

## 23

## Triangular Grid

## 三角网格

将给定的几何形状划分为一组不重叠的三角形



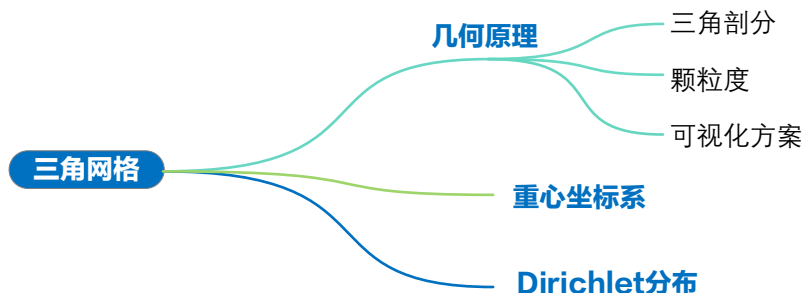
天地间唯一的英雄主义——参破尘世真相，依旧热爱生活。

*There is only one heroism in the world: to see the world as it is, and to love it.*

—— 罗曼·罗兰 (Romain Rolland) | 法国作家、诺贝尔文学奖得主 | 1866 ~ 1944



- ◀ matplotlib.pyplot.plot\_trisurf() 在三角形网格上绘制平滑的三维曲面图
- ◀ matplotlib.pyplot.tricontourf() 在三角形网格上绘制填充的等高线图
- ◀ matplotlib.pyplot.triplot() 在三角形网格上绘制线条
- ◀ matplotlib.tri.Triangulation() 生成三角剖分对象
- ◀ matplotlib.tri.UniformTriRefiner() 对三角形网格进行均匀细化，生成更密集的三角形网格，以提高绘制的精细度和准确性
- ◀ numpy.asarray() 将输入数据，比如列表、元组等，转换为 NumPy 数组
- ◀ numpy.clip() 用于将数组中的元素限制在指定的范围内，将小于最小值的元素设置为最小值，将大于最大值的元素设置为最大值，返回经过剪切后的数组
- ◀ numpy.column\_stack() 将两个矩阵按列合并
- ◀ numpy.meshgrid() 产生网格化数据
- ◀ scipy.spatial.Delaunay() 生成一个点集的 Delaunay 三角剖分
- ◀ scipy.stats.dirichlet.pdf() 计算 Dirichlet 分布的概率密度函数
- ◀ sympy.diff() 求解符号导数和偏导解析式
- ◀ sympy.exp() 符号自然指数
- ◀ sympy.lambdify() 将符号表达式转化为函数



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](mailto:jiang.visualize.ml@gmail.com)

## 23.1 三角剖分

**Delaunay 三角剖分** (Delaunay triangulation) 是计算几何学中的一个重要概念，用于将给定的几何形状划分为一组不重叠的三角形。它的名字来源于它的发明者 Boris Delaunay。

详细来说，三角剖分的目标是将一个多边形或多边形的集合分解成一组互不相交的三角形，使得这些三角形的顶点恰好是原始几何形状的顶点，并且任意两个三角形之间的交集只能是共享一个边或顶点。Delaunay 三角剖分是计算机图形学、计算几何和计算机视觉中常用的技术之一，它在三维重建、图像处理、自然语言处理、机器学习等领域都有广泛的应用。

`matplotlib.tri` 是 Matplotlib 中的一个模块，提供了三角剖分的绘图功能。`scipy.spatial` 中的 `Delaunay` 类可以帮助我们生成一个点集的 Delaunay 三角剖分，它可以用于构建三角形网格、寻找最近邻等等。

图 1 (a) 所示的网格可以手动设定，也可以自动生成。自动生成的三角网格采用 Delaunay 三角剖分。三角网格可以帮助我们绘制各种类型的三角形网格，例如等高线图、三角形色块图和三角形曲面图等。图 1 (b) 所示为三角网格等高线图。

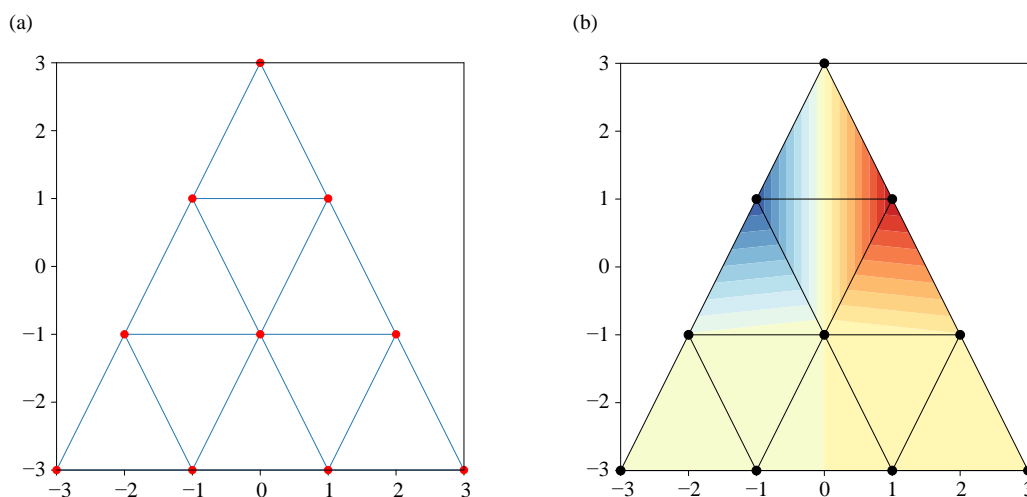


图 1. 三角网格和等高线 | Bk\_2\_Ch23\_1.ipynb

Bk\_2\_Ch23\_1.ipynb 绘制图 1 两幅子图，下面聊聊其中关键语句。

代码 1 绘制图 1 (a)。注意，图中三角形是等腰三角形，不是等边三角形。

- a** 从 `scipy.spatial` 导入 `Delaunay` 对象，`Delaunay` 用于三角剖分。
- b** 创建 `Delaunay` 的一个实例。输入 `points` 是前文代码定义的一组坐标，结果是对这些点进行 Delaunay 三角剖分。
- c** 在轴对象 `ax` 上用 `triplot()` 方法绘制三角剖分图。  
`points[:,0]` 和 `points[:,1]` 是代表给定散点的 `x` 坐标和 `y` 坐标。  
`tri_from_scipy.simplices` 是 `Delaunay` 对象的一个属性，它包含了三角形的顶点索引，这将被用于定义三角剖分的连接关系。
- d** 绘制红色圆点代表 `points` 具体位置。

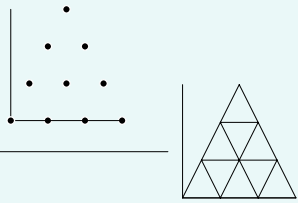
```


a from scipy.spatial import Delaunay
b tri_from_scipy = Delaunay(points)

fig, ax = plt.subplots(figsize = (5,5))
c ax.triplot(points[:,0], points[:,1],
             tri_from_scipy.simplices)

d ax.plot(points[:,0], points[:,1],
          '.r', markersize = 10, lw = 0.25)
ax.set_aspect('equal')
ax.set_xlim(-3,3); ax.set_ylim(-3,3)

```



代码 1. 绘制三角网格 |  Bk\_2\_Ch23\_1.ipynb

代码 2 绘制图 1 (b)。

**a** 将 `matplotlib.tri` 导入，简作 `mtri`。这个模块提供了很多处理和可视化三角剖分工具。这一句被注释掉，是因为在 `Bk_2_Ch23_1.ipynb` 中，模块已经在前文导入。

**b** 利用 `matplotlib.tri` 模块（简作 `mtri`）中的 `Triangulation` 根据给定的坐标点自动构建三角剖分。

**c** 利用 `tricontourf()` 在轴对象 `ax` 上绘制一个基于三角剖分的等高线填充图。

`triang_auto` 之前创建的 `Triangulation` 对象示例，它定义了三角剖分的结构。

`f_fcn()` 是代码前文定义的一个二元函数。函数值将用于确定填充图中每个三角形的颜色。

`cmap='RdYlBu_r'` 指定了使用的颜色映射。

`levels=20` 指定了等高线层数。

**d** 用 `triplot()` 在轴对象 `ax` 上绘制三角剖分线条图。

`'ko-'` 是线条图的样式设置，其中 `'k'` 表示黑色，`'o'` 表示在每个顶点处用圆圈标记，`'-'` 表示连接线样式。

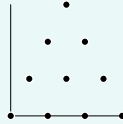
```


a # import matplotlib.tri as mtri
b triang_auto = mtri.Triangulation(x_tri, y_tri)

fig, ax = plt.subplots(figsize = (5,5))

# 三角剖分网格绘制等高线
c ax.tricontourf(triang_auto, f_fcn(x_tri, y_tri),
                cmap = 'RdYlBu_r',
                levels = 20)
d ax.triplot(triang_auto, 'ko-')
ax.set_aspect('equal')
ax.set_xlim(-3,3); ax.set_ylim(-3,3)

