

3

Layout of A Figure

布局

各种子图布局方案



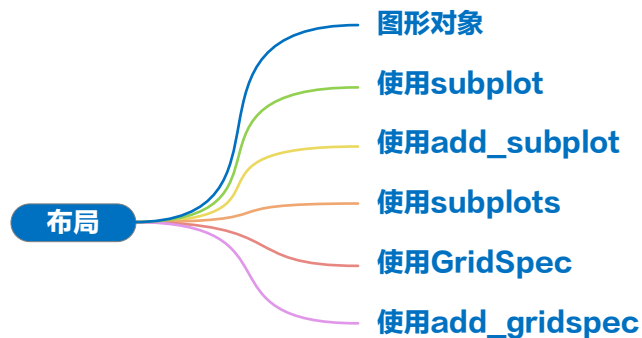
艺术洗涤心灵的浮尘。

Art washes away from the soul the dust of everyday life.

—— 毕加索 (Pablo Picasso) | 西班牙艺术家 | 1881 ~ 1973



- ◀ matplotlib.gridspec.GridSpec() 创建和配置复杂的子图网格布局，以便在一个图形窗口中放置多个子图
- ◀ matplotlib.gridspec.SubplotSpec 用于定义和控制子图在网格布局中的位置和大小
- ◀ matplotlib.pyplot.contour() 绘制等高线图
- ◀ matplotlib.pyplot.contourf() 绘制填充等高线图
- ◀ matplotlib.pyplot.figure() 创建一个新的图形窗口或图表对象，以便在其上进行绘图操作
- ◀ matplotlib.pyplot.rcParams 获取或设置全局绘图参数的默认值，如图形尺寸、字体大小、线条样式等
- ◀ matplotlib.pyplot.scatter() 绘制散点图
- ◀ matplotlib.pyplot.subplot() 用于在当前图形窗口中创建一个子图，并定位该子图在整个图形窗口中的位置
- ◀ matplotlib.pyplot.subplots() 一次性创建一个包含多个子图的图形窗口，并返回一个包含子图对象的元组
- ◀ numpy.linspace() 在指定的间隔内，返回固定步长的数据
- ◀ numpy.meshgrid() 产生网格化数据
- ◀ numpy.random.multivariate_normal() 用于生成多元正态分布的随机样本
- ◀ numpy.vstack() 返回竖直堆叠后的数组
- ◀ scipy.stats.gaussian_kde() 高斯核密度估计
- ◀ statsmodels.api.nonparametric.KDEUnivariate() 构造一元 KDE



本 PDF 文件为作者草稿，发布目的为方便读者在移动终端学习，终稿内容以清华大学出版社纸质出版物为准。

版权归清华大学出版社所有，请勿商用，引用请注明出处。

代码及 PDF 文件下载：<https://github.com/Visualize-ML>

本书配套微课视频均发布在 B 站——生姜 DrGinger：<https://space.bilibili.com/513194466>

欢迎大家批评指教，本书专属邮箱：jiang.visualize.ml@gmail.com

3.1 图形对象

本节先聊一聊图形对象基本规格。

大小尺寸

Matplotlib 中默认图片尺寸为：宽 6.4 英寸，高 4.8 英寸。1 英寸 (inch) 约为 2.54 厘米 (cm)。也就是说默认图片尺寸为，宽约 16 厘米，高约 12 厘米。可以使用 `figure` 函数，并指定 `figsize` 参数来设置图像的宽度和高度。`figsize` 参数接受一个元组 (宽度, 高度)，单位为英寸。

比如，`import matplotlib.pyplot as plt; plt.figure(figsize=(3, 3))` 将图像尺寸修改为 3 英寸 × 3 英寸。图 7 上图展示为 3 英寸 × 3 英寸图片真实大小。图 7 下图为换算为厘米的图片。

分辨率

默认图片以一个 dpi (dots per inch, 每英寸点数) 为 100 的分辨率显示。利用 `import matplotlib.pyplot as plt` 将绘图模块导入后，可以利用 `plt.rcParams['figure.dpi'] = 300` 将图像 dpi 提高到 300。

如果想保存图像到文件，可以使用 `savefig` 函数，并通过设置 `dpi` 参数来指定分辨率。例如，如果希望保存图像为 300dpi 的高质量 PNG 文件，可以用 `plt.savefig('plot_name.png', dpi=300)`。

对于 Seaborn，可以在 `import seaborn as sns` 导入库后，设置 `sns.set(rc={"figure.dpi":300, 'savefig.dpi':300})` 修改图片分辨率。

当然，如果情况允许尽量导出矢量图，比如 SVG 格式。

边距

一张图少不了上下左右留白，这个留白就是**边距** (margin)。在 Matplotlib 默认情况下，图像周围边距为：

- ▶ `figure.subplot.left`: 0.125
- ▶ `figure.subplot.right`: 0.9
- ▶ `figure.subplot.top`: 0.88
- ▶ `figure.subplot.bottom`: 0.11

如图 1 所示，这些参数的值为 0 到 1 之间的浮点数，相当于图像的宽度或高度的百分比。(0, 0) 表示图形左下角，(1, 1) 表示图形右上角。

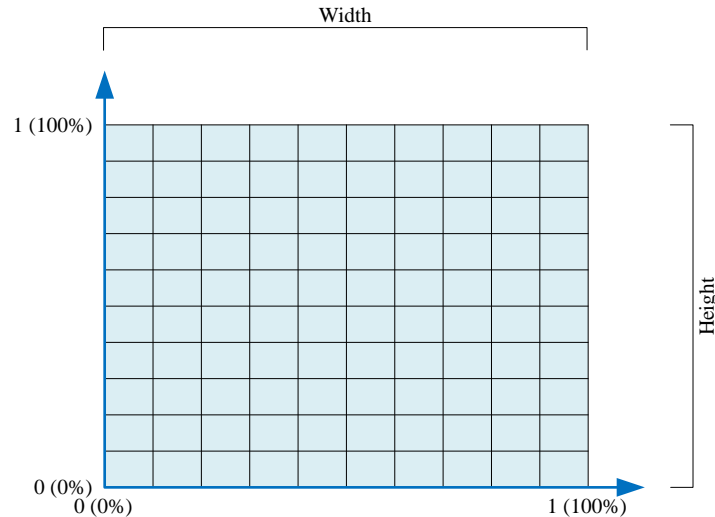


图 1. 图片宽度、高度百分比

如图 8 所示，默认情况下，宽度方向来看，`left = 0.125`，表示左边距相对于图像宽度的 12.5%，`right = 0.9` 表示右边距相对于图像宽度的 90%。

高度方向来看，`top = 0.88` 表示图片的顶边位于图片高度的 88%。而 `bottom = 0.1` 表示底边距相对于图像高度的 10%。

此外，绘制“图中图”时也需要类似的定位方式，具体如图 11 所示。

从图 8、图 11 中，我们可以看到一个 **Figure** 中可以有不止一幅子图。子图是指将整个图形区域划分为多个小的绘图区域，每个区域可以用于绘制不同的图形。下面我们就介绍 Matplotlib 常用的子图布局方法。

3.2 使用 subplot

在 Matplotlib 中，`matplotlib.pyplot.subplot`（下文简作 `subplot`）是一个函数，用于创建和管理图形中的子图。它的基本语法为 `subplot(nrows, ncols, index)`。其中，`nrows` 为子图的行数，`ncols` 为子图的列数，`index`：当前子图的索引（从 1 开始，按先行后列顺序递增）。

图 9 给出示例介绍如何用 `subplot()` 绘制子图，并分别修饰子图。这个例子来自于 *Scientific Visualization: Python & Matplotlib*。

▲ 注意，这幅图中的文本（英文、数字）已经扁平化（`flatten`）。在矢量图中，“`flatten`”通常指的是将文本对象转换为矢量线条，以便更好地支持艺术处理。当文本以矢量形式表示时，它由数学定义的几何形状组成。

比较 `plt.plot()` 和 `ax.plot()`

以绘制二维线图为例，大家肯定会看到 `plt.plot()` 和 `ax.plot()` 这两种不同方法，它俩有一些区别需要大家注意。

本 PDF 文件为作者草稿，发布目的为方便读者在移动终端学习，终稿内容以清华大学出版社纸质出版物为准。

版权归清华大学出版社所有，请勿商用，引用请注明出处。

代码及 PDF 文件下载：<https://github.com/Visualize-ML>

本书配套微课视频均发布在 B 站——生姜 DrGinger：<https://space.bilibili.com/513194466>

欢迎大家批评指教，本书专属邮箱：jiang.visualize.ml@gmail.com

简单来说，`plt.plot()` 相当于“提笔就画”。`plt.plot()` 是使用 `matplotlib` 的 `pyplot` 接口中的函数。它是一种简便的方式来创建图形并进行快速绘图。当你只需要创建一个简单的图形时，可以直接使用 `plt.plot()` 函数，它会自动创建一个图形窗口并在该窗口中绘制图形。如果在同一个图形窗口中绘制多个图形，可以连续多次调用 `plt.plot()` 函数。

⚠ 注意，使用 `plt` 时，需要先导入 `import matplotlib.pyplot as plt`。

`ax.plot()` 基于一个 `Axes` 对象来绘制图形，`Axes` 对象是一个图形窗口中的一个独立坐标系。使用面向对象接口时，需要显式地创建一个 `Figure` 对象和一个或多个 `Axes` 对象，并在指定的 `Axes` 对象上调用 `plot()` 方法进行绘图。比如，如果事先定义了两个 `Axes` 对象，`ax1`、`ax2`，`ax1.plot()` 指定在 `ax1` 上绘图，而 `ax2.plot()` 则在 `ax2` 上绘图。

`ax.plot()` 适合更复杂的绘图需求，并且具有更高的灵活性。

3.3 使用 `add_subplot`

`add_subplot()` 函数用于在图形中添加子图。`add_subplot()` 的基本语法为 `fig.add_subplot(nrows, ncols, index)`。其中，`fig` 为 `fig = plt.figure()` 产生的 `Figure` 对象，`nrows` 为子图的行数，`ncols` 为子图的列数，`index` 为当前子图的索引（从 1 开始，先后列顺序递增）。

