

8

2D Line Plots

平面线图

实际上也是散点顺序相连的折线图



艺术的目的不在于展示事物外在的美，而是其内在的价值。

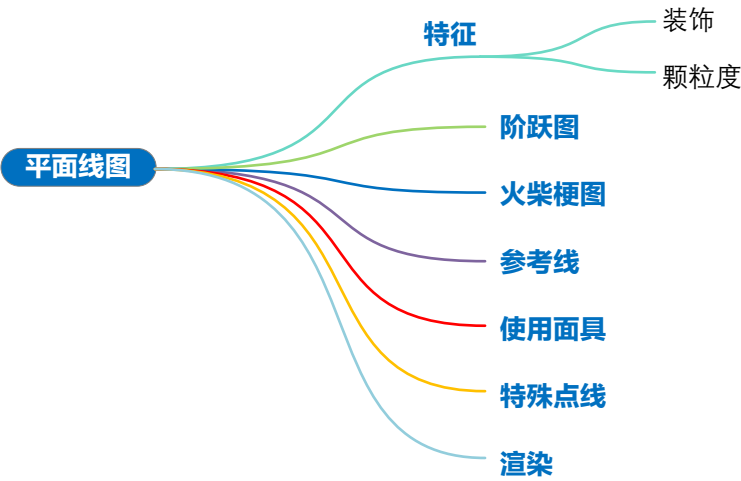
The aim of art is to represent not the outward appearance of things, but their inward significance.

—— 亚里士多德 (Aristotle) | 古希腊哲学家 | 384 ~ 322 BC



- ◀ matplotlib.collections.LineCollection() 是 Matplotlib 中的一个集合对象，用于绘制多条线段的集合
- ◀ matplotlib.pyplot.axhline() 绘制水平线
- ◀ matplotlib.pyplot.axvline() 绘制竖直线
- ◀ matplotlib.pyplot.Normalize() 函数是用于将数据归一化或标准化到指定的范围内的函数
- ◀ matplotlib.pyplot.stem() 绘制火柴梗图
- ◀ numpy.arange() 根据指定的范围以及设定的步长，生成一个等差数组
- ◀ numpy.argwhere() 返回一个数组中满足指定条件的元素的索引
- ◀ numpy.concatenate() 将多个数组进行连接
- ◀ numpy.cumsum() 计算累计求和
- ◀ numpy.linspace() 在指定的间隔内，返回固定步长的数据
- ◀ numpy.log() 底数为 e 自然对数函数
- ◀ numpy.log10() 底数为 10 对数函数
- ◀ numpy.log2() 底数为 2 对数函数
- ◀ numpy.random.normal() 生成满足高斯分布的随机数
- ◀ numpy.sign() 函数返回一个数组中每个元素的符号值
- ◀ numpy.sin() 计算正弦值
- ◀ numpy.vstack() 返回竖直堆叠后的数组
- ◀ zip(*) 用于将可迭代的对象作为参数，将对象中对应的元素打包成一个个元组，然后返回由这些元组组成的列表。*代表解包，返回的每一个都是元祖类型，而并非是原来的数据类型





8.1 点动成线

点动成线，线动成面。散点顺序连线的结果就是线图，所以用 Python 第三方库绘制的曲线本质上也是折线图。

线条装饰

鸢尾花书《编程不难》介绍过，在用 `matplotlib.pyplot.plot()` 绘制平面线图时，我可以调整粗细 (`linewidth, lw`)、样式 (`linestyle, ls`)、颜色 (`color, c`)、标记 (`marker`)、透明度 (`alpha`) 等等。这些内容相对简单，本章不再重复。

有关线条样式，请大家参考。

https://matplotlib.org/stable/gallery/lines_bars_and_markers/linestyles.html

标记 `marker` 还可以调整其大小 (`markersize, ms`)、边缘颜色 (`markeredgecolor, mec`)、边缘线宽度 (`markeredgewidth, mew`)、标记填充颜色 (`markerfacecolor, mfc`) 等等。此外，我们还可以用 `markevery` 这个参数显示特定数据点，请大家参考。

https://matplotlib.org/stable/gallery/lines_bars_and_markers/markevery_demo.html

值得一提的是图 1。大家在《编程不难》已经见过这幅图。在绘制线图时，如果不指定具体颜色，在绘制若干线图时，会采用如图 1 右侧由上至下颜色依次渲染。颜色不够用时，重复颜色序列循环。如果大家不满意这些默认颜色，可以对轴对象采用 `ax.set_prop_cycle()` 方法来修改颜色序列循环，比如 `ax.set_prop_cycle(color=['red', 'orange', 'yellow'])`。

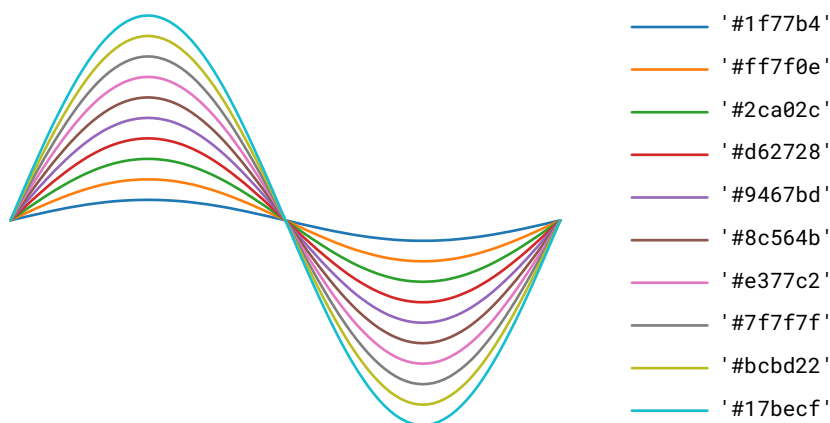


图 1. Matplotlib 线图默认颜色序列

颗粒度

绘制线图时，大家首先注意**颗粒度** (`granularity`)，即采样。

多数情况，在绘制一元函数线图时，我们用 `numpy.linspace()` 生成自变量的等差数列。图 2 的两幅图中的散点都来自于正弦函数 $f(x) = \sin(x)$ 。显然，颗粒度粗糙时，用线图可视化一元函数可能会误导读者。

