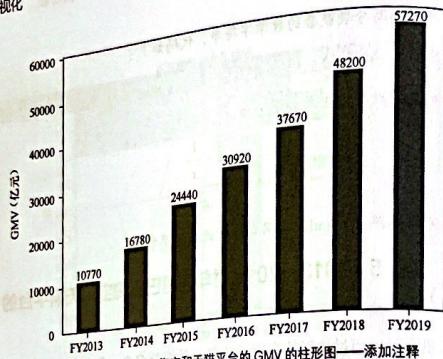


图 3-17 2013—2019 财年淘宝和天猫平台的 GMV 的柱形图——添加注释

### 3.7 添加表格

#### 3.7.1 添加自定义样式的表格

matplotlib 可以绘制各种各样的图表，以便用户发现数据间的规律。为了更加凸显数据间的规律与特点，便于用户从多元分析的角度深入挖掘数据潜在的含义，可将图表与数据表格结合使用，使用数据表格强调图表某部分的数值。matplotlib 中提供了为图表添加数据表格的函数 `table()`，该函数的语法格式如下所示：

```
table(cellText=None, cellColours=None, cellLoc='right', colWidths=None,
      rowLabels=None, rowColours=None, rowLoc='left', colLabels=None,
      colColours=None, colLoc='center', loc='bottom', bbox=None,
      edges='closed', **kwargs)
```

该函数常用参数表示的含义如下。

- `cellText`：表示表格单元格中的数据，是一个二维列表。
- `cellColours`：表示单元格的背景颜色。
- `cellLoc`：表示单元格文本的对齐方式，支持 `'left'`、`'right'` 和 `'center'` 三种取值，默认值为 `'right'`。
- `colWidths`：表示每列的宽度。
- `rowLabels`：表示行标题的文本。
- `rowColours`：表示行标题所在单元格的背景颜色。
- `rowLoc`：表示行标题的对齐方式。
- `colLabels`：表示列标题的文本。
- `colColours`：表示列标题所在单元格的背景颜色。
- `colLoc`：表示列标题的对齐方式。
- `loc`：表示表格与绘图区域的对齐方式。

此外，还可以使用 `Axes` 对象的 `table()` 方法为图表添加数据表格，此方法与 `table()` 函数的用法相似，此处不再赘述。请参考了书中第 3 章中有关如何为图表添加注释以及在 3.6.2 节绘制的正弦和余弦曲线图中添加数据表格，增加的代码如下。

```
# 添加表格
plt.table(cellText=[[6, 6, 6], [8, 8, 8]],
           colWidths=[0.1] * 3,
           rowLabels=['第1行', '第2行'],
           colLabels=['第1列', '第2列', '第3列'], loc='lower right')
```

