

## 11

## Heatmap and Others

## 热图和其他

鸢尾花书中常用来可视化矩阵运算



每个孩子都是艺术家。问题在于他长大后如何保持艺术家的本质。

*Every child is an artist. The problem is how to remain an artist once he grows up.*

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



```

< numpy.linalg.cholesky() Cholesky 分解
< numpy.linalg.eig() 特征值分解
< numpy.linalg.svd() 奇异值分解
< numpy.zeros_like() 用来生成和输入矩阵形状相同的零矩阵
< seaborn.clustermap() 绘制聚类热图
< seaborn.heatmap() 绘制热图
< sklearn.datasets.load_iris() 加载鸢尾花数据
< matplotlib.image.imread() 读取图像文件并返回对应的图像数据
< matplotlib.pyplot.hist() 绘制直方图
< matplotlib.pyplot.imshow() 显示图像数据
< numpy.zeros() 返回给定形状和类型的新数组，用零填充
< numpy.zeros_like() 用来生成和输入矩阵形状相同的零矩阵
< skimage.color.rgb2gray() 将彩色图像转换为灰度图像
< skimage.io.imread() 读取图像文件并返回对应的图像数据

```



热图和其他

热图

伪彩色网格图

非矢量图片

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)

## 11.1 热图

### Seaborn 中的热图

**热图** (heatmap)，也叫热力图，是“鸢尾花书”中极为常见的可视化手段。特别是在展示数据、矩阵分解时，我们常用热图可视化矩阵。

虽然，matplotlib 中也有绘制热图的工具；但是，推荐大家使用 seaborn 中的 heatmap 函数。这个函数绘制热图更方便。

Seaborn 是一款基于 matplotlib 的数据可视化库，其中包括了各种绘图函数，其中之一就是 heatmap。使用 Seaborn 的 heatmap 函数可以让大家快速而方便地可视化矩阵数据，使得数据分析更加直观和易于理解。

热图可以用于可视化二维数组。图 1 所示为用热图可视化鸢尾花四个量化特征数据。在 Jupyter notebook 中，大家可以看到我们用 cmap 控制色谱，用 xticklabels、yticklabels 分别控制横轴、纵轴标签，用 cbar\_kws 设置色谱条位置，并用 vmin、vmax 控制色谱条起止位置。

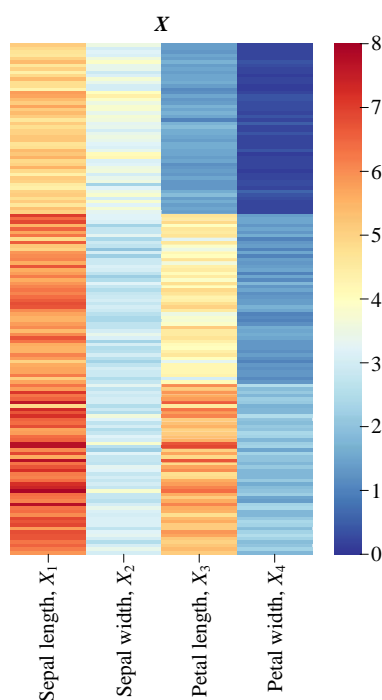


图 1. 热图可视化鸢尾花数据 |  BK\_2\_Ch11\_01.ipynb

Seaborn 中的 heatmap 函数还包括许多其他参数，用于自定义热图的外观和行为。例如，大家可以使用 annot 参数在热图中显示数值，使用 fmt 参数指定数字格式，使用 linewidths 参数调整单元格边框宽度等等。

### 聚类热图

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)

Seaborn 中，`clustermap` 是一个用于绘制聚类热图的函数，其原理是将矩阵中的行和列进行聚类，并以聚类后的顺序重新排列矩阵的行和列。这样可以将具有相似特征的行和列放在一起，从而更容易地发现它们之间的相似性和差异性。图 2 所示为鸢尾花数据的聚类热图。