等差数列的公差越小，曲线的颗粒度越高，这样平面线图看上去“光滑”。如图 3 (a) 所示，等差数列有 101 个元素。将这些散点顺序连接便得到图 3 (b)。对于 $f(x) = \sin(x)$ 这个并不复杂的一元函数，图 3 (a) 的颗粒度显然足够用了。

如图 4 所示，为了可视化 $f(x) = \sin(1/x)$ 在靠近 0 附近的振荡，我们需要极其细腻的颗粒度。

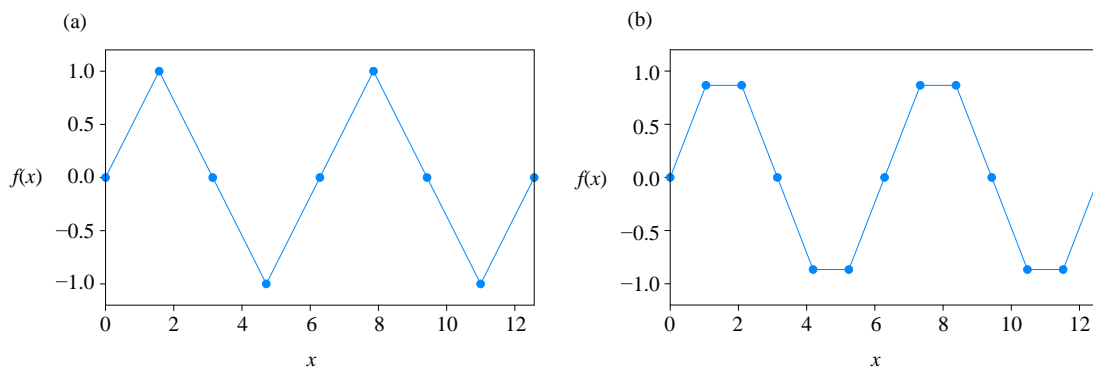


图 2. 颗粒度粗糙 | [BK_2_Ch08_1.ipynb](#)

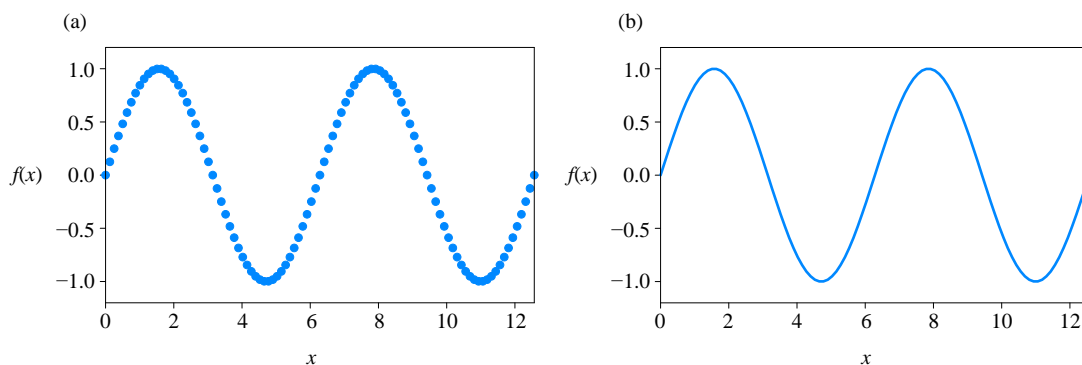
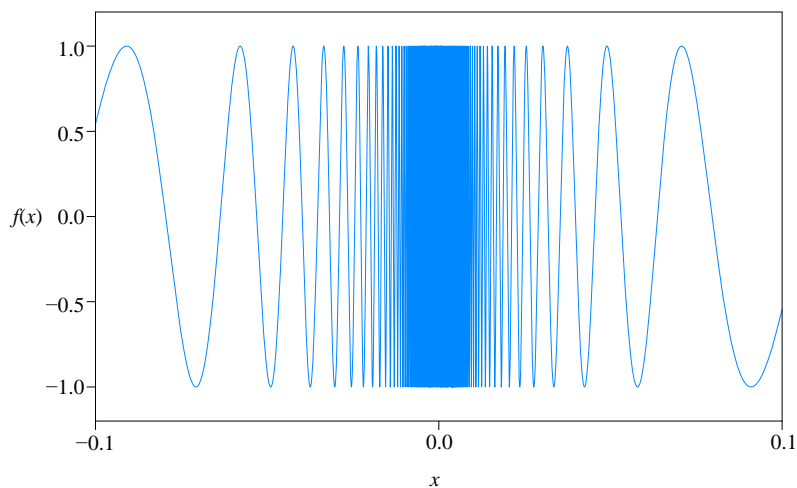


图 3. 颗粒度合适 | [BK_2_Ch08_1.ipynb](#)




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 4. 特殊函数需要极其细腻的颗粒度 |  BK_2_Ch08_1.ipynb

但是，颗粒度过高也不可取，也就是等差数列的公差过小，会增大计算量。这一点在一元函数上并不明显，但是用 `numpy.meshgrid()` 生成网格时，大家就会发现维数灾难（curse of dimensionality）。

维数灾难是指在高维空间中，数据变得非常稀疏，而且距离变得非常远，使得许多常用的数据分析技术和算法无法有效地处理和分析数据。通俗点讲，假设我们有一个只有两个特征（比如，鸢尾花花萼长度、宽度）的数据集，我们可以很容易地将其可视化成二维平面上的点。但是如果我们有许多特征，比如几百个，那么我们将无法在三维或更高维空间中可视化数据。

此外，保持每个特征的采样数量，当维度增加时会导致数据量急剧增长。比如，单一维度的采样点数为 100，两个特征的网格点数就变成了 10000 (100^2)，三个特征的网格点数就增大到了惊人的 1000000 (100^3)。本书后文还会遇到这个问题。