`add_subplot()` 返回一个 `AxesSubplot` 对象，它表示创建的子图。我们可以使用此对象进行进一步的图形操作，例如绘制数据、设置轴标签和标题等。

比较 `add_subplot()` 和 `subplot()`

`add_subplot()` 和 `subplot()` 在功能上是相似的，都可以用于创建和管理图形中的子图。它们的主要区别在于使用方式和语法。

`add_subplot()` 是 `Figure` 对象的方法，用于在特定的 `Figure` 上添加子图。`add_subplot()` 语法为 `fig.add_subplot(nrows, ncols, index)`。因此，使用 `add_subplot()` 方法时，首先需要创建一个 `Figure` 对象，然后调用该方法来添加子图，并将子图对象存储在变量中以进行后续的操作。

`subplot()` 是 `pyplot` 模块的函数，用于在当前的图形中添加子图。`subplot()` 语法为 `plt.subplot(nrows, ncols, index)`。使用 `subplot()` 函数时，不需要显式地创建 `Figure` 对象。可以直接调用 `subplot()` 函数，并在同一个代码块中添加多个子图。

混合二维、三维

图 10 中子图混合平面和三维可视化方案。大家可以在配套代码中看到如何分别指定每个子图轴的投影方式。

此外，我们还可以使用 `insert_axes` 和 `add_axes` 在指定位置插入特定宽高的图像。位置、宽高这四个数值均为 0 和 1 之间的浮点数，代表百分比。`insert_axes` 是 `Figure` 对象的方法，用于在

指定位置插入轴。`add_axes` 是 `Figure` 对象或 `Subplot` 对象的方法，用于在指定的图形或子图中添加轴。图 11 给出两个例子。

本章利用了很多对图轴、图脊的操作，这些内容将在下一章系统讲解。

3.4 使用 subplots

在 `Matplotlib` 中，`subplots` 函数用于创建一个包含多个子图的图形布局，并返回一个包含子图对象的元组。以下是使用 `subplots` 函数的基本步骤：

- ▶ 导入绘图模块 `import matplotlib.pyplot as plt`
- ▶ 使用 `subplots` 函数创建子图 `fig`，`axes = plt.subplots(nrows, ncols)`，其中 `nrows` 和 `ncols` 是整数，分别表示子图行和列的数量。
- ▶ `axes[i, j]` 表示在第 `i` 行和第 `j` 列的位置上的子图对象。

此外，可以对每个子图进行布局调整 and 美化。

下面我们使用 `subplots` 可视化极坐标和直角坐标转化。

如图 2 左图所示， O 是极坐标的极点 (pole)，从 O 向右引一条射线作为极轴 (polar axis)，规定逆时针角度为正。这样，平面上任意一点 P 的位置可以由线段 OP 的长度 r 和极轴到 OP 的角度 θ 来确定。 (r, θ) 就是 P 点的极坐标。

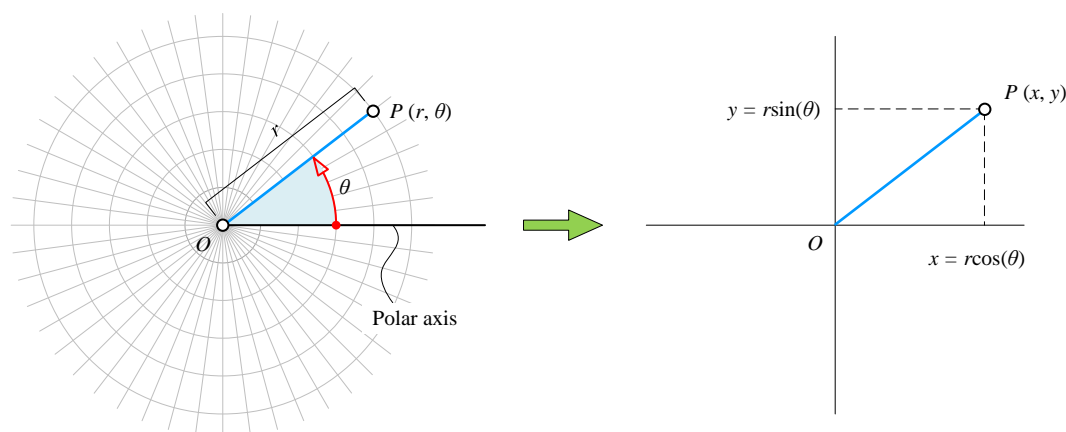


图 2. 从极坐标系到平面直角坐标系

一般， r 称为极径 (radial coordinate 或 radial distance)， θ 称为极角 (angular coordinate 或 polar angle 或 azimuth)。

如图 2 所示，平面上，极坐标 (r, θ) 可以转化为直角坐标系坐标 (x, y) 。

换个角度来看，如图 3 所示，余弦值作为横坐标，正弦值作为纵坐标，画在一幅图上，我们便可以得到圆心位于原点、半径为 1 的正圆，也叫单位圆 (unit circle)。

本 PDF 文件为作者草稿，发布目的为方便读者在移动终端学习，终稿内容以清华大学出版社纸质出版物为准。

版权归清华大学出版社所有，请勿商用，引用请注明出处。

代码及 PDF 文件下载：<https://github.com/Visualize-ML>

本书配套微课视频均发布在 B 站——生姜 DrGinger：<https://space.bilibili.com/513194466>

欢迎大家批评指教，本书专属邮箱：jiang.visualize.ml@gmail.com

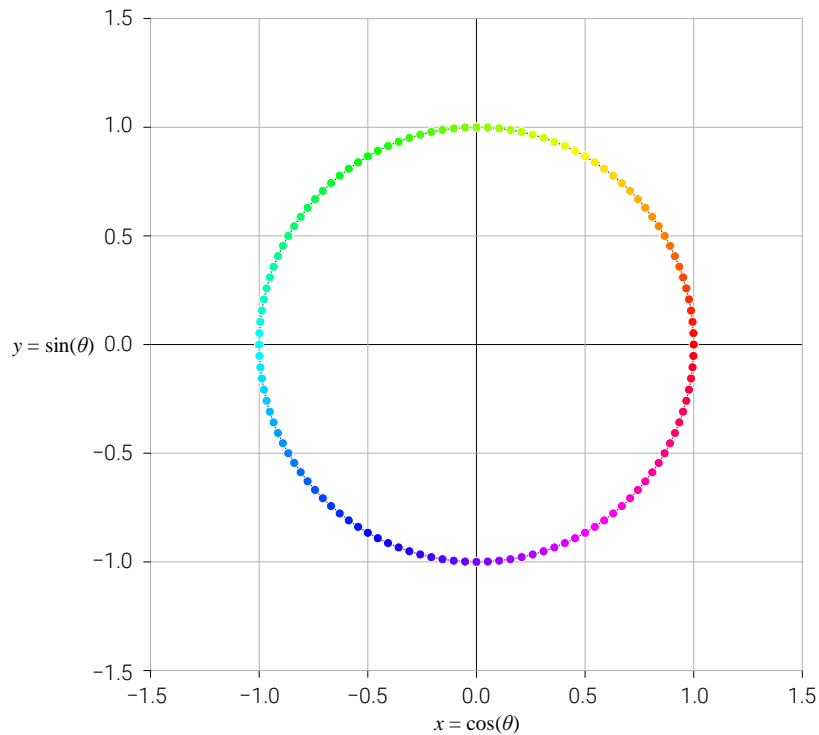


图 3. 单位圆

下面聊聊绘制图 3 的代码。

a 利用 `matplotlib.pyplot.cm.hsv()`，简作 `plt.cm.hsv()`，生成满足 hsv 颜色映射的一组颜色。`np.linspace(0, 1, len(cos_y))` 使用 NumPy 库的 `linspace()` 函数生成一个从 0 到 1 的等间距数组，数组的长度与 `cos_y` 数组的长度相同。

b 利用 `matplotlib.pyplot.subplots()`，简作 `plt.subplots()`，创建图形对象 `fig`、轴对象 `ax`。`figsize=(6, 6)` 参数指定了图像的大小为 6 × 6 英寸。

c 在轴对象 `ax` 上，用 `plot()` 方法绘制线图。

`cos_y` 和 `sin_y` 分别表示 x 轴和 y 轴上的坐标。

`zorder = 1` 指定了图形的层次顺序。`zorder` 值越大，图形就越靠前，即越置顶。

`color = 'k'` 指定了线的颜色。在这里，'k' 代表黑色。本书后续会专门介绍颜色。

`lw = 0.25` 是线的宽度参数 `linewidth` 简称，指定了绘制的线的粗细程度。

d 在轴对象 `ax` 上，用 `scatter()` 方法绘制散点图。

`marker = '.'` 指定了用于标记散点的符号。

`s = 88` 是散点的大小参数。

`c=colors` 是散点的颜色参数，指定了每个散点的颜色。`colors` 是之前生成的包含一系列 HSV 颜色的数组。

`edgecolor='w'` 定了散点边缘的颜色，这里是白色 'w'。

`zorder = 2` 将散点图置于之前绘制的线图之上。

e 用 `axhline()` 在轴对象 `ax` 上绘制水平参考线。

f 用 `axvline()` 在轴对象 `ax` 上绘制竖直参考线。

本 PDF 文件为作者草稿，发布目的为方便读者在移动终端学习，终稿内容以清华大学出版社纸质出版物为准。

版权归清华大学出版社所有，请勿商用，引用请注明出处。

代码及 PDF 文件下载：<https://github.com/Visualize-ML>

本书配套微课视频均发布在 B 站——生姜 DrGinger：<https://space.bilibili.com/513194466>

欢迎大家批评指教，本书专属邮箱：jiang.visualize.ml@gmail.com

g 将图像四周图脊隐去，即不显示。本书下一章将专门介绍图像美化。
请大家在 JupyterLab 中自行实践代码 1。

```
# 导入包
import numpy as np
import matplotlib.pyplot as plt

# 生成数据
theta_array = np.linspace(0, 2*np.pi, 120, endpoint = False)
sin_y = np.sin(theta_array)
cos_y = np.cos(theta_array)
# 用hsv色谱产生一组渐变色，颜色种类和散点数相同
a colors = plt.cm.hsv(np.linspace(0, 1, len(cos_y)))

# 设置图片大小
b fig, ax = plt.subplots(figsize=(6, 6))

# 绘制正圆，横轴坐标为 cos，纵轴坐标为 sin
c ax.plot(cos_y, sin_y,
          zorder = 1, color = 'k', lw = 0.25)
d ax.scatter(cos_y, sin_y, marker = '.', s = 88,
             c=colors, edgecolor='w', zorder = 2)
e ax.axhline(0, c = 'k', zorder = 1)
f ax.axvline(0, c = 'k', zorder = 1)
ax.set_xlabel(r'$x = \cos(\theta)$')
ax.set_ylabel(r'$y = \sin(\theta)$')

# 设置横轴和纵轴范围
ax.set_xlim(-1.5, 1.5)
ax.set_ylim(-1.5, 1.5)
ax.grid(True)
# 横纵轴采用相同的scale
ax.set_aspect('equal')

g ax.spines['top'].set_visible(False)
ax.spines['right'].set_visible(False)
ax.spines['bottom'].set_visible(False)
ax.spines['left'].set_visible(False)
```