```



代码 2. 绘制三角网格等高线 |  Bk\_2\_Ch23\_1.ipynb

`Bk_2_Ch23_1.ipynb` 笔记中给出更多三角剖分以及相关可视化的范例，请大家自行学习。

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](mailto:jiang.visualize.ml@gmail.com)

## 颗粒度

三角网格也存在颗粒度的问题。图 2 所示为给定等边三角形不同的颗粒度的三角网格划分。颗粒度越高，三角网格越细腻，但是计算量也急剧增大。

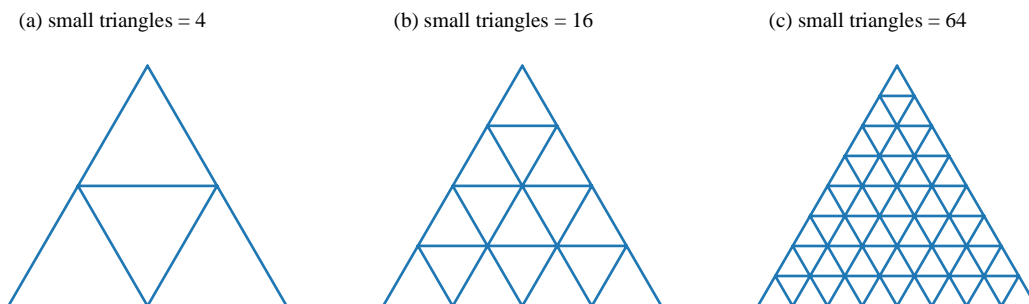
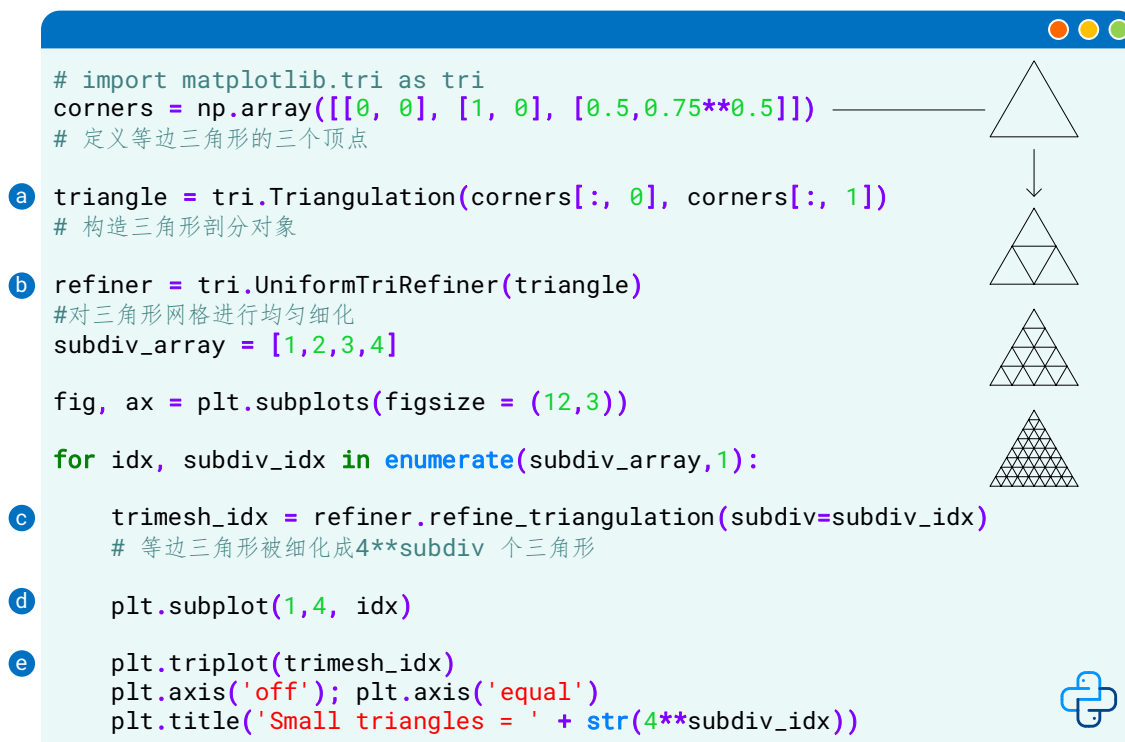


图 2. 三角网格的颗粒度 |  Bk\_2\_Ch23\_2.ipynb

Bk\_2\_Ch23\_2.ipynb 绘制图 2。下面聊聊其中关键语句。

- a** 利用 `matplotlib.tri.Triangulation()`，简作 `tri.Triangulation()`，根据给定的坐标点自动构建三角剖分。`corners` 中有三个点，它们是等边三角形的三个顶点。
- b** 利用 `matplotlib.tri.UniformTriRefiner()`，简作 `tri.UniformTriRefiner()`，创建三角网格均匀细化对象实例。
- c** 利用 `refine_triangulation()` 方法对 **b** 中创建的 `refiner` 进行细分操作。  
参数 `subdiv=subdiv_idx` 指定三角形边的细化次数，我们可以控制细化程度，以便生成更精细的三角形网格。
- d** 利用 `matplotlib.pyplot.subplot()`，简作 `plt.subplot()`，创建 1 行 4 列子图，并选择当前操作的子图序号 `idx`。注意，`idx` 从 1 开始编号，这是因为 `for` 循环中使用 `enumerate()` 时，加入了 1 这个参数。
- e** 利用 `matplotlib.pyplot.triplot()`，简作 `plt.triplot()`，绘制三角剖分线条图。



代码 3. 提高三角网格颗粒度 | Bk\_2\_Ch23\_2.ipynb

## 23.2 重心坐标系

“鸢尾花书”中三维网格常常用来可视化重心坐标系。从物理角度来看，**重心坐标系** (barycentric coordinate system) 是一种描述一个几何形状内部任意点位置的方法。它是以该形状的重心作为原点建立的坐标系。

在平面上的一个三角形中，任何一点都可以表示为三个定点的加权平均值，其中每个定点的权重由它到该点的距离与该三角形的周长之比确定。这些权重称为该点在三角形的重心坐标。

实际上，用三维直角坐标系解释重心坐标系更方便。图 3 左图所示为三维直角坐标系，为了区分坐标系的横轴、纵轴、竖轴分别记做  $\theta_1$ 、 $\theta_2$ 、 $\theta_3$ 。这个三维直角坐标系中坐标可以记做  $(\theta_1, \theta_2, \theta_3)$ 。

图 3 左图浅蓝色平面上的每个坐标  $(\theta_1, \theta_2, \theta_3)$  都满足  $\theta_1 + \theta_2 + \theta_3 = 1$ 。 $\theta_1 + \theta_2 + \theta_3 = 1$  这个限制条件，让原本三维的空间降维成二维。即便如此，如图 3 右图所示，三角网格的每一点仍旧对应  $(\theta_1, \theta_2, \theta_3)$ 。