注意，如果绘图采用对数坐标，建议采用 `numpy.logspace()` 生成数列。



Jupyter 笔记 BK_2_Ch08_1.ipynb 绘制图 2、图 3、图 4。

8.2 阶跃图

再次强调，在绘制线图时，默认散点之间两点顺序连线。这就意味着，任意顺序两点之间的线段是通过线性插值方法得到。

但是，有很多场合，我们需要避免“线性插值”，而采用阶跃方法绘制线图。

`matplotlib.pyplot.plot()` 函数本身可以设定阶跃绘图。此外，`matplotlib.pyplot.step()` 函数是专门绘制阶跃线图的函数。这个函数有三种设置：'pre'、'post'、'mid'。

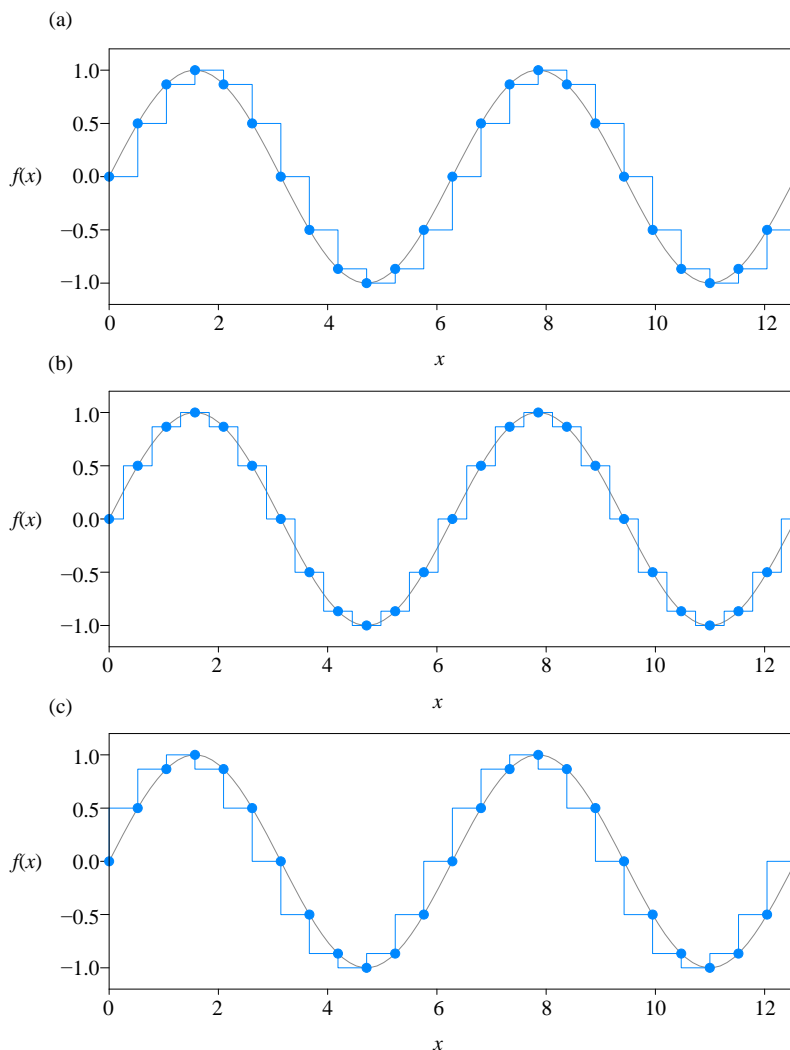


图 5. 三种阶跃 | BK_2_Ch08_2.ipynb

连接两点的插值方法有很多，《数据有道》第 5 章专门介绍。此外，本书后文会介绍贝塞尔曲线 (Bézier curve)。贝塞尔曲线是一种平滑曲线，在计算机图形学、工程和设计领域中广泛应用。

举个例子，贝塞尔二次曲线由三个控制点组成，其中两个控制点定义曲线的端点，第三个控制点定义曲线在端点之间的弯曲。



Jupyter 笔记 BK_2_Ch08_2.ipynb 绘制图 5。

8.3 火柴图

火柴图 (stem plot)，也称火柴梗图、脊柱图，常用来可视化离散数据序列和趋势。火柴图垂直线所在横轴位置代表样本点的位置，圆点纵轴高度表示样本点的值。

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

本系列图册中，火柴图常用来可视化数列、离散随机变量概率质量函数（Probability Mass Function, PMF），如图 6 所示。

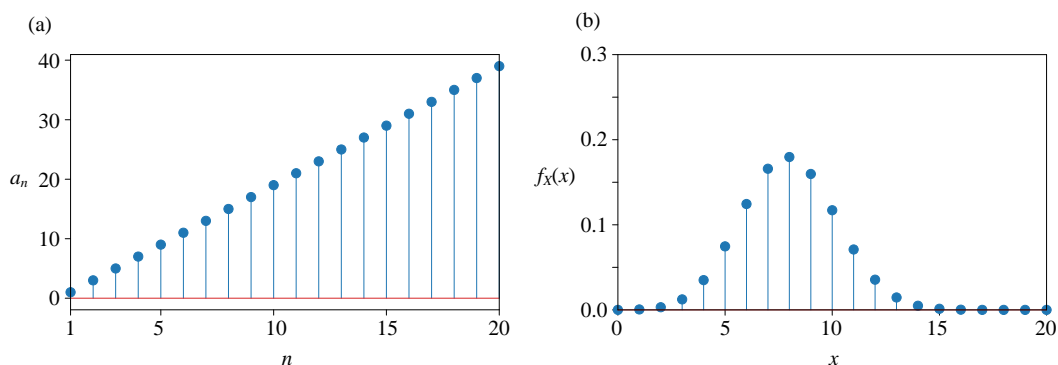



图 6. 火柴图可视化数列、概率质量函数 |  BK_2_Ch08_3.ipynb

图 6 的两个子图都可以看成是离散函数。而前文的 $f(x) = \sin(x)$ 则是连续函数。离散函数、连续函数的主要区别在于自变量取值方式不同。离散函数自变量只能取有限或可数无限个值。也就是说，离散函数的函数图像是一系列散点。

例如，一个函数 $f(x)$ 表示了投掷一枚骰子后得到点数。因为骰子点数是有限的，所以自变量 x 的取值为 1、2、3、4、5、6 这几个离散值。而连续函数的定义域是一个连续的区间，比如 $(-\infty, \infty)$ 、 $[0, 2]$ 。