代码 1. 绘制单位圆

代码 2 ~ 代码 5 绘制图 4。大家可以发现这幅图有四副子图，下面分别讲解每幅子图对应的代码。

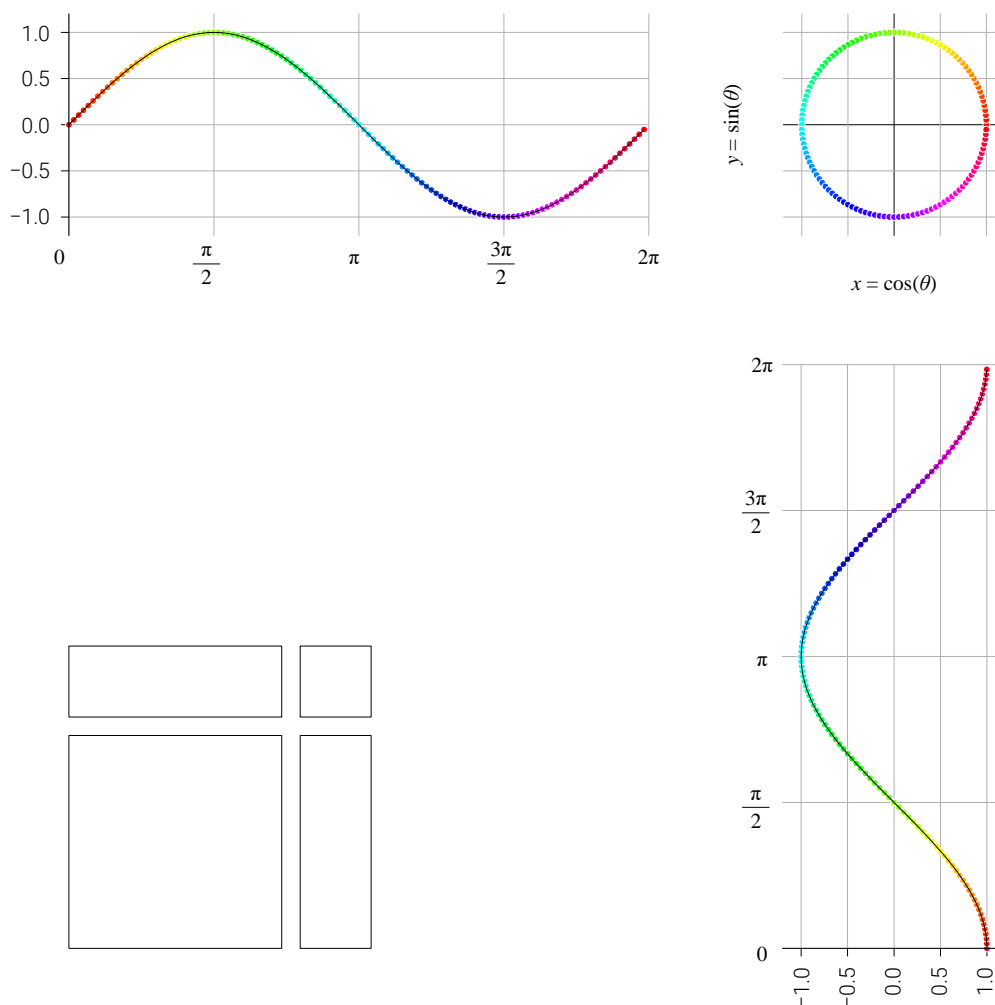


图 4. 极坐标中单位圆原理

代码 2 生成数据，并设置子图布局。

- a** 用 `matplotlib.pyplot.subplots()` 创建图形对象 `fig`，和 2×2 子图对象 `axes`。
 参数 `2, 2` 表示要创建 2 行 2 列的子图。
`figsize=(8, 8)` 指定了整个图表的大小为 8×8 英寸。
`gridspec_kw` 用于指定子图的网格规格。通过 `'width_ratios':[3, 1]` 和 `'height_ratios':[1, 3]` 分别指定了列和行的宽高比。这表示第一列的宽度是第二列的 3 倍，第一行的高度是第二行的 3 倍。这样可以创建不同宽高比的子图。
`axes` 中有四个子图对象，`axes[0,0]` 为左上角子图，`axes[1,0]` 为左下角子图，`axes[0,1]` 为右上角子图，`axes[1,1]` 为右下角子图。

- b** 关闭左下角子图，对应的索引为 `[1,0]`。


```

# 导入包
import numpy as np
import matplotlib.pyplot as plt

# 生成数据
theta_array = np.linspace(0, 2*np.pi, 120, endpoint = False)
sin_y = np.sin(theta_array)
cos_y = np.cos(theta_array)
colors = plt.cm.hsv(np.linspace(0, 1, len(cos_y)))

# 设置子图长宽比例
a fig, axes = plt.subplots(2, 2, figsize = (8,8),
                           gridspec_kw = {
                               'width_ratios':[3, 1],
                               'height_ratios':[1, 3]})

# 刻度
radian_ticks = np.arange(0, 2*np.pi+np.pi/2, np.pi/2)
radian_ticklabels = [r'$0$', r'$\frac{\pi}{2}$',
                     r'$\pi$', r'$\frac{3\pi}{2}$',
                     r'$2\pi$']
level_ticks = [-1, -0.5, 0, 0.5, 1]

# 关闭左下角子图
b axes[1,0].axis('off')

```

代码 2. 生成数据，设置子图布局

在代码 3 中，我们在左上角子图轴对象 `axes[0,0]` 上可视化正弦图像。

```

# 左上角子图：正弦曲线
a axes[0,0].plot(theta_array, sin_y,
                  color = 'k', lw = 0.25)
b axes[0,0].scatter(theta_array, sin_y,
                    marker = '.', s = 38, c=colors,
                    edgecolor='w', zorder = 2)

# 图片美化
axes[0,0].set_xlim(0, 2 * np.pi)
axes[0,0].set_ylim(-1.2, 1.2)
axes[0,0].set_xticks(radian_ticks)
axes[0,0].set_xticklabels(radian_ticklabels)
axes[0,0].set_yticks(level_ticks)
axes[0,0].grid()
axes[0,0].spines['top'].set_visible(False)
axes[0,0].spines['right'].set_visible(False)
axes[0,0].spines['bottom'].set_visible(False)
axes[0,0].spines['left'].set_visible(True)
axes[0,0].set_aspect('equal', adjustable='box')

```

代码 3. 绘制左上角子图

代码 4 在右上角子图轴对象 `axes[0,1]` 上可视化单位圆图像。

```

# 右上角子图：单位圆
a axes[0,1].plot(cos_y, sin_y,
                 color = 'k', lw = 0.25, zorder = 1)
b axes[0,1].scatter(cos_y, sin_y,
                   marker = '.', s = 38, c=colors,
                   edgecolor='w', zorder = 2)

# 图片美化
axes[0,1].axhline(0, c = 'k', zorder = 1)
axes[0,1].axvline(0, c = 'k', zorder = 1)
axes[0,1].set_xlim(-1.2, 1.2)
axes[0,1].set_ylim(-1.2, 1.2)
axes[0,1].set_xticks(level_ticks)
axes[0,1].set_yticks(level_ticks)
axes[0,1].set_xticklabels([])
axes[0,1].set_yticklabels([])
axes[0,1].set_xlabel(r'$x = \cos(\theta)$')
axes[0,1].set_ylabel(r'$y = \sin(\theta)$')
axes[0,1].grid()
axes[0,1].set_aspect('equal', adjustable='box')
axes[0,1].spines['top'].set_visible(False)
axes[0,1].spines['right'].set_visible(False)
axes[0,1].spines['bottom'].set_visible(False)
axes[0,1].spines['left'].set_visible(False)

```

代码 4. 绘制右上角子图

代码 5 在右下角子图轴对象 `axes[1,1]` 上可视化余弦图像。

```

# 右下角子图：余弦曲线
a axes[1,1].plot(cos_y, theta_array,
                 color = 'k', lw = 0.25, zorder = 1)
b axes[1,1].scatter(cos_y, theta_array,
                   marker = '.', s = 38, c=colors,
                   edgecolor='w', zorder = 2)

# 图片美化
axes[1,1].set_ylim(0, 2 * np.pi)
axes[1,1].set_xlim(-1.2, 1.2)
axes[1,1].set_xticks(level_ticks)
axes[1,1].set_yticks(radian_ticks)
axes[1,1].tick_params(axis='x', labelrotation=90)
axes[1,1].set_yticklabels(radian_ticklabels)
axes[1,1].grid()
axes[1,1].spines['top'].set_visible(False)
axes[1,1].spines['right'].set_visible(False)
axes[1,1].spines['bottom'].set_visible(True)
axes[1,1].spines['left'].set_visible(False)
axes[1,1].set_aspect('equal', adjustable='box')