如图 4 所示，本章并用两种变量  $\theta_1$ 、 $\theta_2$ 、 $\theta_3$  标注位置。

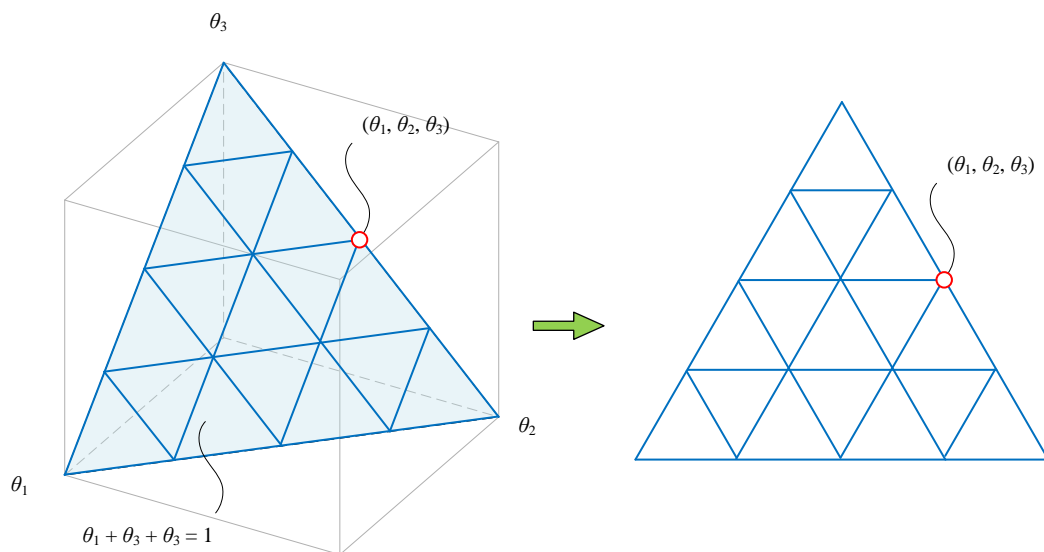


图 3. 从三维直角坐标系到重心坐标系

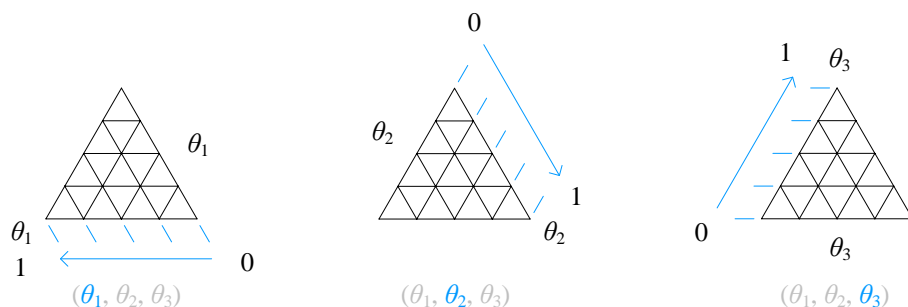


图 4. 重心坐标系的三个变量

图 7 从三维直角坐标系视角观察中心坐标系中的  $\theta_1$ 、 $\theta_2$ 、 $\theta_3$ 。

图 8 上下两幅子图比较**等边三角形** (equilateral triangle) 中的两个坐标系坐标关系；上图给出的是利用三角网格表达平面直角坐标系。下图则是利用相同网格表达重心坐标系坐标。请大家注意区分。

图 9 则展示**等腰直角三角形** (isosceles right triangle) 中两个坐标系坐标关系。图 10 所示为任意形状三角形中两个坐标系的关系。

Bk\_2\_Ch23\_3.ipynb 绘制图 8、图 9、图 10，下面聊聊其中核心语句。

代码 4 在直角坐标系中生成三角网格坐标点。有了本章前文的基础，大家应该对这几句代码很熟了。

**a** 定义三角形的三个顶点坐标，这个三角形是图 8 中等边三角形。Bk\_2\_Ch23\_3.ipynb 还给出两个其他三角形的顶点坐标。

**b** 构造三角网格对象，然后细分网格。

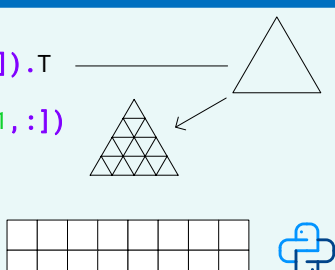
**c** 将三角网格坐标点构造成二维数组（矩阵），每个列向量代表一个坐标点。

```

# 等边三角形
a corners = np.array([[0, 0], [1, 0], [0.5, 0.75**0.5]]).T
b triangle = tri.Triangulation(corners[0,:], corners[1,:])
  refiner = tri.UniformTriRefiner(triangle)
  trimesh_2 = refiner.refine_triangulation(subdiv=2)

# 每个列向量代表一个三角网格坐标点
c r_array = np.row_stack((trimesh_2.x, trimesh_2.y))

```



代码 4. 直角坐标系中的三角网格坐标点 | Bk\_2\_Ch23\_3.ipynb

代码 5 可视化直角坐标系中网格坐标。

a 用 `matplotlib.pyplot.triplot()`，简作 `plt.triplot()`，绘制三角网格。

b 使用 `for` 循环在图片上打印三角网格每个点的坐标。

其中，`zip()` 将多个可迭代对象打包，从而方便 `for` 循环遍历。

每次迭代，创建 `text_idx`，其中包含横纵坐标的字符串。`format()` 将坐标值格式化为小数点后两位。

然后利用 `matplotlib.pyplot.text()`，简作 `plt.text()`，将文本添加到图片中。

`x_idx` 是文本标签的 `x` 坐标。

`y_idx+0.03` 是为了在 `y` 方向上将文本标签稍微上移，以免与数据点重叠。

`text_idx` 是要显示的文本字符串。

`fontsize=8` 设置文本的字体大小为 8 pt。

`horizontalalignment='center'` 将文本水平居中对齐。

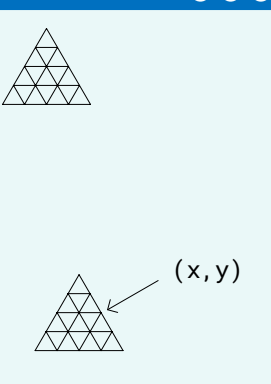
`bbox=dict(facecolor='w', alpha=0.5, edgecolor='None')` 定义了一个包含文本的矩形框。其中 `facecolor='w'` 设置矩形框的背景色为白色，`alpha=0.5` 设置透明度为 0.5，`edgecolor='None'` 表示矩形框没有边框。

```


fig, ax = plt.subplots(figsize = (5,5))
a plt.triplot(trimesh_2)
  plt.plot(r_array[0,:],
           r_array[1,:],
           '.r',
           markersize = 10)

b for x_idx, y_idx in zip(trimesh_2.x, trimesh_2.y):
    text_idx = '(' + format(x_idx, '.2f') +
               ', ' + format(y_idx, '.2f') + ')'
    plt.text(x_idx, y_idx+0.03,
             text_idx,
             fontsize = 8,
             horizontalalignment = 'center',
             bbox=dict(facecolor='w', alpha=0.5, edgecolor = 'None'))
ax.set_aspect('equal')
ax.set_xlim(0,1); ax.set_ylim(0,1)

```





代码 5. 展示直角坐标系中网格坐标 |  Bk\_2\_Ch23\_3.ipynb

代码 6 展示的是直角坐标系和重心坐标系坐标转换运算。图 5 所示为转换过程用到的具体数学工具，大家可以看到其中最重要的运算是矩阵求逆。Bk\_2\_Ch23\_3.ipynb 列出参考文献，大家可以自行学习。

