15 Monte Carlo Simulation
蒙特卡洛模拟

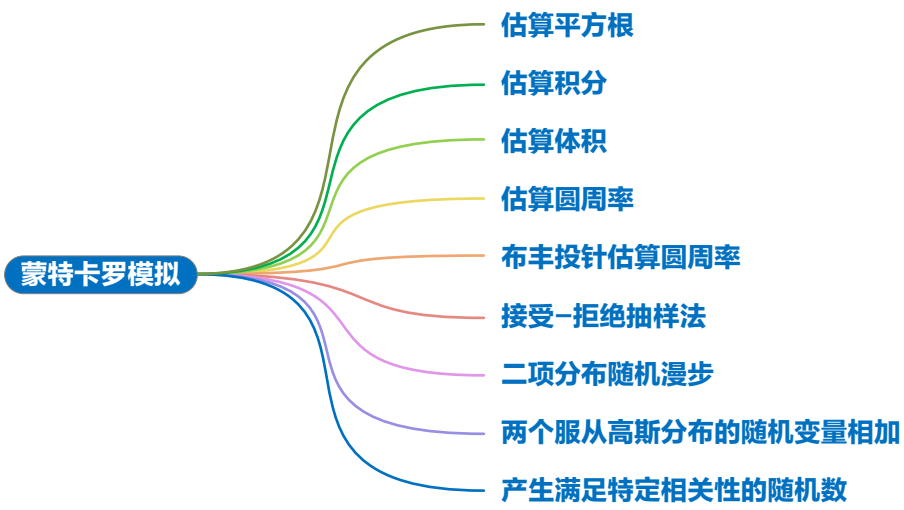
以概率统计为基础，基于伪随机数，进行数值模拟

任何考虑用算术手段来产生随机数的人当然都是有原罪的。

Anyone who considers arithmetical methods of producing random digits is, of course, in a state of sin.

—— 约翰·冯·诺伊曼 (John von Neumann) | 美国籍数学家 | 1903 ~ 1957

- `matplotlib.patches.Circle()` 绘制正圆
- `matplotlib.pyplot.semilogx()` 横轴设置为对数坐标
- `numpy.empty()` 产生全为 NaN 序列
- `numpy.random.beta()` 产生服从 Beta 分布的随机数
- `numpy.random.binomial()` 产生服从二项分布的随机数
- `numpy.random.dirichlet()` 产生服从 Dirichlet 分布的随机数
- `numpy.random.exponential()` 产生服从指数分布的随机数
- `numpy.random.geometric()` 产生服从几何分布的随机数
- `numpy.random.lognormal()` 产生服从对数正态分布的随机数
- `numpy.random.multivariate_normal()` 产生服从多项正态分布的随机数
- `numpy.random.normal()` 产生服从正态分布的随机数
- `numpy.random.poisson()` 产生服从泊松分布的随机数
- `numpy.random.randint()` 产生均匀整数随机数
- `numpy.random.standard_t()` 产生服从学生 t-分布的随机数
- `numpy.random.uniform()` 产生服从连续均匀分布的随机数
- `numpy.where()` 返回满足条件的元素序号
- `scipy.integrate.dblquad()` 求解双重定积分值
- `scipy.integrate.quad()` 求解定积分值
- `scipy.linalg.cholesky()` 对矩阵进行 Cholesky 分解
- `seaborn.distplot()` 绘制频率直方图和 KDE 曲线
- `seaborn.heatmap()` 绘制热图



15.1 蒙特卡洛模拟：基于伪随机数发生器

蒙特卡洛模拟 (Monte Carlo simulation)，也称统计模拟方法，是以概率统计理论为核心的数值计算方法。蒙特卡洛模拟将提供多种可能的结果以及通过大量随机数据样本得出的每种结果的概率。冯·诺伊曼 (John von Neumann) 等三名科学家在 20 世纪 40 年代发明了蒙特卡洛模拟。他们以摩纳哥著名的赌城——蒙特卡洛 (Monte Carlo)——为其命名。

表 1 所示为 NumPy 中和随机数有关的常见函数。

本章介绍几个最基本的蒙特卡洛模拟试验。

表 1. NumPy 中和随机数有关的常见函数

函数名称	函数介绍
numpy.random.beta()	生成指定形状参数的贝塔分布随机数
numpy.random.binomial()	返回给定形状的随机二项分布数组。
numpy.random.chisquare()	生成指定自由度的卡方分布随机数
numpy.random.choice()	随机从给定的数组中选择元素。
numpy.random.dirichlet()	生成指定参数的狄利克雷分布随机数
numpy.random.exponential()	生成指定尺度的指数分布随机数
numpy.random.gamma()	生成指定形状和尺度的伽马分布随机数
numpy.random.lognormal()	生成指定均值和标准差的对数正态分布随机数
numpy.random.multivariate_normal()	生成多元正态分布随机数
numpy.random.normal()	生成指定均值和标准差的正态分布随机数
numpy.random.poisson()	生成指定均值的泊松分布随机数
numpy.random.power()	返回给定形状的随机幂律分布数组。
numpy.random.rand()	返回一个给定形状的随机浮点数数组，值在 0 到 1 之间。
numpy.random.randint()	返回一个给定形状的随机整数数组，值在给定范围之间。
numpy.random.randn()	返回一个给定形状的随机浮点数数组，值遵循标准正态分布。
numpy.random.random()	生成[0, 1)之间的随机数
numpy.random.seed()	设置随机数生成器的种子，确保随机数生成的可重复性。
numpy.random.shuffle()	随机打乱给定的数组。
numpy.random.uniform()	生成指定范围内的均匀分布随机数