```

代码 5. 绘制右下角子图

图 12 所示为利用 `subplots` 绘制的一元高斯分布概率密度函数曲线随 μ 、 σ 变化。高斯分布，也被称为正态分布，是概率论和统计学中一种常见的连续概率分布。一元高斯分布以钟形曲线的形式表示，具有对称的特点。

一元高斯分布由两个参数完全描述：均值 μ 和标准差 σ 。均值确定了分布的中心位置，标准差决定了分布的形状和展宽程度。一元高斯分布在许多领域中具有广泛的应用，比如统计描述、统计推断、数据分析建模、机器学习等。

图 13 所示为使用 `subplots` 绘制 Beta 分布概率密度函数曲线随 α 、 β 变化。

Beta 分布是概率论和统计学中常见的连续概率分布，它定义在 0 到 1 之间，并具有灵活的形状。Beta 分布由两个形状参数 α 、 β 控制，用于描述随机变量在 0 到 1 之间的概率分布。Beta 分布在许多领域中有广泛的应用，比如概率建模、贝叶斯推断等。

比较 `subplots()` 和 `subplot()`

在 Matplotlib 中，`subplots()` 和 `subplot()` 都是用于创建子图的函数，但它们有一些区别。

`subplots()` 是 pyplot 接口中的函数，用于创建包含多个子图的图形窗口。它返回一个包含所有子图的 Figure 对象和一个包含每个子图的 Axes 对象数组。比如，`fig, axes = plt.subplots(2, 2)` 创建 2 行 2 列子图布局，`axes` 含有四个轴对象。

`subplot()` 是在面向对象接口中使用的函数，用于在一个图形窗口中创建单个子图。它接受三个整数参数：行数、列数和当前子图的索引。通过这些参数，可以在图形窗口中创建一个网格布局，并在指定的位置上放置子图。比如，`ax1 = plt.subplot(2, 2, 1)` 或 `ax1 = plt.subplot(221)` 创建了 2 行 2 列子图的第一个（左上角）的轴。

3.5 使用 GridSpec

在 Matplotlib 中，GridSpec 是一个用于灵活地布局子图的工具。它允许在绘图区域中创建规则的网格，并指定每个子图的大小、位置和跨越的行列数。

使用 GridSpec，可以用更高级的方式组织和排列多个子图，而不是使用默认的单行单列布局。这对于创建复杂的图形布局非常有用，例如在一个绘图区域中显示多个子图，并使它们具有不同的大小和位置。

图 5 所示为 4 × 4 网格中两种子图布局。大家可能已经发现这利用了鸢尾花书《编程不难》中介绍的索引和切片。图 5 中蓝色子图为主图，主图分别位于右上、左下。图 14 所示为使用 GridSpec 绘制满足二元高斯分布的随机数散点图和边缘分布直方图。还可以通过 GridSpec 定义子图的宽度比例、高度比例，如图 6 所示。图 15 可视化 Dirichlet 分布、边缘 Beta 分布，子图的宽度比例、高度比例都是 1:3。

本章后文还会介绍如何用 `add_gridspec` 函数完成类似的可视化方案。

此外，请大家思考通过怎样索引和切片布置能够让主图位于右下、左上。

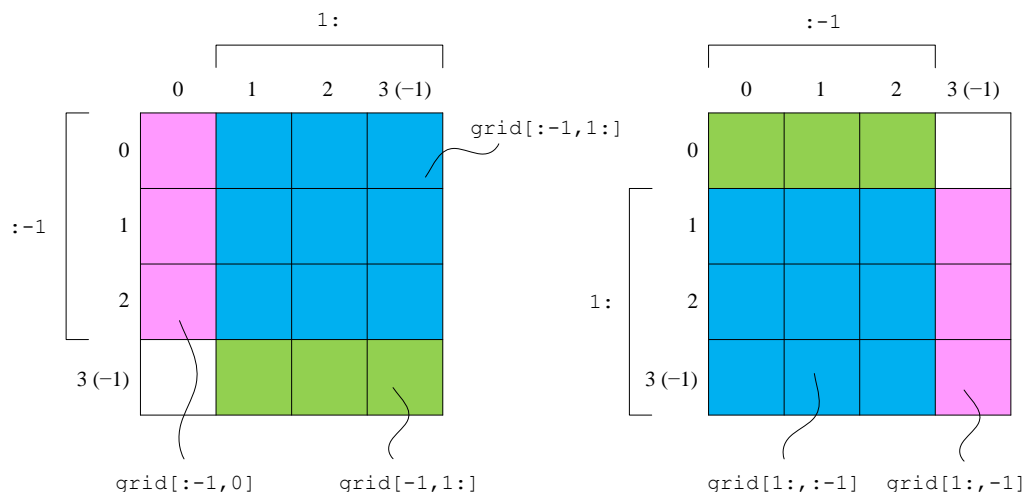


图 5. 4 × 4 网格中两种子图布局

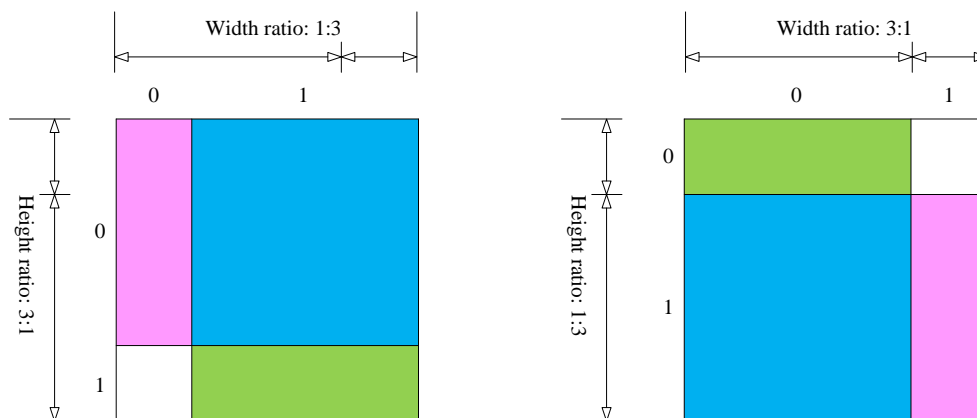


图 6. 2 × 2 网格中两种子图布局以及宽高比例调整

3.6 使用 add_gridspec

在 Matplotlib 中，`add_gridspec` 函数可以用来创建复杂的图形布局。它允许你在图形中创建多个子图，并指定它们的位置和大小。使用 `add_gridspec` 函数的基本步骤如下：

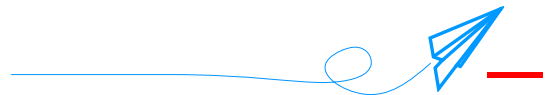
- ▶ 首先导入 `import matplotlib.pyplot as plt`。
- ▶ 然后创建一个 Figure 对象 `fig = plt.figure()`。
- ▶ 再创建一个 GridSpec 对象 `gs = fig.add_gridspec(nrows, ncols)`，其中 `nrows` 和 `ncols` 是整数，分别表示行和列的数量。
- ▶ 最后可以使用 GridSpec 对象创建子图，`gs[i, j]` 表示在第 `i` 行和第 `j` 列的位置创建一个子图。可以使用切片语法来指定多个位置。

在使用 `add_gridspec` 函数绘制子图时同样可以设定轴类型，比如三维、极坐标等等。

在 Matplotlib 中，`subgridspec` 函数用于创建一个更细粒度的子图网格布局，即嵌套子图。它允许你在一个更大的图形布局中创建具有不同大小和位置的子图。图 16 所示为利用 `subgridspec` 函数创建的嵌套子图，这个例子来自 Matplotlib 官网。