**a** 提取大三角形三个顶点坐标，结果为二维数组，相当于列向量。

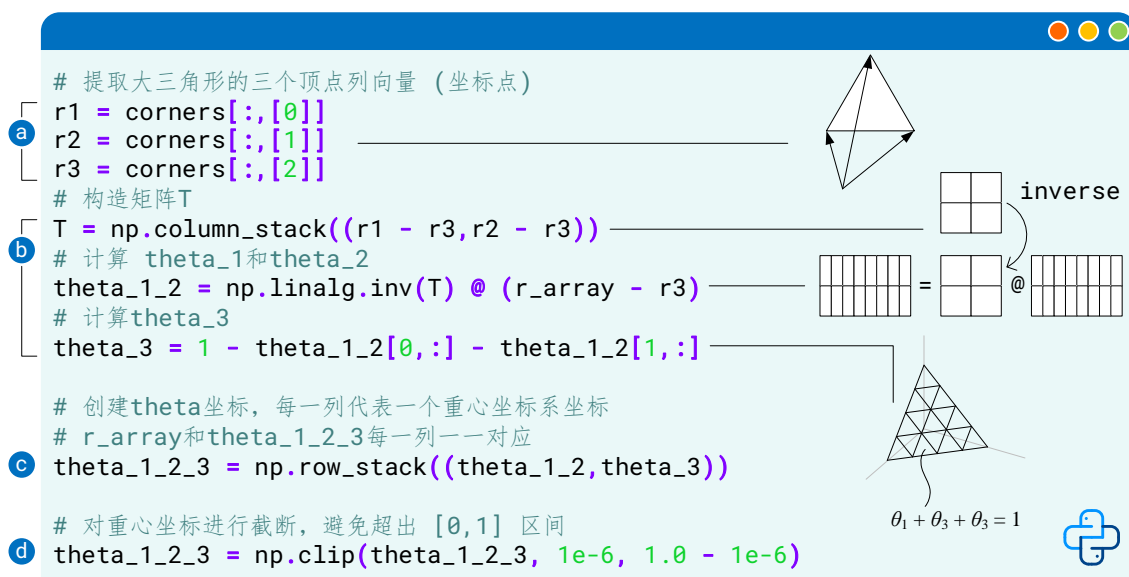
**b** 首先创建矩阵  $T = [r_1 - r_3 \quad r_2 - r_3]$ 。

然后，通过  $\begin{bmatrix} \theta_1 \\ \theta_2 \end{bmatrix} = T^{-1} (r - r_3)$  计算  $\theta_1$ 、 $\theta_2$ 。代码中用到了广播原则。r\_array 是直角坐标系中有待转换的坐标点。

根据等式  $\theta_1 + \theta_2 + \theta_3 = 1$  计算  $\theta_3$ 。注意，r\_array 和 theta\_1\_2\_3 每一列一一对应。也就是说，至此，我们完成了直角坐标系和重心坐标系之间的坐标转换。

**c** 利用 numpy.row\_stack() 创建一个矩阵，每一列代表一个重心坐标坐标。

**d** 利用 numpy.clip() 对计算得到的重心坐标进行截断，确保它们在合理的范围内，避免超出  $[0, 1]$  区间。1e-6 是一个小的容差值，用于避免由于数值误差导致的坐标超出范围。




```

# 提取大三角形的三个顶点列向量（坐标点）
a r1 = corners[:, [0]]
    r2 = corners[:, [1]]
    r3 = corners[:, [2]]
# 构造矩阵T
b T = np.column_stack((r1 - r3, r2 - r3))
# 计算 theta_1和theta_2
    theta_1_2 = np.linalg.inv(T) @ (r_array - r3)
# 计算theta_3
    theta_3 = 1 - theta_1_2[0, :] - theta_1_2[1, :]

# 创建theta坐标，每一列代表一个重心坐标系坐标
# r_array和theta_1_2_3每一列一一对应
c theta_1_2_3 = np.row_stack((theta_1_2, theta_3))

# 对重心坐标进行截断，避免超出 [0, 1] 区间
d theta_1_2_3 = np.clip(theta_1_2_3, 1e-6, 1.0 - 1e-6)
  
```

Diagram illustrating the conversion process: A large triangle is divided into smaller triangles. A point  $r$  is shown within the large triangle. The transformation involves calculating barycentric coordinates  $\theta_1, \theta_2, \theta_3$  such that  $\theta_1 + \theta_2 + \theta_3 = 1$ . The code uses matrix operations to find  $\theta_1$  and  $\theta_2$  from the coordinates of the vertices and the point, then calculates  $\theta_3$  and clips the results to the range  $[0, 1]$ .

代码 6. 直角坐标系和重心坐标系转换 |  Bk\_2\_Ch23\_3.ipynb



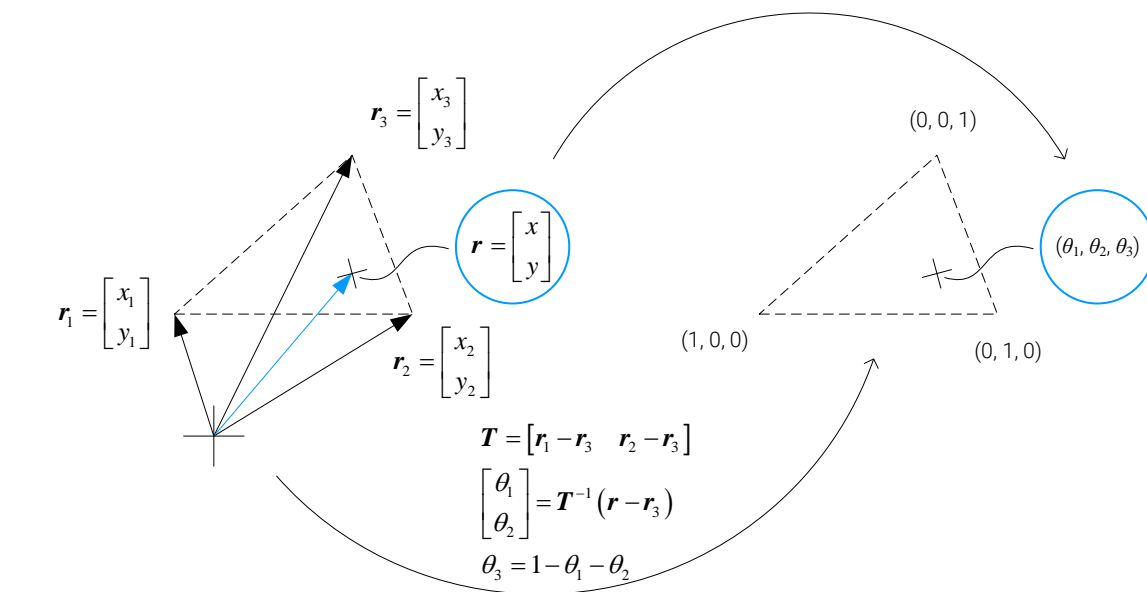


图 5. 平面直角坐标系到重心坐标系转换背后的线性代数工具

**?** 请大家修改 `Bk_2_Ch23_3.ipynb`, 修改原始大三角形的三个顶点坐标为  $(0, 8)$ 、 $(8, 0)$ 、 $(5, 8)$ , 重新绘制重心坐标系网格坐标。

图 11 所示为在重心坐标系中混合红绿蓝三色。`Bk_2_Ch23_3.ipynb` 绘制图 11, 代码很简单, 请大家自行学习。

## 23.3 Dirichlet 分布

“鸢尾花书”中, 重心坐标系常用来可视化 Dirichlet 分布概率密度函数。



《统计至简》一册将专门讲解 Dirichlet 分布及其在贝叶斯推断的应用。

Dirichlet 分布是一种连续的概率分布, 通常用于描述一个多元随机变量的概率分布。它是以德国数学家 Peter Gustav Lejeune Dirichlet 的名字命名的。

本书前文利用其它可视化方案展示过 Dirichlet 分布。图 12、图 13 向大家展示如何利用重心坐标系可视化三元 Dirichlet 分布随参数  $[\alpha_1, \alpha_2, \alpha_3]$  变化。

`Bk_2_Ch23_5.ipynb` 绘制图 12、图 13, 下面聊聊其中关键语句。

代码 7 所示为自定义函数用来可视化 Dirichlet 分布。

**a** 用 `scipy.stats.dirichlet.pdf()`, 简作 `dirichlet.pdf()`, 计算给定重心坐标系网格上点的 Dirichlet 分布的概率密度函数值。这个函数的输入为网格坐标  $(\theta_1, \theta_2, \theta_3)$ , 分布参数  $(\alpha_1, \alpha_2, \alpha_3)$ 。其中,  $\theta_1, \theta_2, \theta_3$  非负, 且满足  $\theta_1 + \theta_2 + \theta_3 = 1$ , 正是这个条件使得我们可以用重心坐标系可视化 Dirichlet 分布 PDF。