Jupyter 笔记 BK_2_Ch08_3.ipynb 绘制图 6。

8.4 参考线

水平线图中，我们经常需要添加水平或竖直参考线。图 7 所示两种不同绘制参考线的方法。

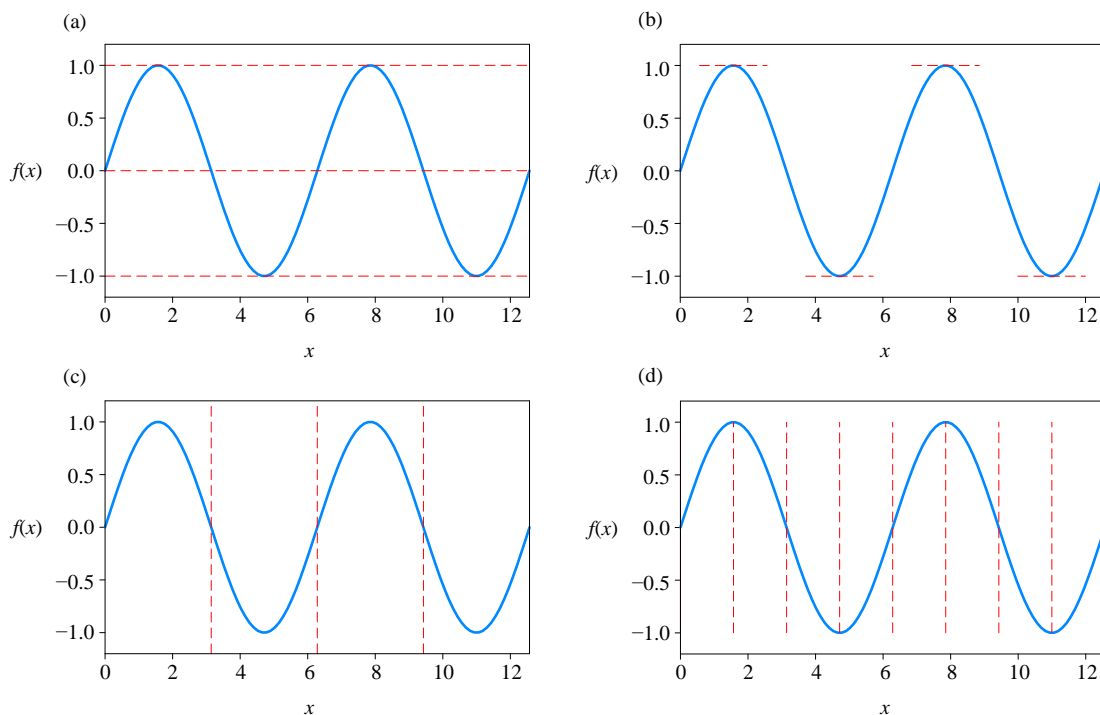


图 7. 两种绘制参考线的方法 | BK_2_Ch08_4.ipynb



Jupyter 笔记 BK_2_Ch08_4.ipynb 绘制图 7。

8.5 使用面具

图 8 所示使用面具 (mask) 分段渲染线图。采用的方法和上一章一致。

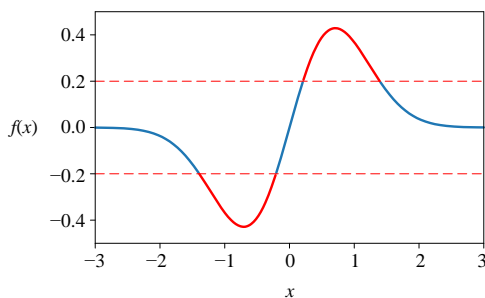


图 8. 分段渲染线图 | BK_2_Ch08_5.ipynb



Jupyter 笔记 BK_2_Ch08_5.ipynb 绘制图 8。

8.6 特殊点线

交点

如图 9 所示，通过寻找 $f_1(x) - f_2(x)$ 的正负号变号的位置，我们可以估计 $f_1(x)$ 、 $f_2(x)$ 的交点。

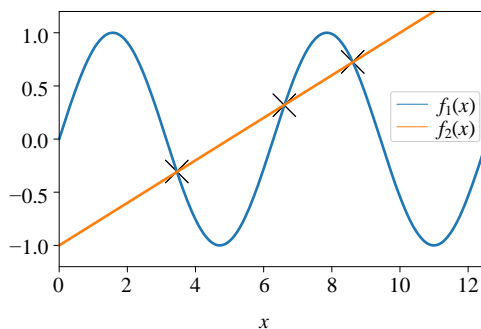



图 9. 可视化交点 |  BK_2_Ch08_6.ipynb



Jupyter 笔记 BK_2_Ch08_6.ipynb 绘制图 9。

极大、极小值

`numpy.argmax()`、`numpy.argmin()` 可以寻找数组中的极大、极小值，如图 10 所示。

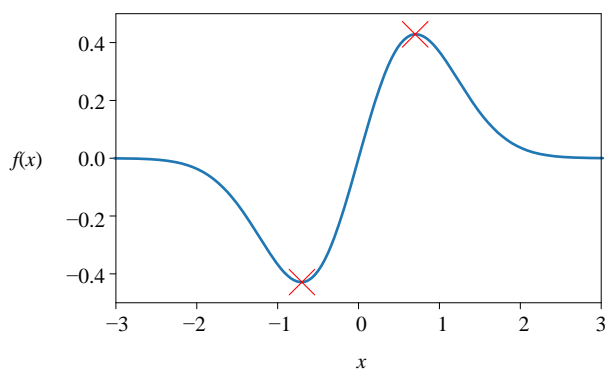


图 10. 可视化极值点 |  BK_2_Ch08_7.ipynb



Jupyter 笔记 BK_2_Ch08_7.ipynb 绘制图 10。

8.7 渲染

渲染一组曲线着色

图 13 所示为三种用色谱给一组曲线着色的方法。图 13 (a) 采用 `for` 循环，分别给每一条曲线着色。

调用 `RdYlBu` 色谱，用 `sigma` 数量产生若干连续色号。用 `for` 循环分别绘制每条曲线，曲线依次调用连续色号。

图 13 (b) 用 `LineCollection()` 分别渲染每条曲线，并添加色谱条展示 `sigma` 变化。

图 13 (c) 则用 `set_prop_cycle()` 修改默认线图颜色。



图 13 中曲线为一元高斯分布的概率密度函数。《统计至简》第 9 章专门讲解一元高斯分布。



Jupyter 笔记 BK_2_Ch08_8.ipynb 绘制图 13 子图。

分段渲染曲线

下面，我们用颜色映射和 `LineCollection()` 渲染一条曲线的不同分段。

如图 11 所示，我们先将一条线段打散成一系列线段。然后用 `LineCollection()`，用 `RdYlBu_r` 色谱分别给每条线段分别着色。这幅图中四副子图用来渲染的依据完全不同，请大家自行学习。