Matplotlib 最近还推出了马赛克 `mosaic` 函数用来完成子图布置，请大家自行学习：

https://matplotlib.org/stable/gallery/subplots_axes_and_figures/mosaic.html



此外请大家注意，Plotly 和 Seaborn 有自己安排子图布局方法，请大家自行学习。

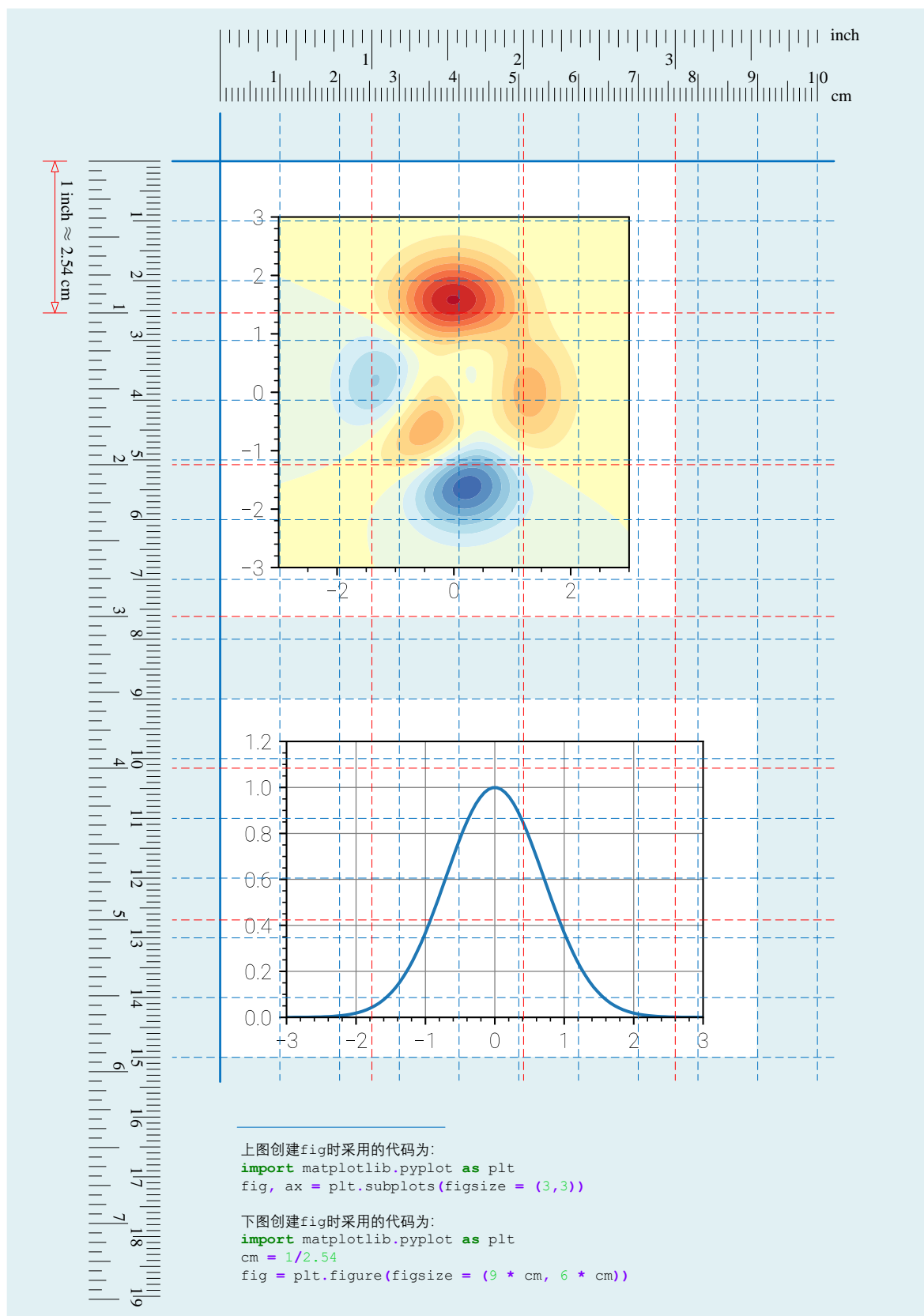


图 7. 自定义图片尺寸 | Bk_2_Ch03_01.ipynb

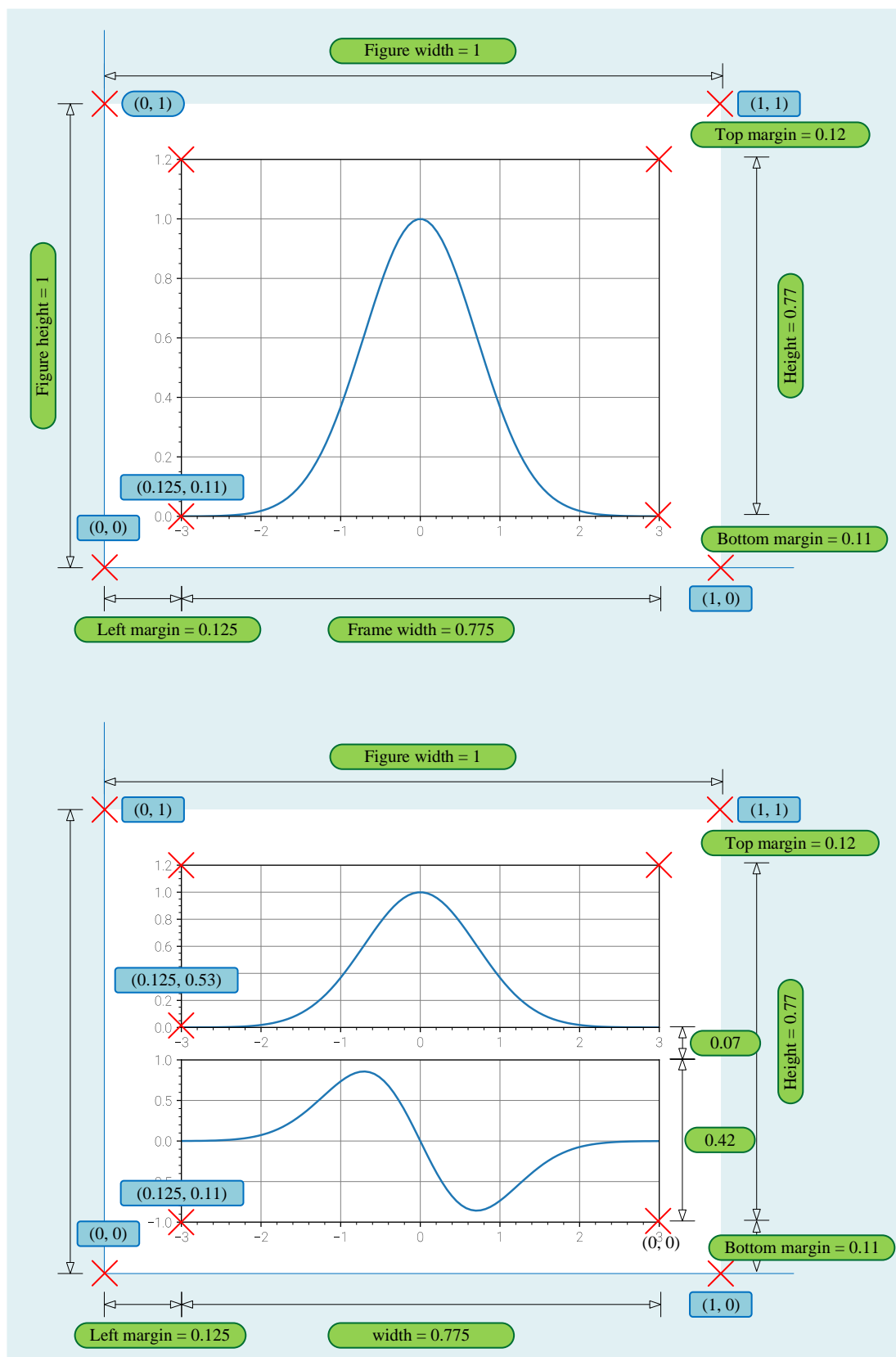


图 8. 图片边距 | Bk_2_Ch03_01.ipynb

本 PDF 文件为作者草稿，发布目的为方便读者在移动终端学习，终稿内容以清华大学出版社纸质出版物为准。

版权归清华大学出版社所有，请勿商用，引用请注明出处。

代码及 PDF 文件下载：<https://github.com/Visualize-ML>

本书配套微课视频均发布在 B 站——生姜 DrGinger：<https://space.bilibili.com/513194466>

欢迎大家批评指教，本书专属邮箱：jiang.visualize.ml@gmail.com

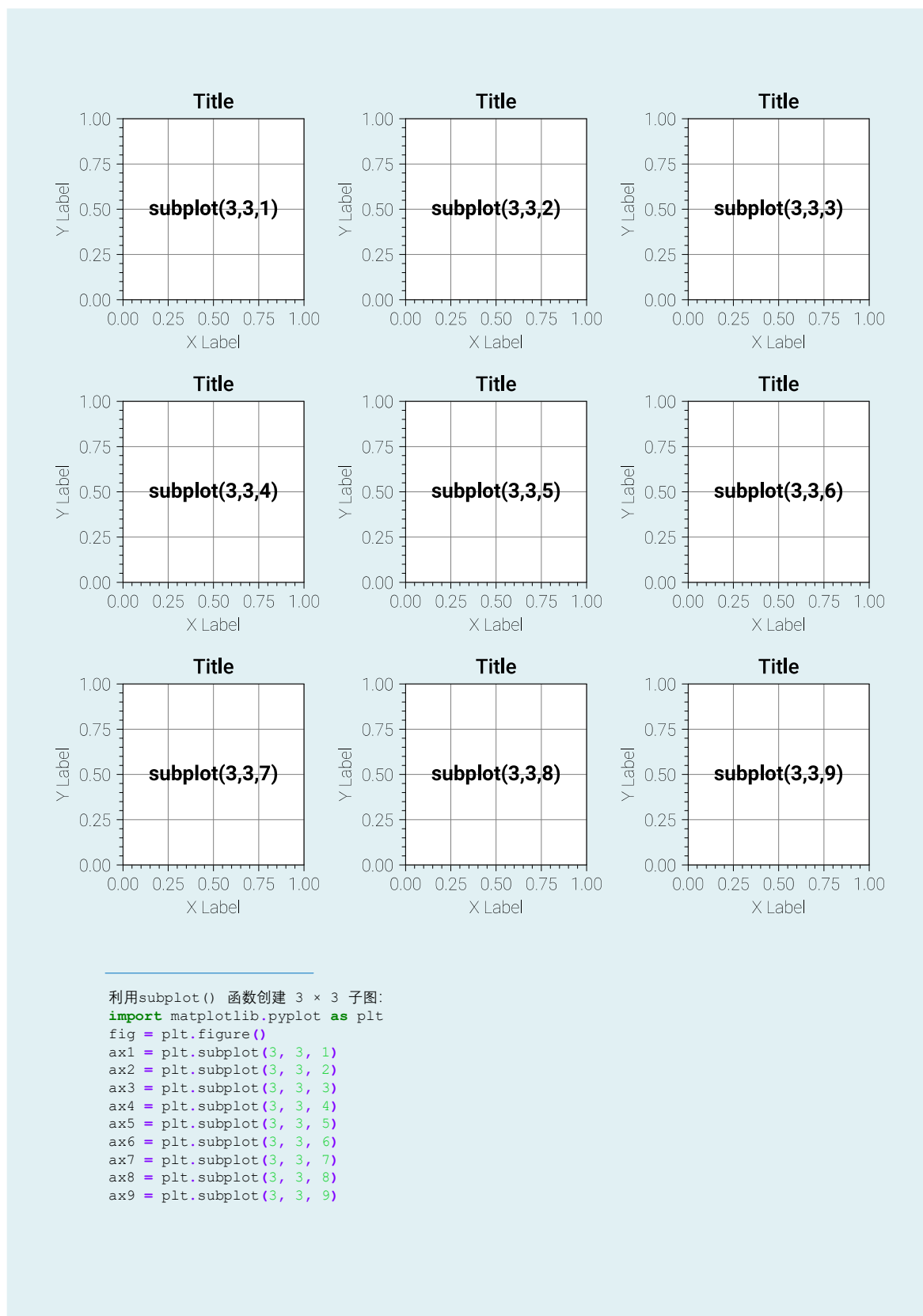


图 9. 使用 subplot() 函数完成子图设置, 来源 <https://github.com/rougier/scientific-visualization-book>