其中, `xy2bc()` 为自定义函数, 将直角坐标系坐标转化为重心坐标系坐标。这个自定义函数的核心代码来自代码 6。

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。

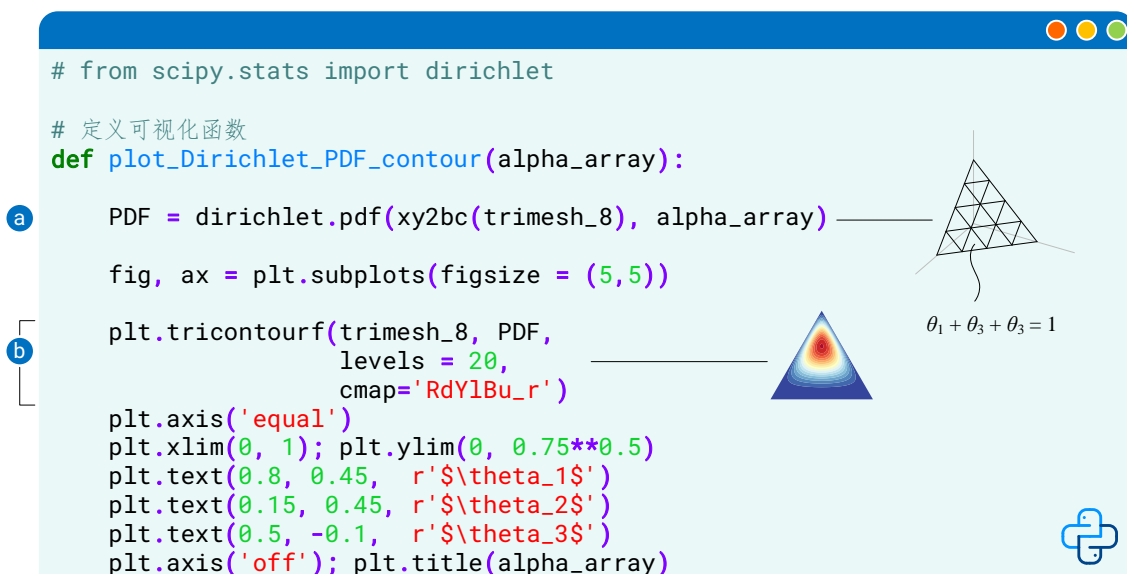
版权归清华大学出版社所有, 请勿商用, 引用请注明出处。

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](mailto:jiang.visualize.ml@gmail.com)

**b** 利用 `matplotlib.pyplot.tricontourf()`，简作 `plt.tricontourf()`，绘制基于三角形网格的等高线填充图。



代码 7. 可视化 Dirichlet 分布 | Bk\_2\_Ch23\_5.ipynb

Plotly 中也有绘制基于三角形网格等高线的函数，具体图 14 所示。

对应的代码文件为 Bk\_2\_Ch23\_6.ipynb。

利用的函数为 `plotly.figure_factory.create_ternary_contour()`；代码相对简单，请大家自行学习使用这个函数。

如图 6 所示，在 Bk\_2\_Ch23\_6.ipynb 上，我们用 Streamlit 制作了一个 App，用来展示 Dirichlet 分布参数对概率密度函数图像影响。

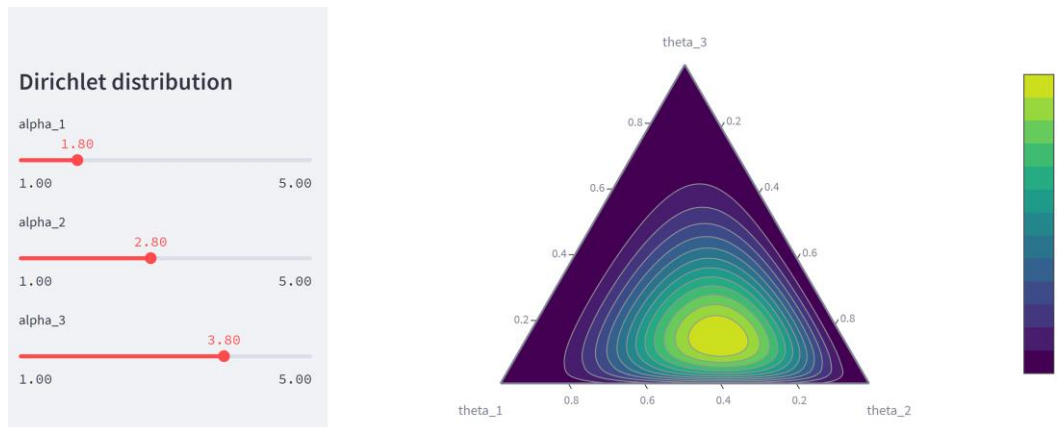


图 6. Streamlit 创建 Dirichlet 分布 App | Bk\_2\_Ch23\_7.py

本章介绍了 Delaunay 三角剖分、重心坐标系、可视化 Dirichlet 分布 PDF。稍有挑战性的知识点是如何理解重心坐标系，以及直角坐标系和重心坐标系坐标的转换。

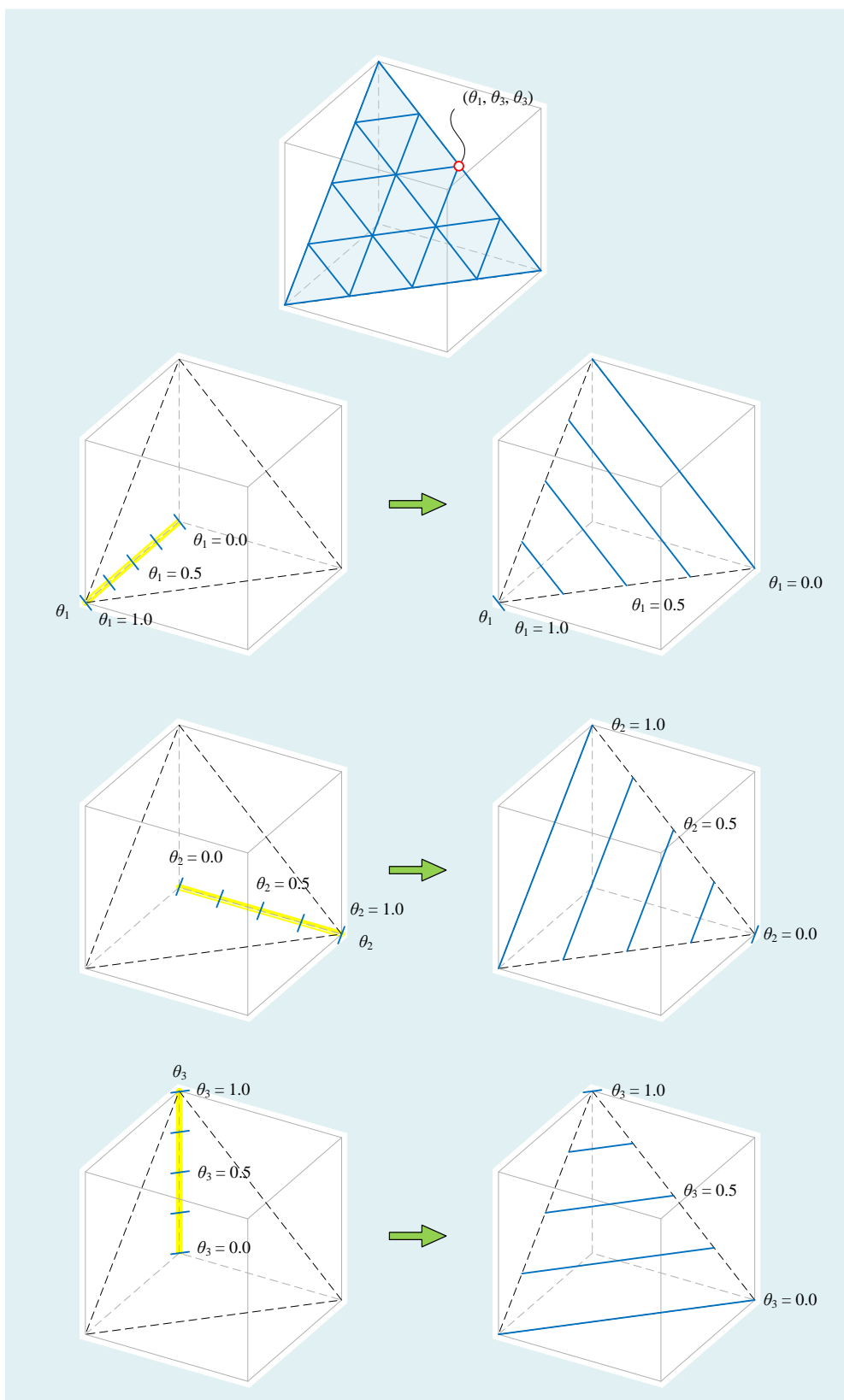


图 7. 三维直角坐标系角度看重心坐标系

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](mailto:jiang.visualize.ml@gmail.com)