➡ “鸢尾花书”的《机器学习》将专门讲解各种聚类算法。

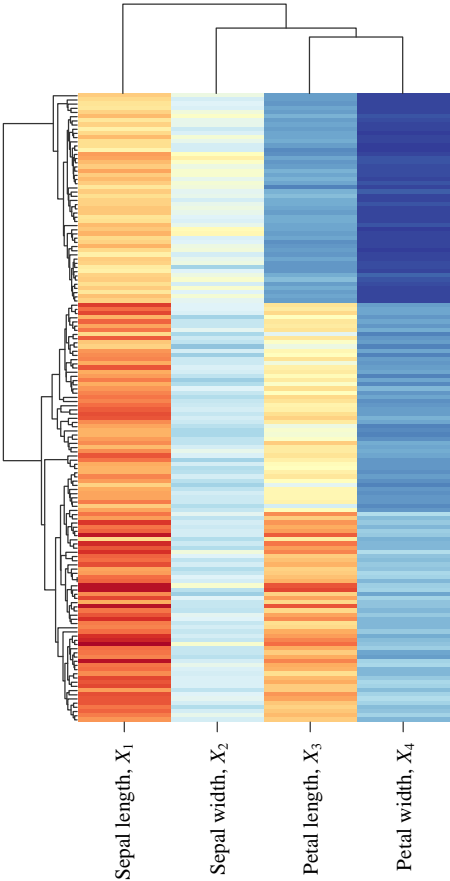


图 2. 热图可视化鸢尾花数据 |  BK\_2\_Ch11\_01.ipynb

矩阵运算

“鸢尾花书”中，大家会经常看到用一组热图可视化矩阵运算，特别是矩阵分解。图 15 所示为常见的几个矩阵运算。注意，后期制作时，热图的形状做了修改。

➡ 《矩阵力量》一册将从代数、数据、线性组合、优化、几何、统计等角度和大家讨论这些矩阵运算。此外，大家还会看到我们用热图可视化协方差矩阵、相关性系数矩阵，以及这些矩阵对应的线性代数运算。本节就不再展开讨论了。

# 11.2 伪彩色网格图

在 Matplotlib 中，`pcolormesh` 函数用于创建一个伪彩色网格图，类似热图。它可以用于绘制二维数据的色彩填充图，其中每个数据点的颜色根据其对应的数值进行映射。

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)

在 `pcolormesh` 函数中，可以使用 `rasterized` 参数来控制是否将图形渲染为矢量图形或光栅图像。`rasterized` 参数是一个布尔值，用于指定是否将图形渲染为光栅图像。当设置为 `True` 时，图形将以光栅化的形式保存，这对于包含大量数据点或复杂图形时可以提高渲染性能和文件大小。默认情况下，`rasterized` 参数的值为 `False`，即图形以矢量格式渲染。

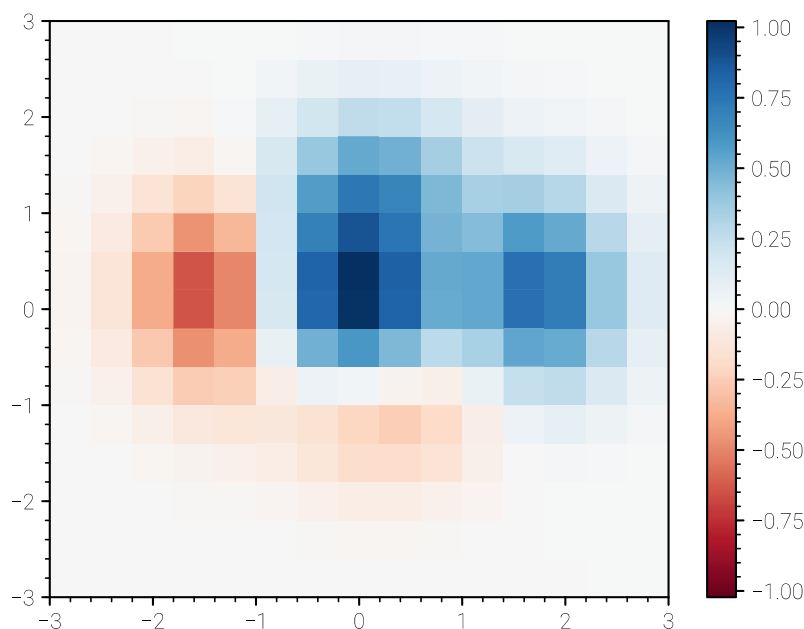


图 3. 利用 `pcolormesh` 绘制的伪彩色图 | [BK\\_2\\_Ch11\\_02.ipynb](#)

如图 16 所示，`pcolormesh` 函数还常用来绘制分类算法的决策边界。此外，`pcolormesh` 函数可以绘制网格，并用来可视化线性、非线性变换，具体如图 4、图 17 所示。