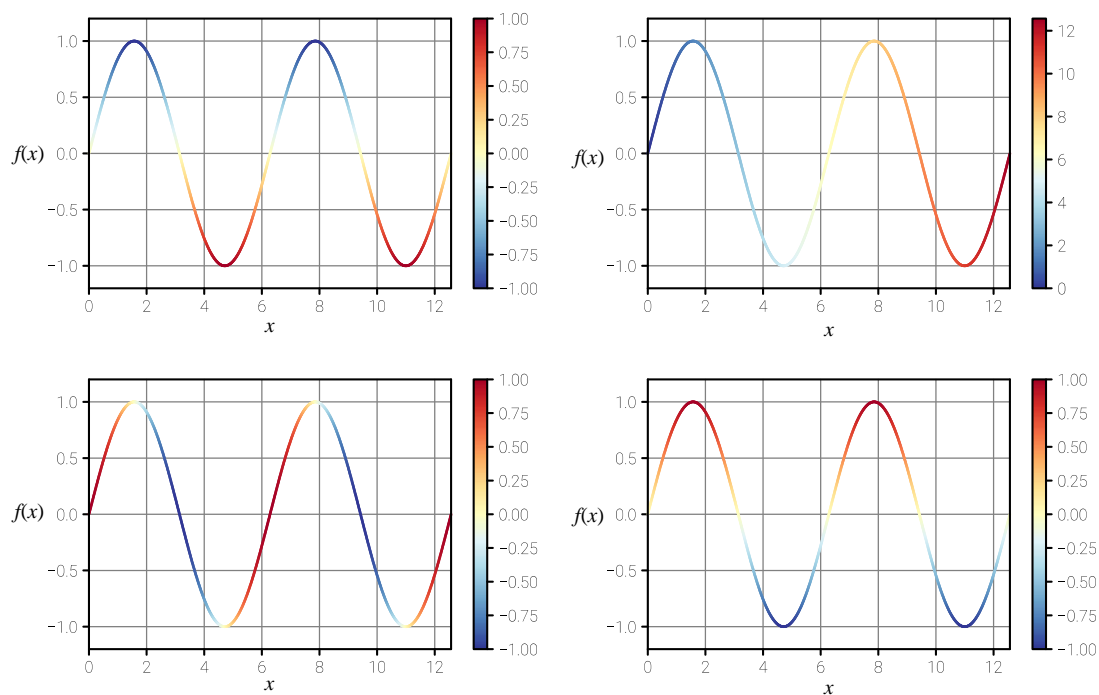


图 11. 分段渲染 | BK_2_Ch08_9.ipynb

绘制网格

图 12 所示为利用 `plot()` 两点连线绘制的正方方格，并采用 `rainbow` 色谱分段着色渲染。图 14 所示为在此基础上可视化线性、非线性变换。

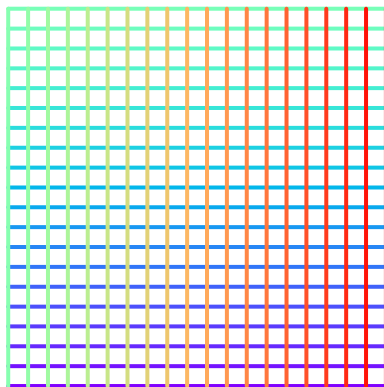
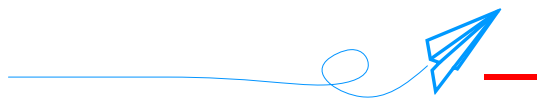


图 12. 用 `plot()` 绘制的网格，利用 `rainbow` 色谱渲染 | [BK_2_Ch08_10.ipynb](#)