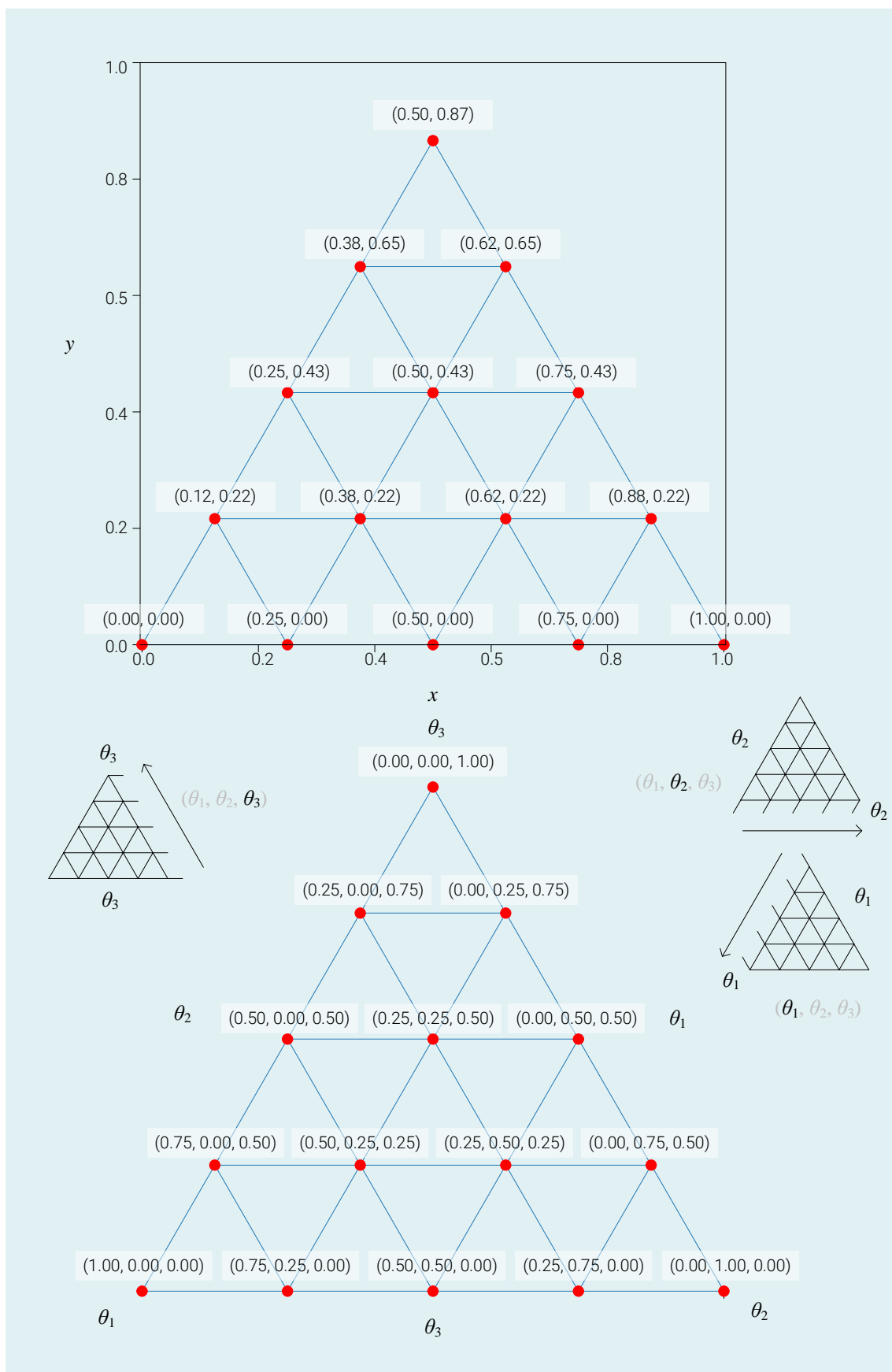


图 8. 比较平面直角坐标系坐标、重心坐标系坐标，等边三角形 | Bk\_2\_Ch23\_3.ipynb

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](mailto:jiang.visualize.ml@gmail.com)

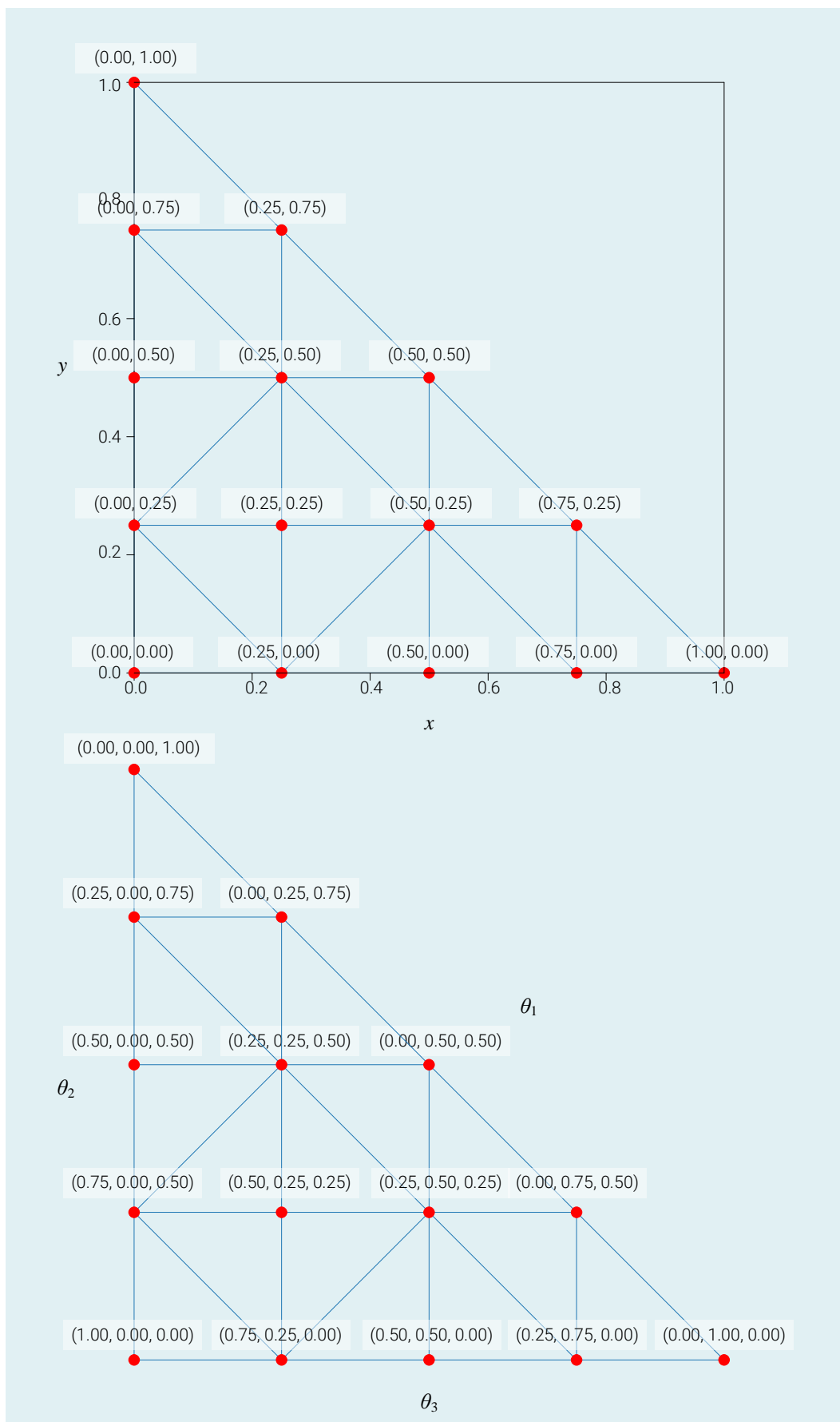


图 9. 比较平面直角坐标系坐标、重心坐标系坐标, 等腰直角三角形 | Bk\_2\_Ch23\_3.ipynb

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](mailto:jiang.visualize.ml@gmail.com)