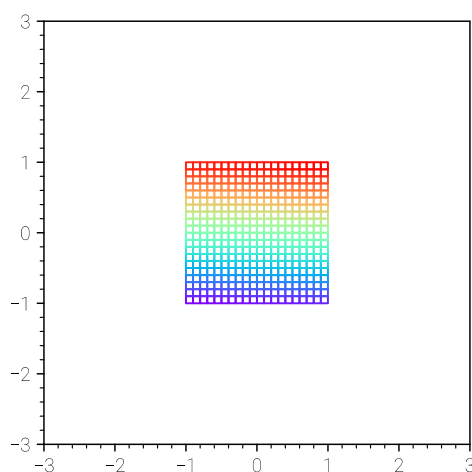


图 4. 利用 `pcolormesh` 绘制方正网格 | [BK\\_2\\_Ch11\\_04.ipynb](#)

`pcolor` 函数也是 `matplotlib` 库中的函数，用于绘制伪彩色图，效果和 `pcolormesh` 类似。与 `pcolor` 相比，`pcolormesh` 在效率上更高，特别适用于绘制大型数据集。

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)

## 11.3 非矢量图片

本章最后再聊聊非矢量图片。`matplotlib.image` 模块提供了读取和处理图像的函数，其中最常用的函数是 `imread`。`imread` 函数可以读取图像文件，并将其解码为一个三维的 `numpy` 数组。

`matplotlib.pyplot.imshow()` 是 `matplotlib` 中用于显示图像的函数。将如图 5 所示鸢尾花照片导入后，容易发现这幅图像实际上是一个  $2990 \times 2714 \times 3$  的数组。

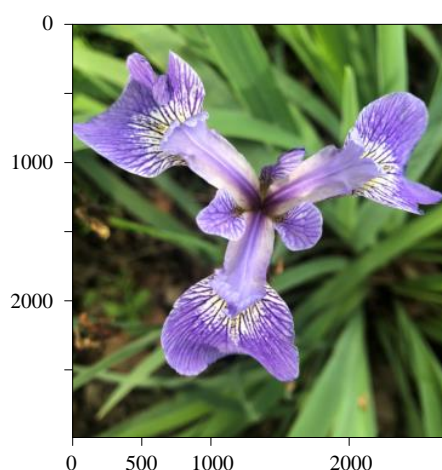


图 5. 鸢尾花照片 | `BK_2_Ch11_05.ipynb`

图片**像素** (pixel) 是图片的基本单位，是构成图片的最小元素。它是一个有限的、离散的、二维的点，有着特定的位置、颜色和亮度值。在数字图像中，每个像素都有一个确定的坐标和值。图片中的像素数量越多，图片的分辨率就越高，图片的清晰度和细节也就越好。

像素的颜色通常使用 RGB 值（红、绿、蓝三种颜色的强度组合）表示。每个像素都有一个红、绿、蓝三个通道的值。红、绿、蓝可以分别被编码为一个数字，例如 8 位的数字可以表示 256 种颜色。

也就是说，图 5 这幅图中每个像素首先分解成红绿蓝三个数值。这些数值的取值范围都在  $[0, 255]$  之间。换个角度，图 5 可以理解成是由三幅图片叠加而成，如图 6 所示。

此外，我们可以获得如图 7 所示的红绿蓝颜色的分布。越靠近 0，颜色越靠近黑，越靠近 255 颜色越靠近纯色。本书前文已经和大家聊过  $[0, 0, 0]$  代表纯黑， $[255, 255, 255]$  代表纯白。注意，在 `matplotlib` 中  $[1, 1, 1]$  代表纯白。

在彩色图像中，每个像素的颜色可以由三个 8 位数字（红、绿、蓝）组成，因此彩色图像中的每个像素可以表示  $2^{3 \times 8}$  种不同的颜色，约为 1600 万种。