图 16 和图 17 所示为利用线条创建的生成艺术。这些图背后的数学工具实际上是线性插值。



本章总结了 Matplotlib 中绘制线图的各种技巧。Plotly 和 Seaborn 也有绘制线图的函数，请大家自行学习探索。

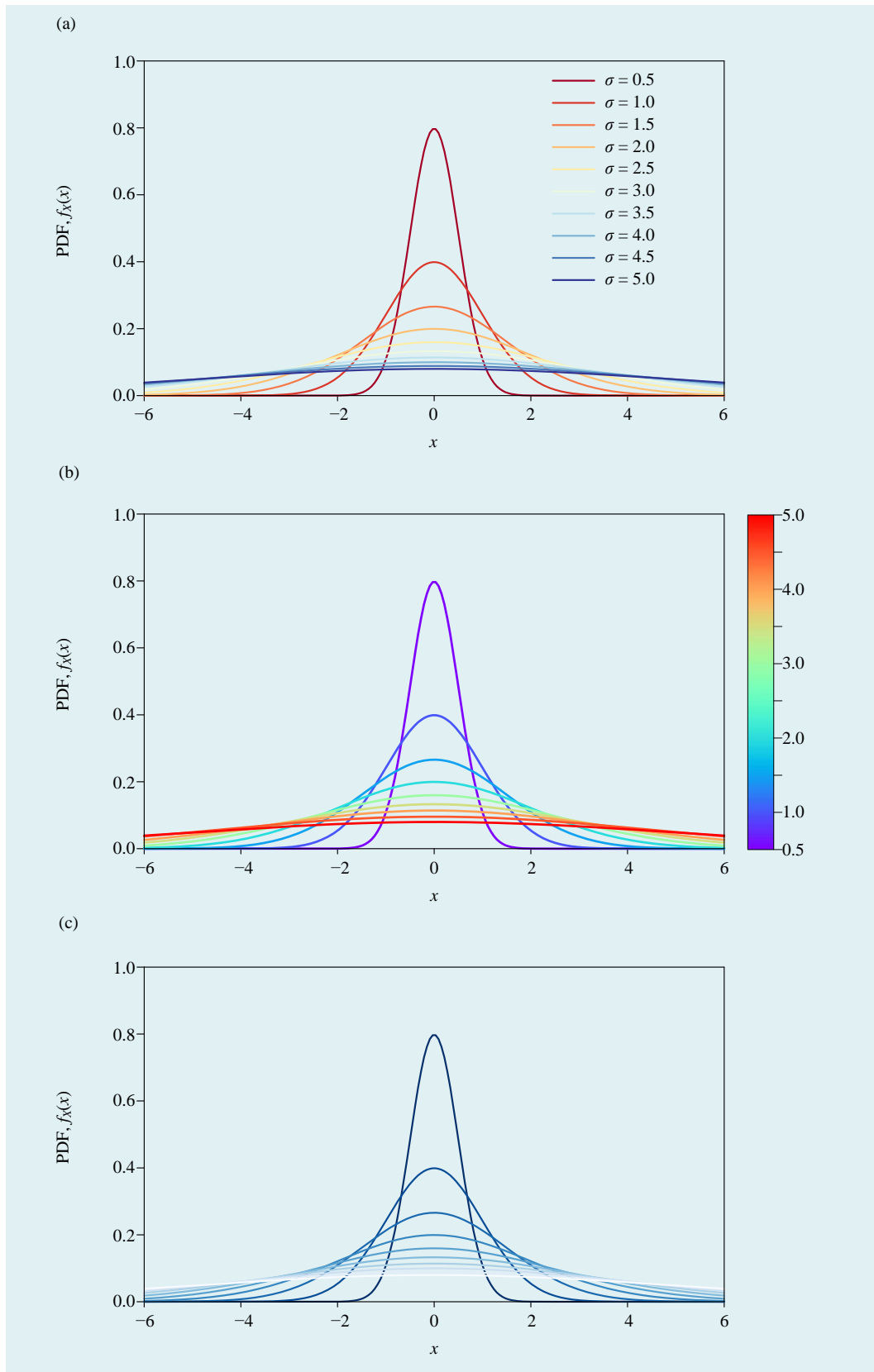
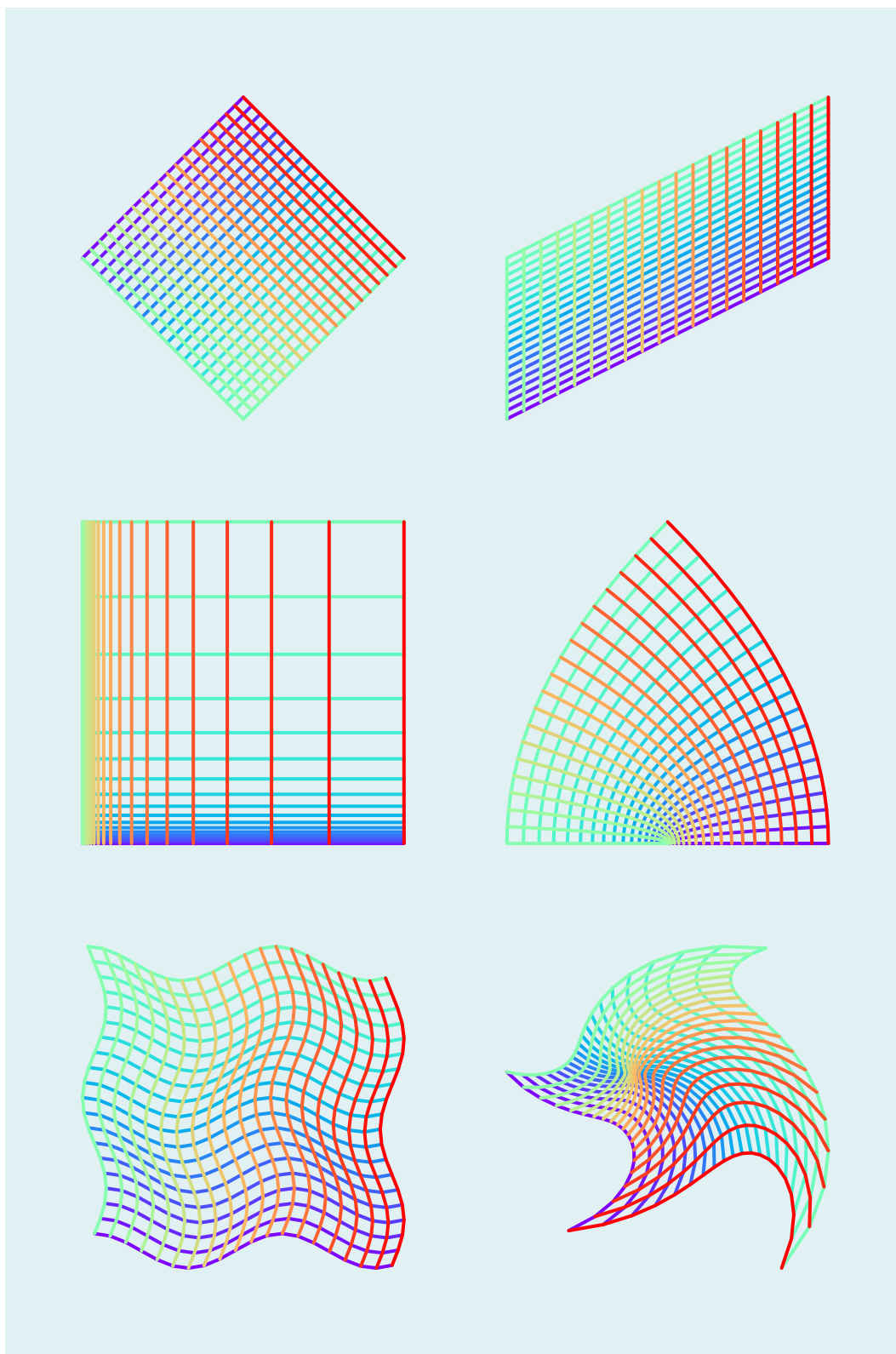