图 10. 比较平面直角坐标系坐标、重心坐标系坐标，任意形状三角形 | Bk\_2\_Ch23\_3.ipynb

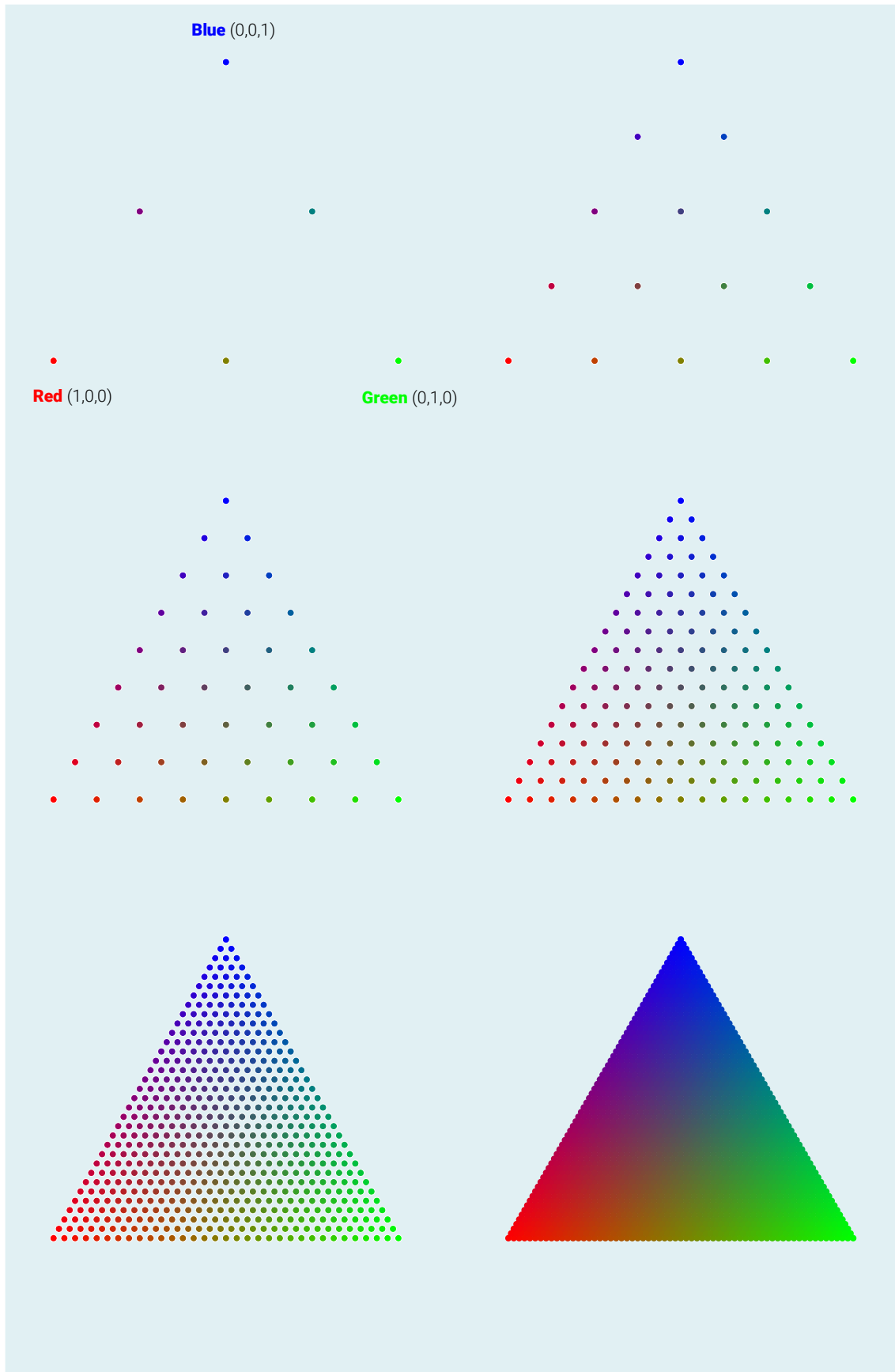

本 PDF 文件为作者草稿，发布目的为方便读者在移动终端学习，终稿内容以清华大学出版社纸质出版物为准。

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](mailto:jiang.visualize.ml@gmail.com)

图 11. 重心坐标系中混合红绿蓝 |  Bk\_2\_Ch23\_4.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](mailto:jiang.visualize.ml@gmail.com)



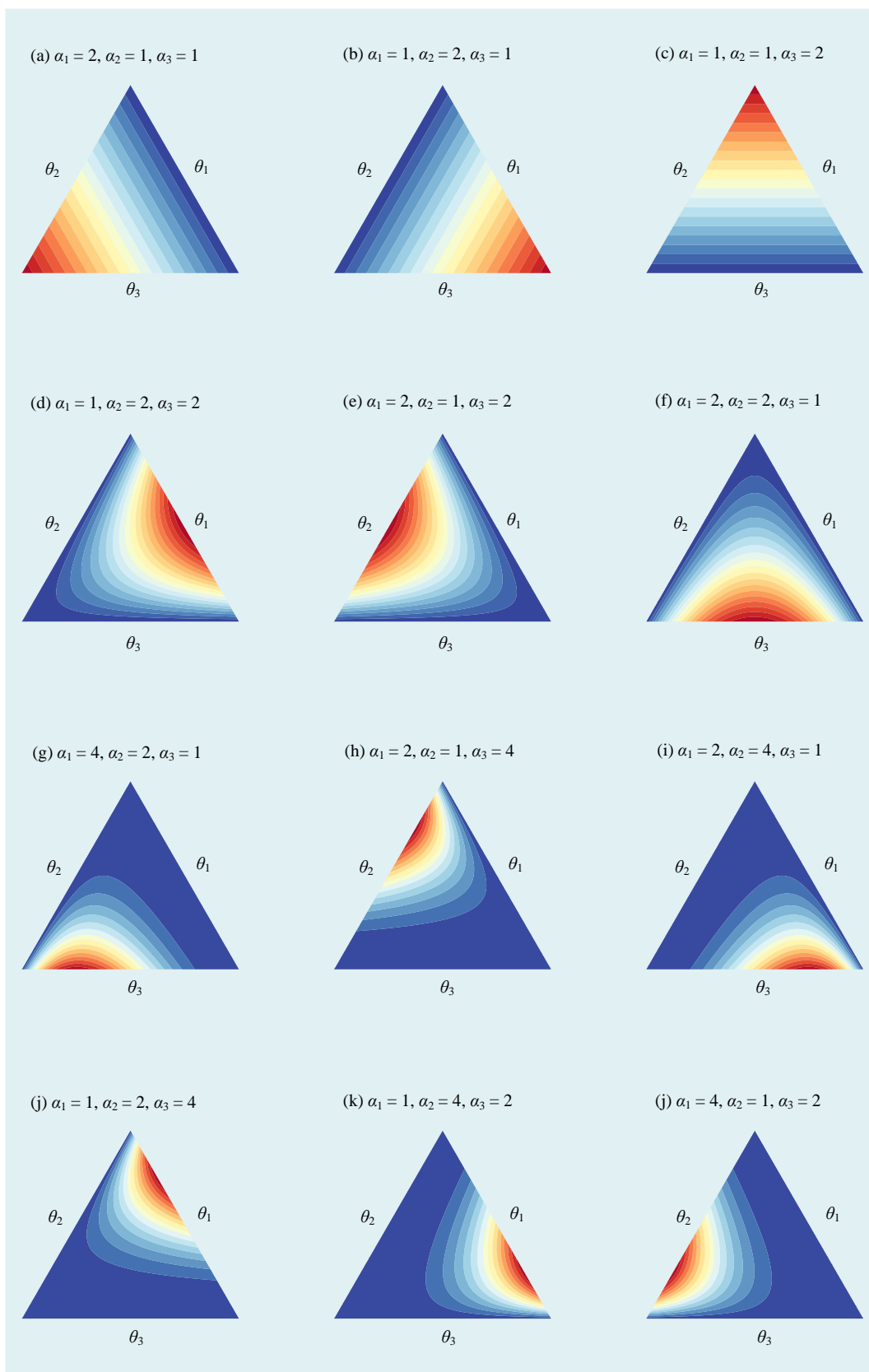


图 12. Dirichlet 分布, 第 1 组 | Bk\_2\_Ch23\_5.ipynb

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](mailto:jiang.visualize.ml@gmail.com)