在数字图像处理中，对图像进行各种操作，例如缩放、旋转、裁剪、调整亮度和对比度等，都会涉及到像素的处理和修改。

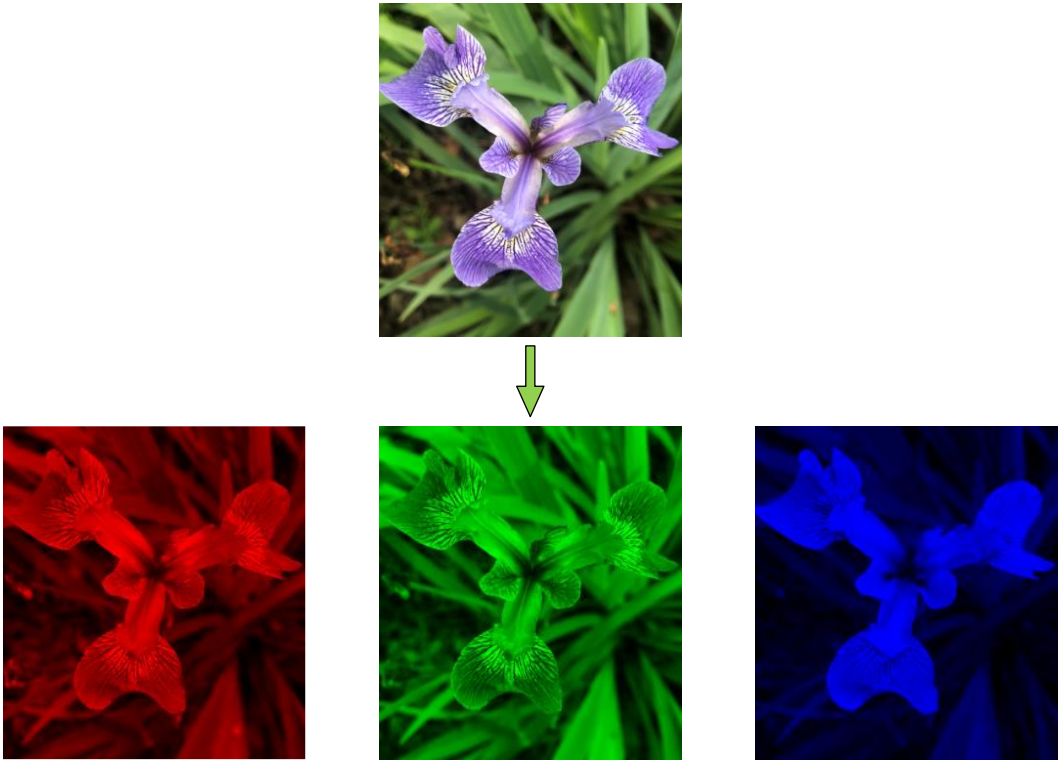



图 6. 鸢尾花照片分解成红绿蓝三个通道 |  BK\_2\_Ch11\_05.ipynb

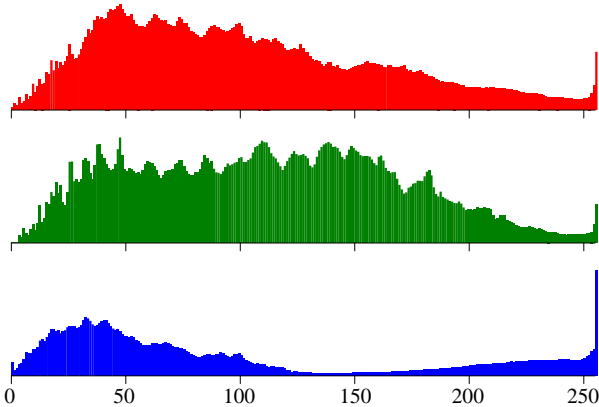



图 7. 鸢尾花照片红绿蓝颜色分布

### 红绿蓝三个通道

图 8 给出的三幅子图，每幅图仅保留两色通道，另外一个通道数值全部置零。

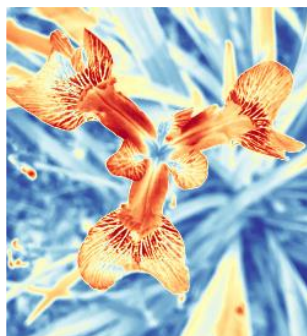




图 8. 鸢尾花照片，只保留两色通道 |  BK\_2\_Ch11\_05.ipynb

## 色谱

它可以用来显示二维数组或图像文件中的图像。`imshow` 函数有很多参数可以控制图像的外观。例如，可以使用 `cmap` 参数指定要使用色谱。图 9 所示为使用色谱展示红色通道。Jupyter notebook 中还给出更多范例。

图 9. 使用色谱展示红色通道 |  BK\_2\_Ch11\_05.ipynb

## 灰度

`Scikit-image (skimage)` 是一个用于图像处理和计算机视觉的 Python 包。它提供了一系列算法，函数和工具，可用于图像处理，包括图像滤波，几何变换，色彩空间转换，图像分割，特征提取等等。具体来说，`skimage` 可以用于：a) 加载和保存图像；b) 调整图像大小，旋转，裁剪等几何变换；c) 进行图像滤波和增强；d) 在不同颜色空间之间进行转换；e) 检测边缘和角点；f) 进行图像分割和分析；g) 进行特征提取和图像匹配。