图 13. 用色谱渲染曲线 | BK_2_Ch08_8.ipynb

图 14. 可视化线性、非线性变换 |  BK_2_Ch08_10.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

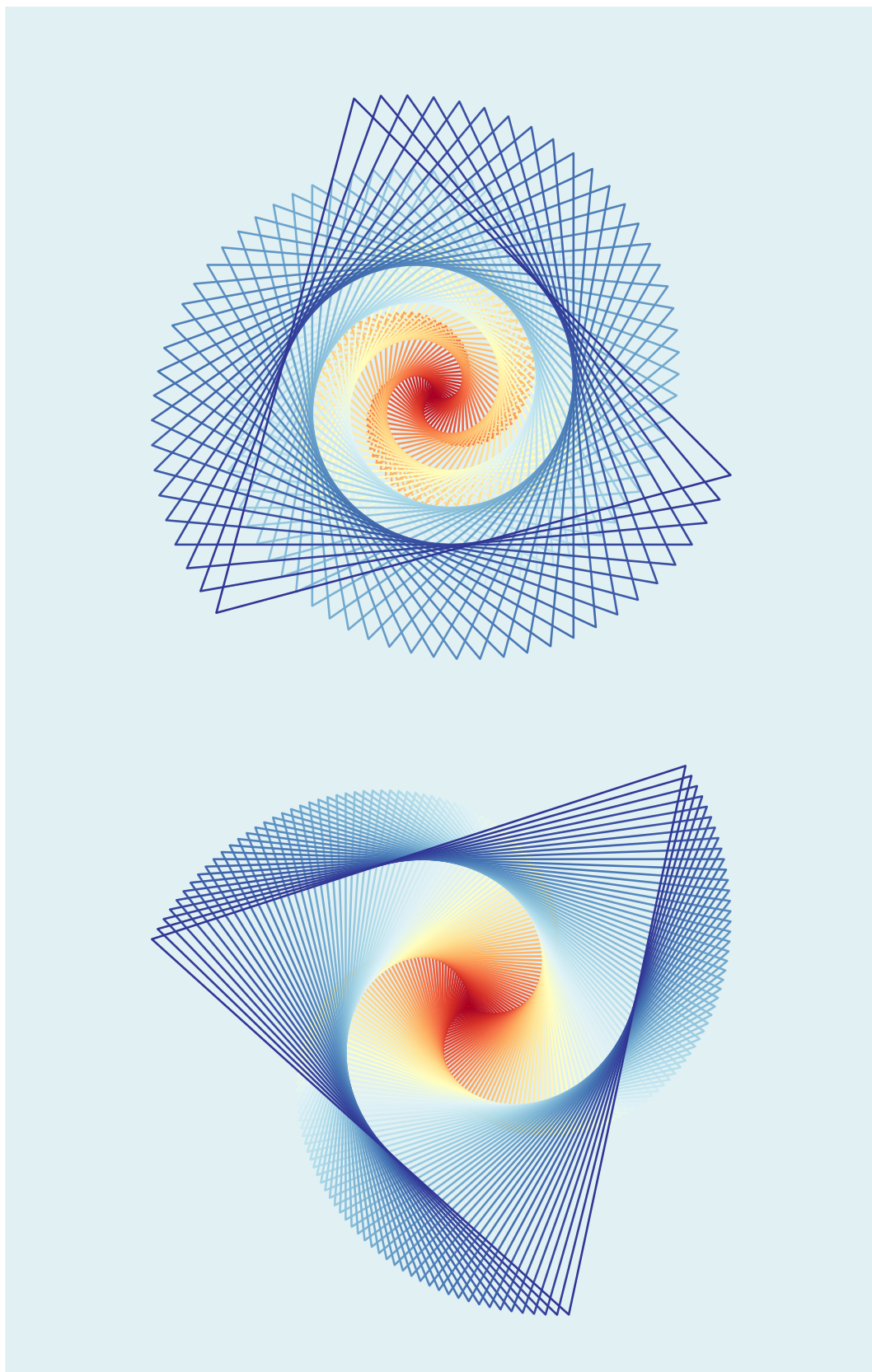


图 15. 两组旋转三角形 |  BK_2_Ch08_11.ipynb