本 PDF 文件为作者草稿, 发布目的为方便读者在移动终端学习, 终稿内容以清华大学出版社纸质出版物为准。

版权归清华大学出版社所有, 请勿商用, 引用请注明出处。

代码及 PDF 文件下载: <https://github.com/Visualize-ML>

本书配套微课程视频均发布在 B 站——生姜 DrGinger: <https://space.bilibili.com/513194466>

欢迎大家批评指教, 本书专属邮箱: jiang.visualize.ml@gmail.com

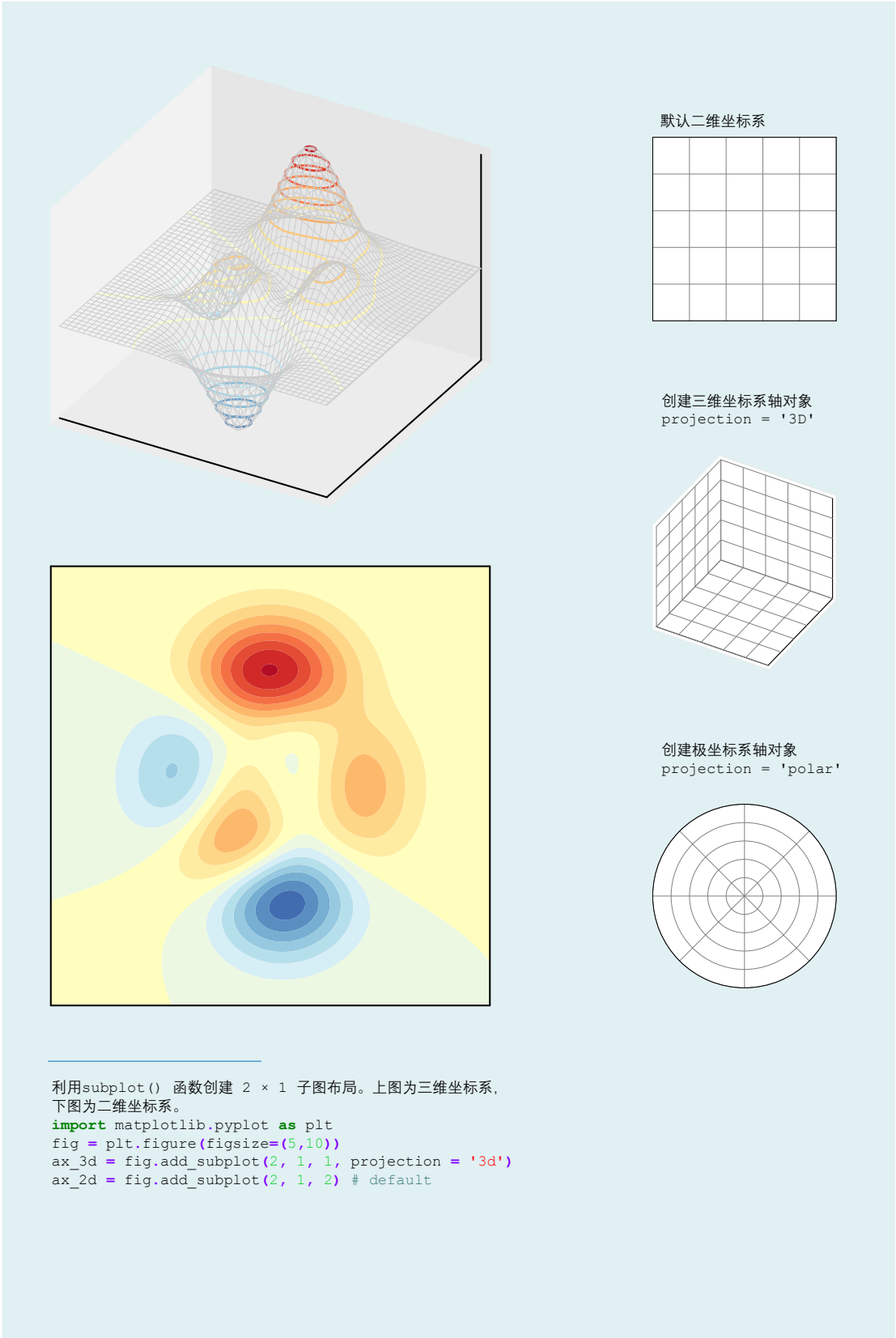

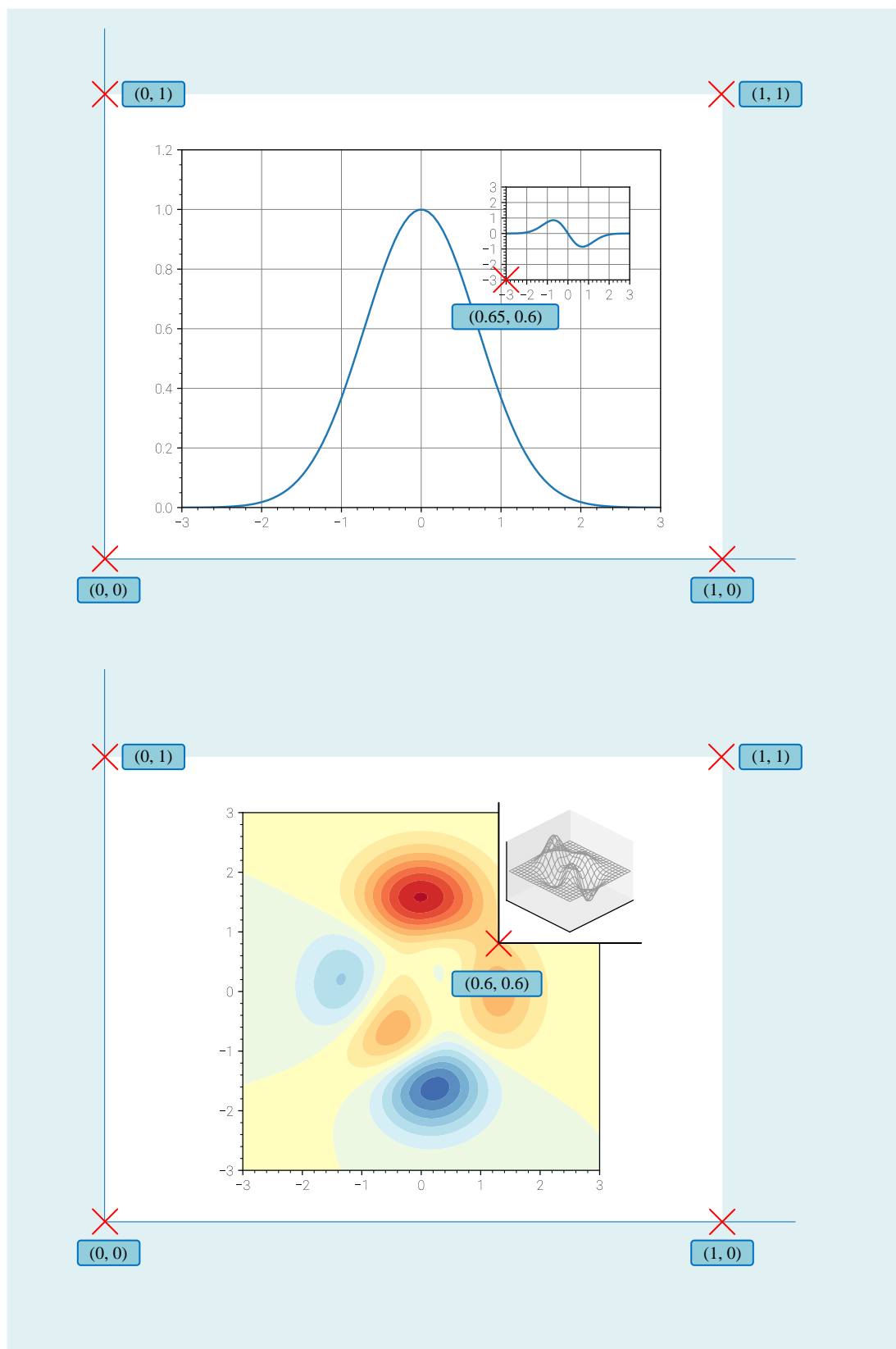
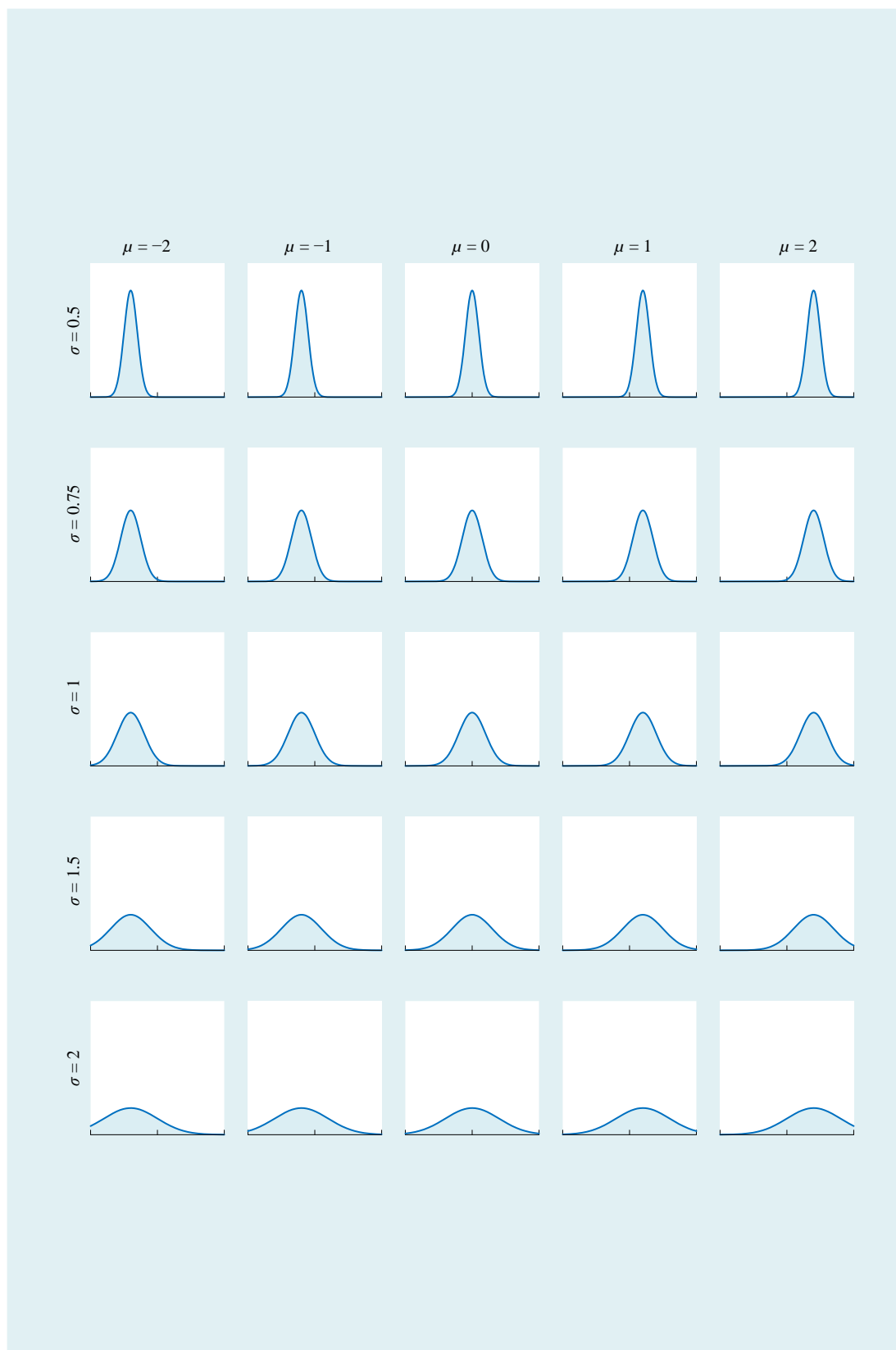