图 10 所示为使用 `skimage` 将彩色图片转化为灰度图片。注意图片的每个像素的取值在  $[0, 1]$  之间。此外，图像识别一般都使用灰度图像。

图 10. 将彩色图片转化成灰度 |  BK\_2\_Ch11\_05.ipynb

## 修改部分像素

由于图片本身就是一个数组，我们可以通过修改数组的具体值来修改图片。如所示，我们将灰度照片的左上角  $500 \times 500$  的像素变为白色。

图 11. 修改图片像素 | [BK\\_2\\_Ch11\\_05.ipynb](#)

## 降低像素

图 12 所示为通过采样降低图像像素。图 5 这幅图片的像素大小为  $2990 \times 2714$ 。每 200 个像素采样一个像素，我们便得到图 12。这幅图的像素为  $15 \times 14$ ，很明显图片的颗粒度很粗糙。

图 12. 采样降低像素 | [BK\\_2\\_Ch11\\_05.ipynb](#)

当图像像素较低时，为了让图片看上去更细腻，我们可以采用插值。

## 插值

`imshow()` 函数中，我们可以通过设置 `interpolation` 参数来控制如何在图像像素之间进行插值，以生成更平滑的图像。

`imshow()` 函数 `interpolation` 参数的默认值是 `'antialiased'`，它使用反走样技术来平滑图像，使其在缩放时更加清晰。这意味着在缩放图像时，`imshow()` 函数会自动对图像进行插值，以获得更平滑的外观。

除了默认的 `'antialiased'` 插值，`imshow()` 函数还支持其他插值方法，包括 `'nearest'`、`'bilinear'`、`'bicubic'` 等。这些插值方法可以通过 `interpolation` 参数来设置。例如，`'nearest'` 插值只是在最近的像素值之间进行插值，而 `'bicubic'` 插值使用更复杂的算法来生成更平滑的图像。图 13 所示为图 12 的两种插值结果。本节的 Jupyter notebook 中给出更多插值方法。