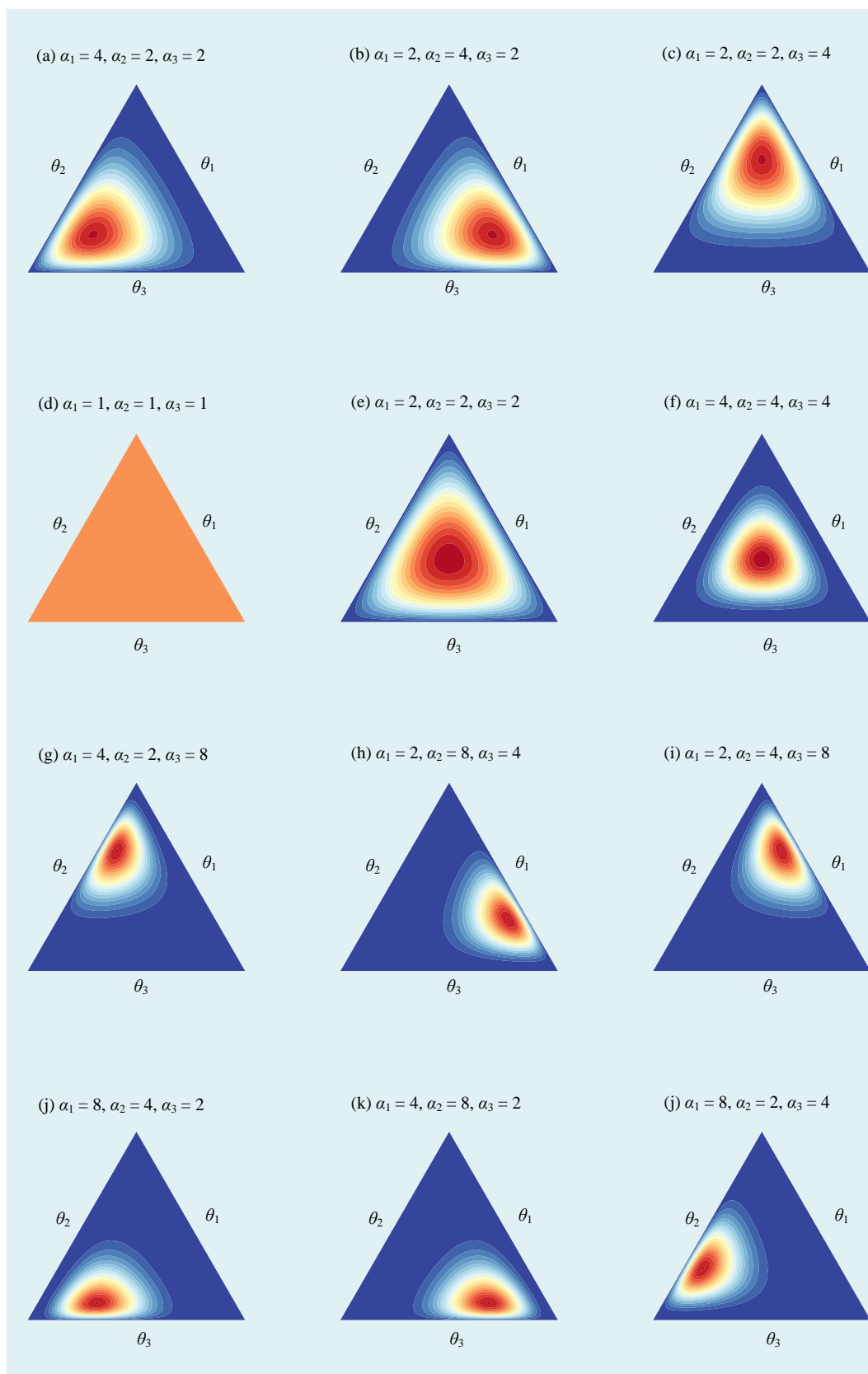


图 13. Dirichlet 分布, 第 2 组 | Bk\_2\_Ch23\_5.ipynb

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](mailto:jiang.visualize.ml@gmail.com)

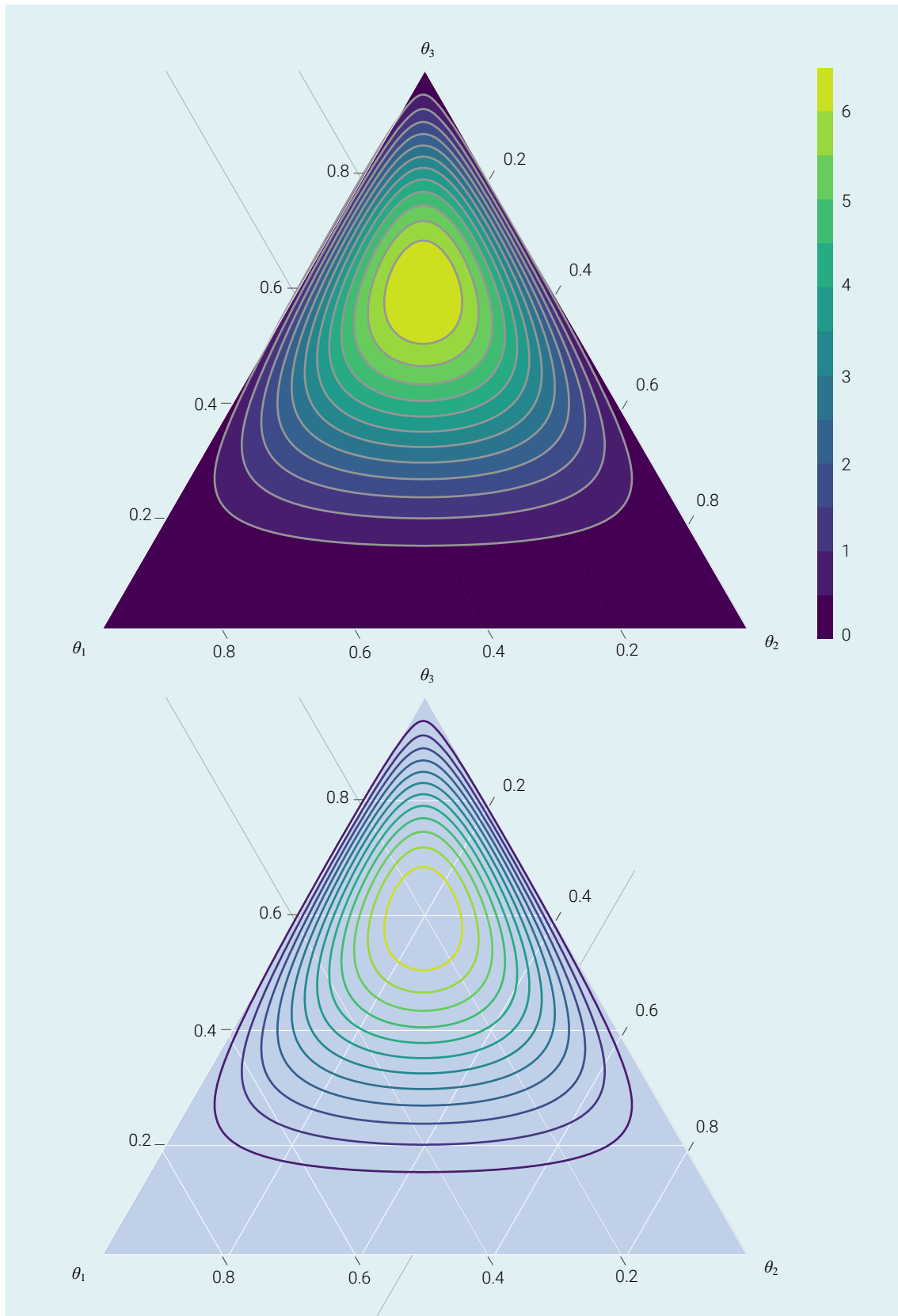



图 14. 用 Plotly 可视化 Dirichlet 分布 |  Bk\_2\_Ch23\_6.ipynb

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](mailto:jiang.visualize.ml@gmail.com)