图 10. 使用 add_subplot 混合二维、三维可视化方案 |  Bk_2-Ch03_02.ipynb

图 11. “图中图”定位 |  Bk_2_Ch03_01.ipynb

图 12. 使用 subplots 绘制一元高斯分布概率密度函数曲线随 μ 、 σ 变化 |  Bk_2_Ch03_03.ipynb

本 PDF 文件为作者草稿，发布目的为方便读者在移动终端学习，终稿内容以清华大学出版社纸质出版物为准。

版权归清华大学出版社所有，请勿商用，引用请注明出处。

代码及 PDF 文件下载：<https://github.com/Visualize-ML>

本书配套微视频均发布在 B 站——生姜 DrGinger：<https://space.bilibili.com/513194466>

欢迎大家批评指教，本书专属邮箱：jiang.visualize.ml@gmail.com

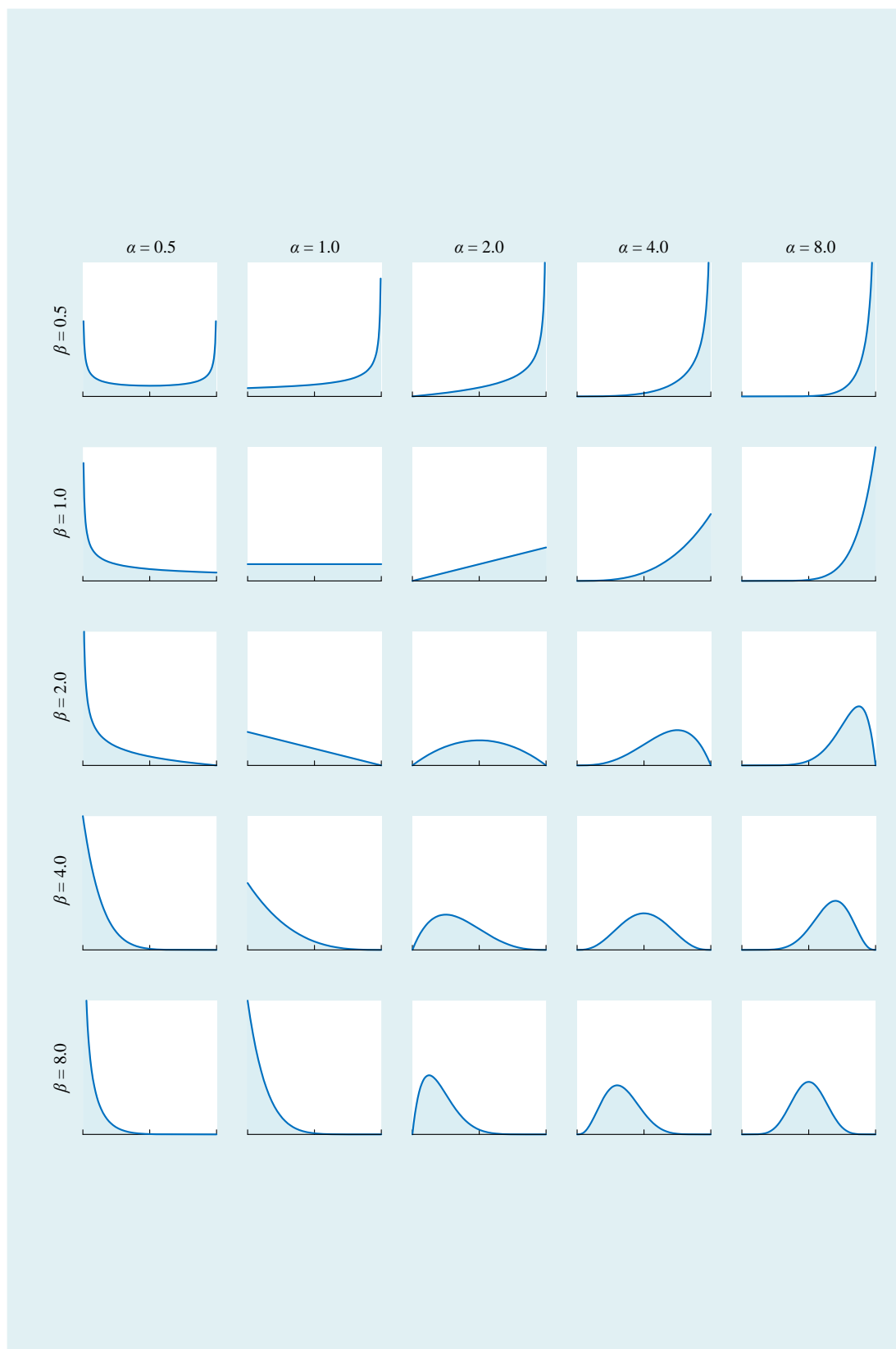



图 13. 使用 subplots 绘制 Beta 分布概率密度函数曲线随 α 、 β 变化 |  Bk_2_Ch03_04.ipynb

本 PDF 文件为作者草稿，发布目的为方便读者在移动终端学习，终稿内容以清华大学出版社纸质出版物为准。

版权归清华大学出版社所有，请勿商用，引用请注明出处。

代码及 PDF 文件下载：<https://github.com/Visualize-ML>

本书配套微课视频均发布在 B 站——生姜 DrGinger：<https://space.bilibili.com/513194466>

欢迎大家批评指教，本书专属邮箱：jiang.visualize.ml@gmail.com

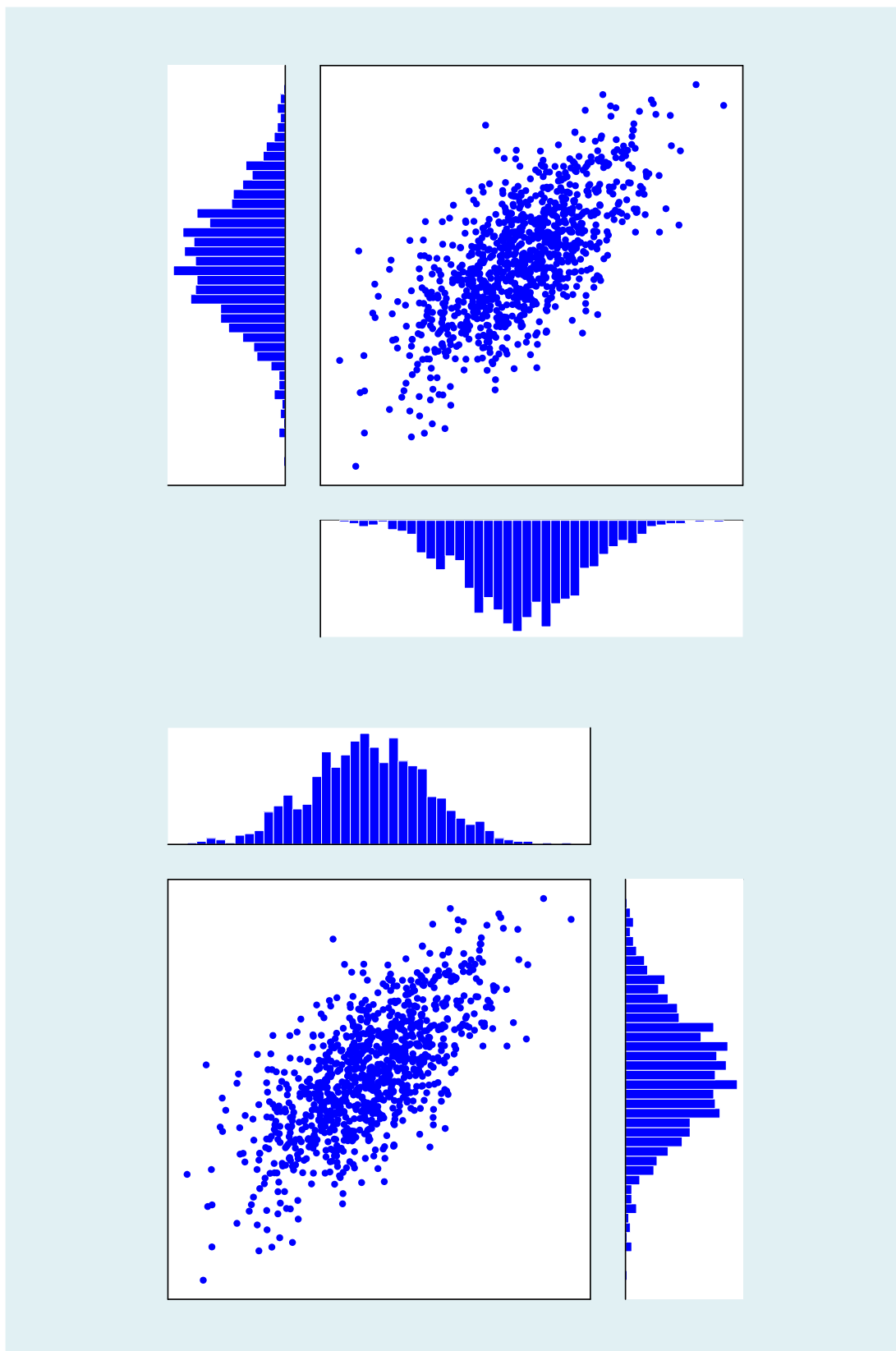