运行程序，效果如图 3-18 所示。

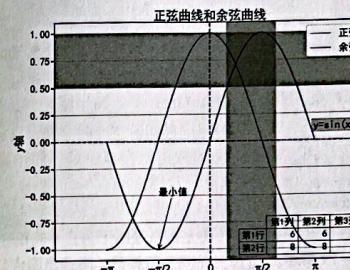


图 3-18 正弦和余弦曲线图——添加数据表格

#### 3.7.2 实例 6：果酱面包配料比例

美好的一天从早餐开始，果酱面包是常见的早餐且深受大家喜爱，无论是大人还是小孩都很爱吃。已知某果酱面包需要准备的配料如表 3-4 所示。

表 3-4 果酱面包配料表

单位：g

配料名称	重量
面粉	250
全麦粉	150
酵母	4
苹果酱	250
鸡蛋	50
黄油	30
盐	4
白糖	20

根据表 3-4 的数据，将“配料名称”一列的数据作为图例项，将“重量”一列的数

据与总重量的比例作为数据，使用 pie() 绘制果酱面包配料比例的饼图，并将各种配料的重量以数据表格的形式添加到图表中，方便用户了解各种配料的占比和重量，具体代码如下。

```
In [7]:
# 06_jam_bread_ingredients
import matplotlib.pyplot as plt
plt.rcParams['font.sans-serif'] = ['SimHei']
plt.rcParams['axes.unicode_minus'] = False
kinds = ['面粉', '全麦粉', '酵母', '苹果酱', '鸡蛋', '黄油', '盐', '白糖']
weight = [250, 150, 4, 250, 50, 30, 4, 20]
total_weight = 0
for i in weight:
    total_weight += i
batching_scale = [i / total_weight for i in weight]
plt.pie(batching_scale, autopct='%.1f%%')
plt.legend(kinds, loc='upper right', bbox_to_anchor=[1.1, 1.1])
# 添加表格
plt.table(cellText=[weight],
          cellLoc='center',
          rowLabels=['重量(g)'],
          colLabels=kinds,
          loc='lower center')
plt.show()
```

运行程序，效果如图 3-19 所示。

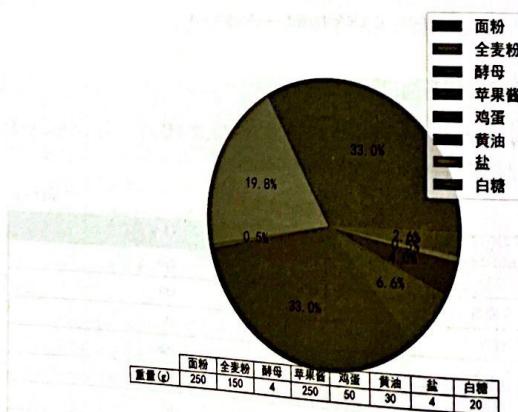


图 3-19 果酱面包配料的饼图

图 3-19 中，表格位于饼图的下方。由图 3-19 可知，蓝色和红色扇形的面积最大，说明苹果酱和面粉在果酱面包中占比最大，重量都为 250 g。

## 3.8 本章小结

本章主要介绍了图表辅助元素的定制，包括认识常用的辅助元素，设置坐标轴的标签，设置刻度范围和刻度标签，添加标题和图例，显示网格，添加参考线和参考区域，添加注释文本，添加表格。通过学习本章的内容，读者能熟悉常见图表辅助元素的用途和用法，可以为图表选择合适的辅助元素。

## 3.9 习题

### 一、填空题

- 图表的辅助元素是指除了根据数据绘制的\_\_\_\_\_之外的元素。
- 图例是一个列举图表中各组图形\_\_\_\_\_方式的方框图。
- 指向型注释文本是通过\_\_\_\_\_的注释方式对图形进行解释的文本。
- \_\_\_\_\_是标记坐标轴上特殊值的一条直线。
- matplotlib 自带的引擎可以自动识别数学字符串，并将该数学字符串解析成相应的\_\_\_\_\_。

### 二、判断题

- matplotlib 中图例一直位于图表的右上方，它的位置是不可变的。（ ）
- 参考线可以为图形数据与特殊值之间的比较提供参考。（ ）
- 坐标轴的标签代表图表名称，一般位于图表顶部居中的位置。（ ）
- 若坐标轴没有刻度，则无法显示网格。（ ）
- 坐标轴的刻度范围取决于数据的最大值和最小值。（ ）

### 三、选择题

- 关于图表的辅助元素，下列描述错误的是（ ）。
  - 标题一般位于图表的顶部中心，可以帮助用户理解图表要说明的内容
  - 参考区域是标记坐标轴上特殊值的一条直线
  - 图例由图例标识和图例项构成，可以帮助用户理解每组图形的含义
  - 表格主要用于强调比较难以理解的数据
- 下列函数中，可以设置坐标轴刻度标签的是（ ）。
  - xlim()
  - grid()
  - xticks()
  - axhline()
- 当使用 pyplot 的 legend() 函数添加图例时，可以通过以下哪个参数控制图例的列数？（ ）
  - loc
  - ncol
  - bbox\_to\_anchor
  - fancybox
- 下列选项中，可以为图表添加一条值为 1.5 的水平参考线的是（ ）。
  - plt.axhline(y=1.5, ls='--', linewidth=1.5)
  - plt.axhline(y=1, ls='--', linewidth=1.5)