《数据有道》一册将详细讲解常见插值算法。

选择不同的插值方法会影响图像的视觉效果，因此选择合适的插值方法可以使图像更清晰或更平滑，更符合数据的视觉表达。



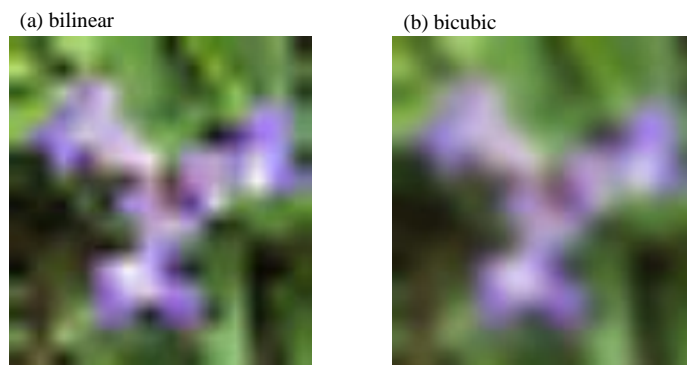


图 13. 插值平滑 | BK\_2\_Ch11\_05.ipynb

## 仿射变换

图 14 所示为对图片采取各种仿射变换。本章后续将专门介绍各种平面、立体几何变换。

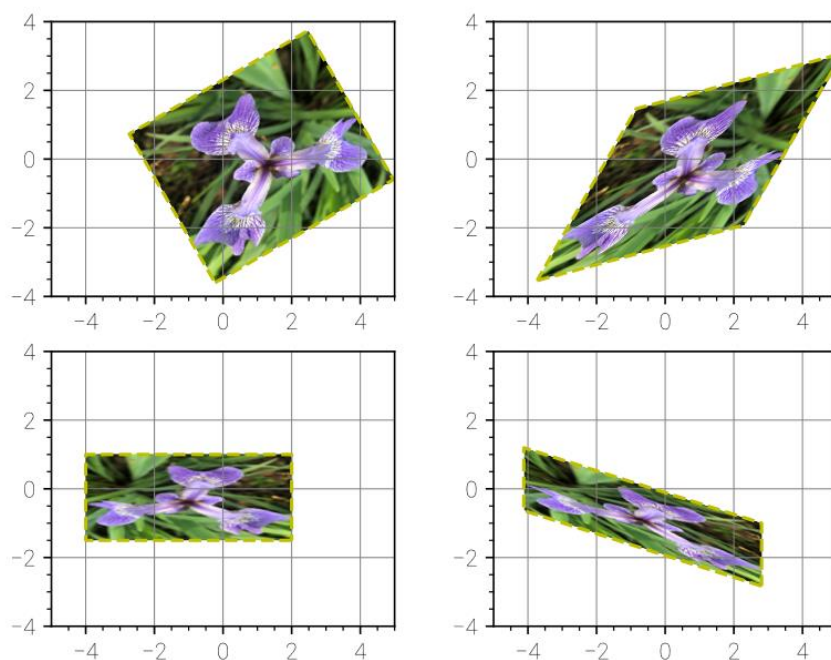
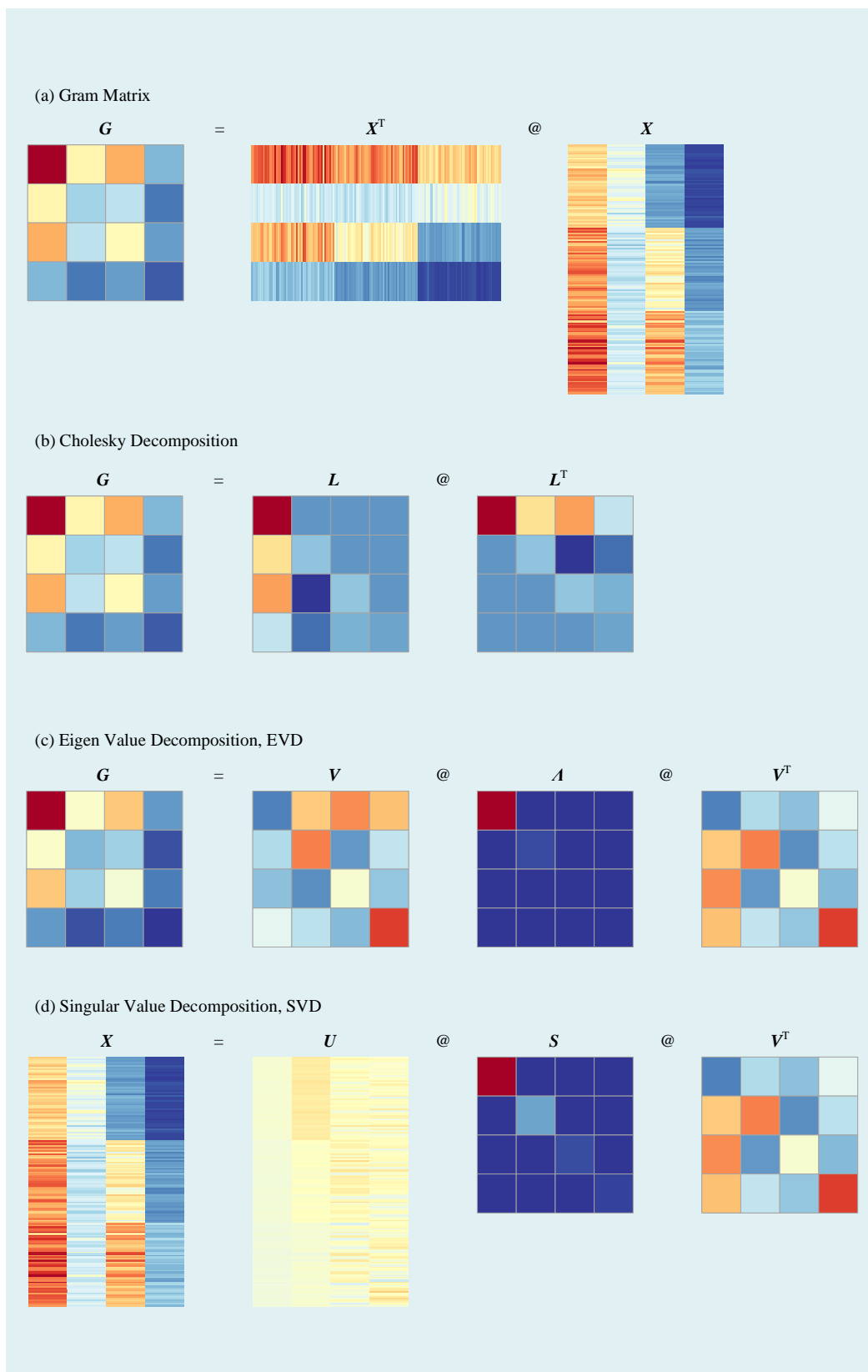