图 14. 使用 GridSpec 绘制满足二元高斯分布的随机数散点图和边缘分布直方图 |  Bk_2_Ch03_05.ipynb

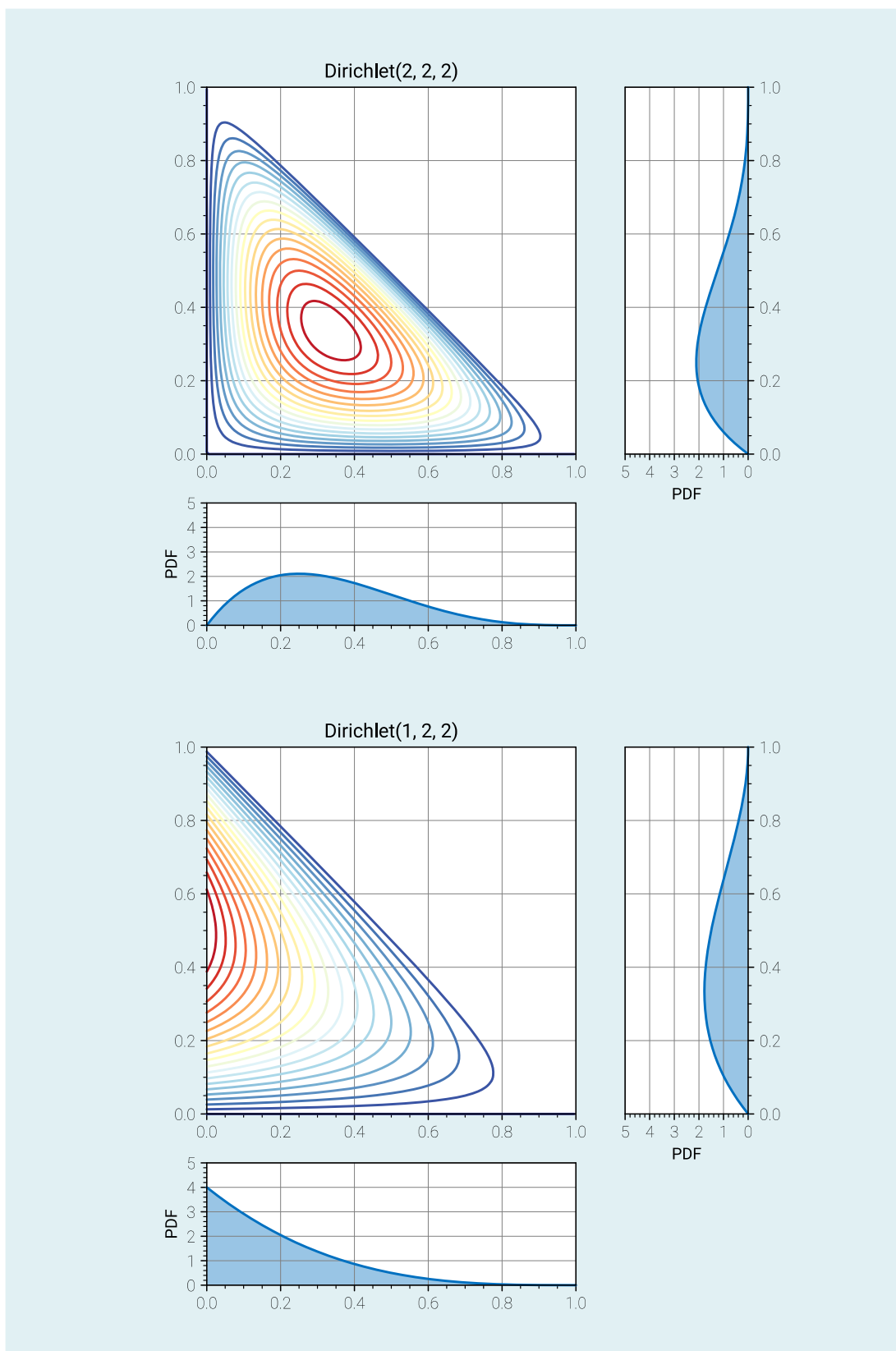


图 15. 使用 GridSpec 绘制的 Dirichlet 分布和边缘 Beta 分布 | [Bk_2_Ch03_06.ipynb](#)

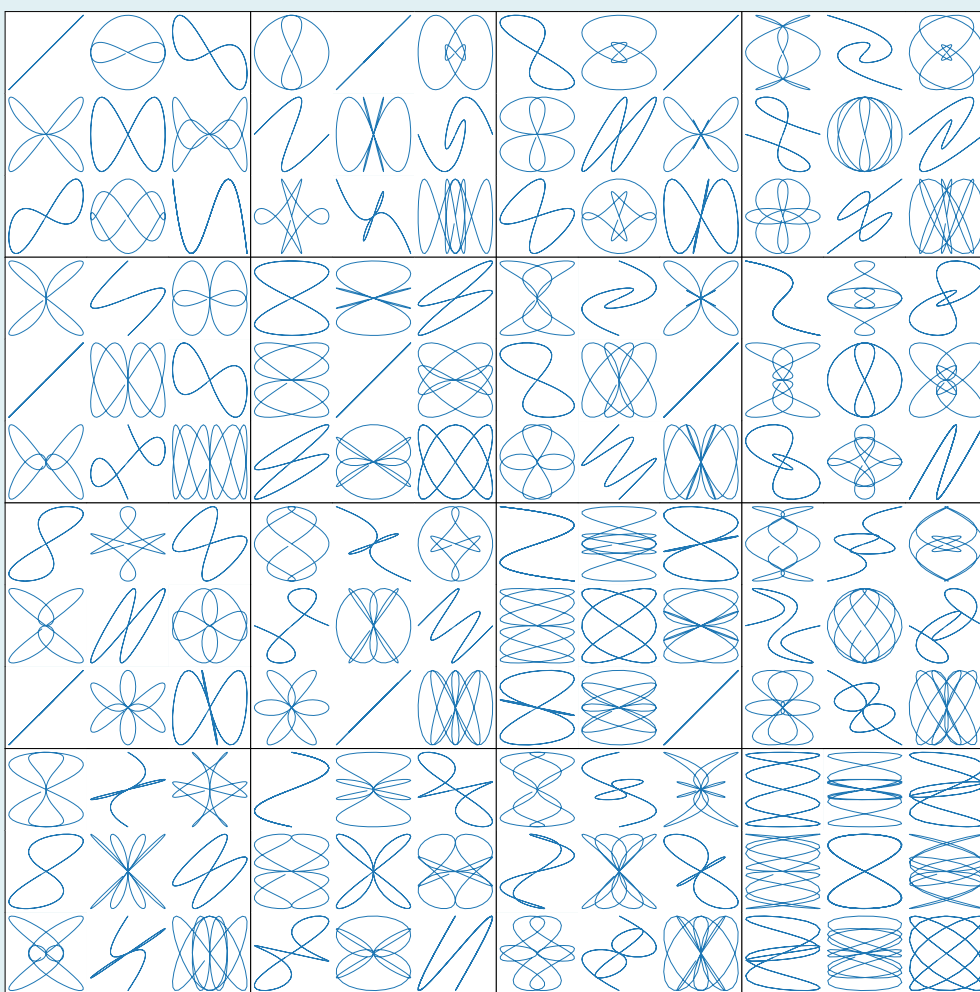

本 PDF 文件为作者草稿，发布目的为方便读者在移动终端学习，终稿内容以清华大学出版社纸质出版物为准。

版权归清华大学出版社所有，请勿商用，引用请注明出处。

代码及 PDF 文件下载：<https://github.com/Visualize-ML>

本书配套微课视频均发布在 B 站——生姜 DrGinger：<https://space.bilibili.com/513194466>

欢迎大家批评指教，本书专属邮箱：jiang.visualize.ml@gmail.com

图 16. 利用 subgridspec 函数创建嵌套子图 |  Bk_2_Ch03_07.ipynb

本 PDF 文件为作者草稿，发布目的为方便读者在移动终端学习，终稿内容以清华大学出版社纸质出版物为准。

版权归清华大学出版社所有，请勿商用，引用请注明出处。

代码及 PDF 文件下载：<https://github.com/Visualize-ML>

本书配套微课视频均发布在 B 站——生姜 DrGinger：<https://space.bilibili.com/513194466>

欢迎大家批评指教，本书专属邮箱：jiang.visualize.ml@gmail.com