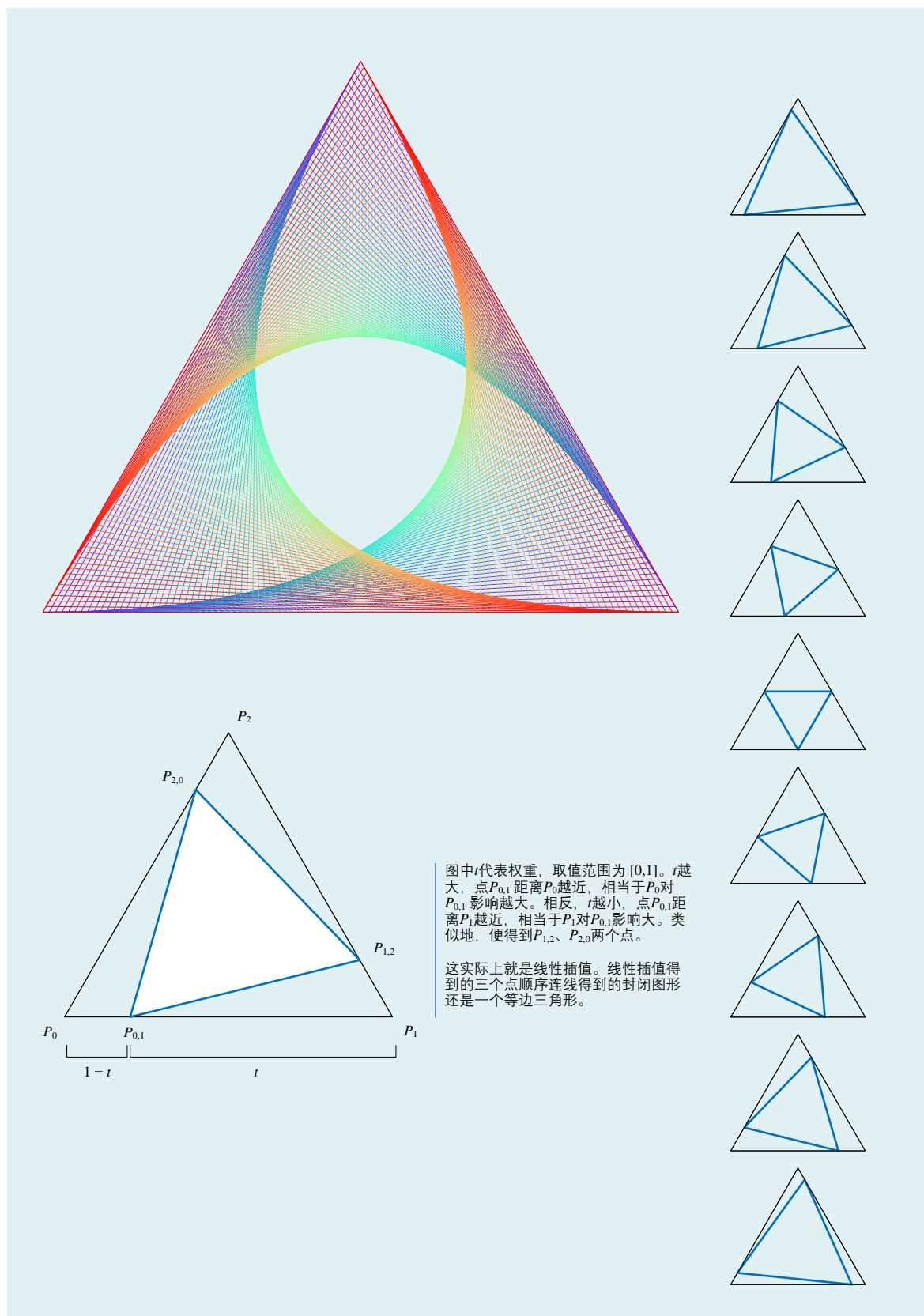


图 16. 线性插值，等边三角形 | BK_2_Ch08_12.ipynb

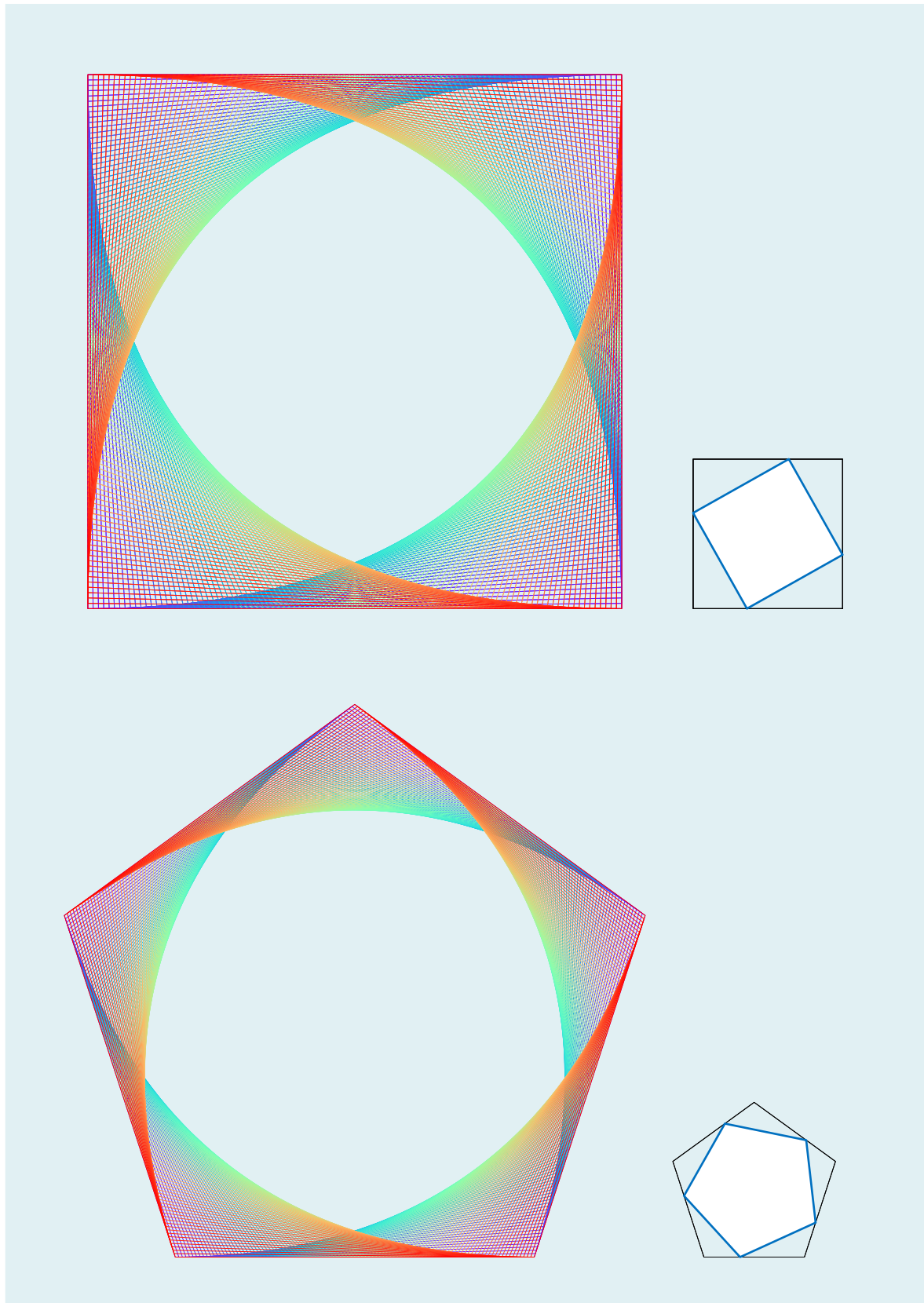



图 17. 线性插值, 正方形、正五边形 |  BK_2_Ch08_12.ipynb

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