图 14. 图片的仿射变换 | BK\_2\_Ch11\_05.ipynb

本章介绍了三种可视化方案，热图、伪彩色网格图、非矢量图片。鸢尾花书常用 `seaborn` 中的 `heatmap` 展示各种矩阵运算，需要大家格外留意。

图 15. 用热图可视化矩阵运算 |  BK\_2\_Ch11\_01.ipynb

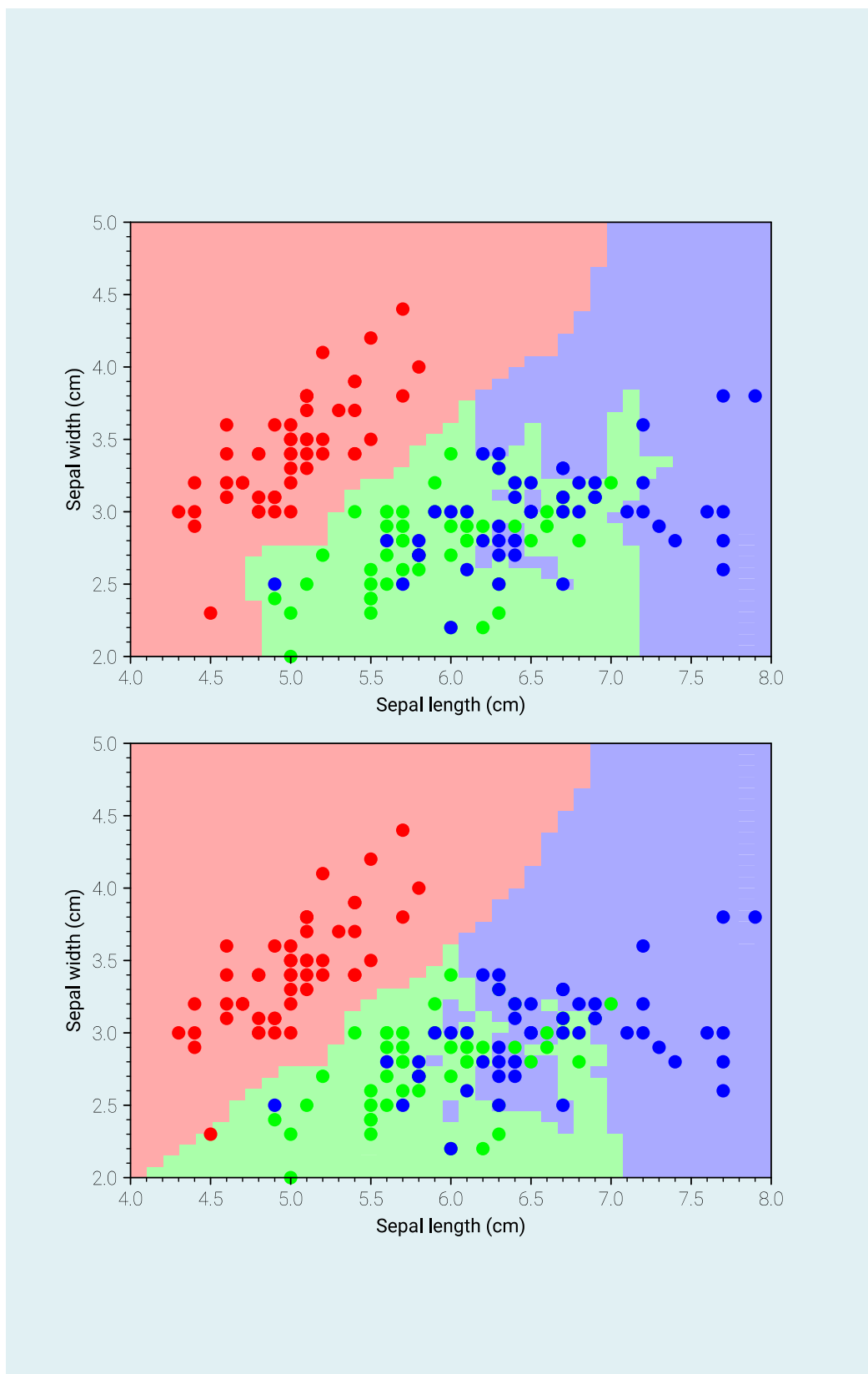



图 16. 用 `pcolormesh` 函数绘制分类决策边界 |  BK\_2\_Ch11\_03.ipynb

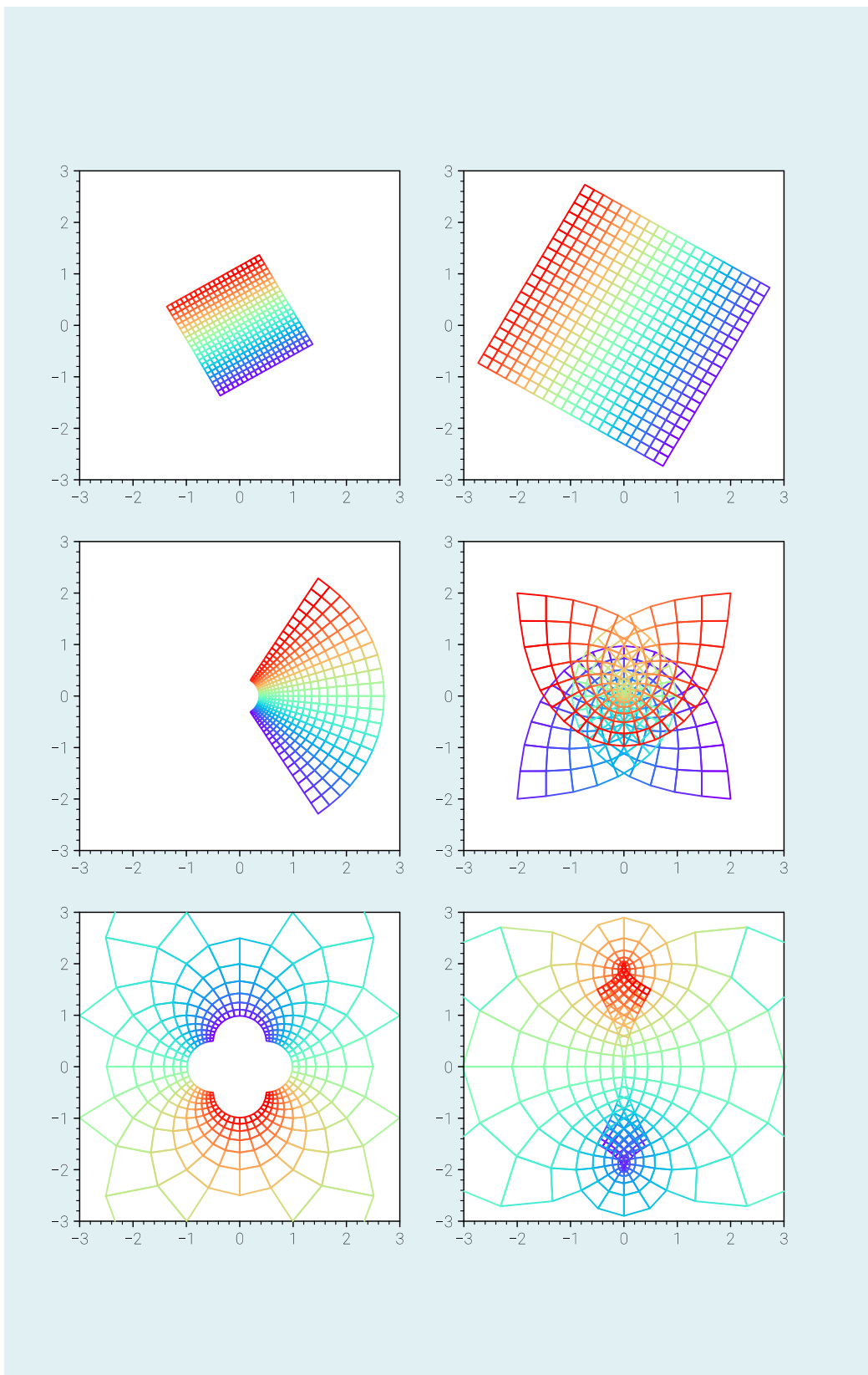



图 17. 用 `pcolormesh` 函数可视化线性、非线性变换 |  BK\_2\_Ch11\_04.ipynb

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