15.2 估算平方根

本节用蒙特卡洛模拟估算 $\sqrt{2}$。如图 1 所示，为了估算 $\sqrt{2}$，可以在 0 ~ 2 的范围内产生大量服从均匀分布的随机数。在 0 ~ 2 范围内，随机数在 0 ~ $\sqrt{2}$ 出现的概率为 $\sqrt{2}/2$，$\sqrt{2}$ 则可以根据下式估计得到：

$$\sqrt{2} \approx 2 \times \frac{n\left(0 \le x \le \sqrt{2}\right)}{n\left(0 \le x \le 2\right)} \tag{1}$$

其中 $n()$计算频数。

由于 $\sqrt{2}$ 未知，所以采用图 1 所示平方技巧，即 $\sqrt{2}$ 可以根据下式得到：

$$\sqrt{2} \approx 2 \times \frac{n\left(0 \le x^2 \le 2\right)}{n\left(0 \le x^2 \le 4\right)} \tag{2}$$

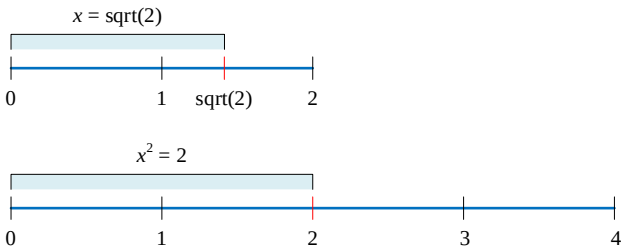


图 1. 估算 sqrt(2)

代码文件 Bk5_Ch15_01.py 估算 sqrt(2)。

15.3 估算积分

本节给出的例子用蒙特卡罗模拟方法估算积分。

给出如下函数 $f(x)$：

$$f\left(x\right) = \frac{x \cdot \sin\left(x\right)}{2} + 8 \tag{3}$$

计算 $f(x)$ 在 [2, 10] 区间内定积分：

$$\int_2^{10} \left(\frac{x \cdot \sin\left(x\right)}{2} + 8 \right) \mathrm{d}x \tag{4}$$

如图 2 所示，在 [2, 10] 区间中，函数 $f(x)$ 的最大值为 12。在横轴取值从 2 ~ 10，纵轴取值从 0 ~ 12 的长方形空间里，产生满足均匀分布的 1000 个数据点。图 2 中蓝色 ● 在曲线之下，红色 × 在曲线之上。图 2 中整个长方形的面积为 96，定积分对应曲线之下的面积 A，可以通过下式估算得到：

$$A \approx 96 \times \frac{n\left(\text{below } f\left(x\right)\right)}{1000} \tag{5}$$

$n(\text{below } f(x))$ 为 1000 个数据点中位于 $f(x)$ 曲线之下的数量。

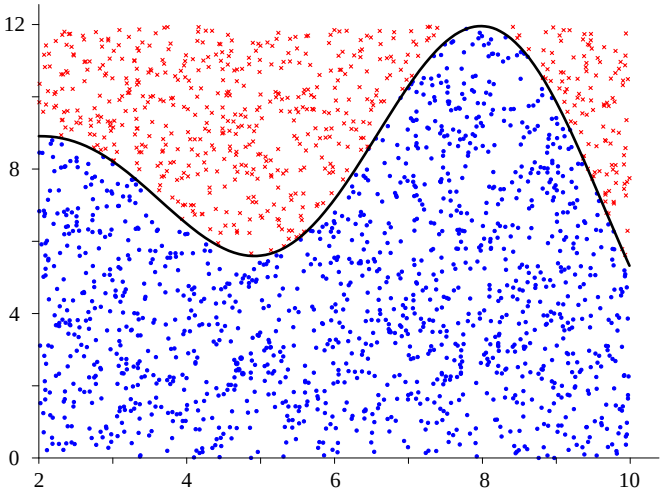


图 2. 用蒙特卡罗模拟法求积分

代码文件 Bk5_Ch15_02.py 估算积分。

15.4 估算体积

本节用蒙特卡洛模拟估算空间体积大小。图 3 (a) 所示二次曲面解析式如下：

$$z = 2 - x^2 - y^2 \tag{6}$$

当 x 和 y 均在 [−1, 1] 范围内时，编写代码用蒙特卡洛模拟估算图 3 (a) 曲面和 $z = 0$ 平面 (蓝色) 构造的空间体积。这个体积相当于如下双重定积分：

$$\int_{-1}^{1}\int_{-1}^{1}\left(2 - x^2 - y^2\right)\mathrm{d}x\,\mathrm{d}y \tag{7}$$

整个立方体空间体积为 8，在这个空间均匀产生 5000 个随机点。如图 3 (b) 所示，二次曲面之上随机点为红色，曲面之下随机点为蓝色。类似上一节，根据随机点的比例，可以估算 (7) 定积分。

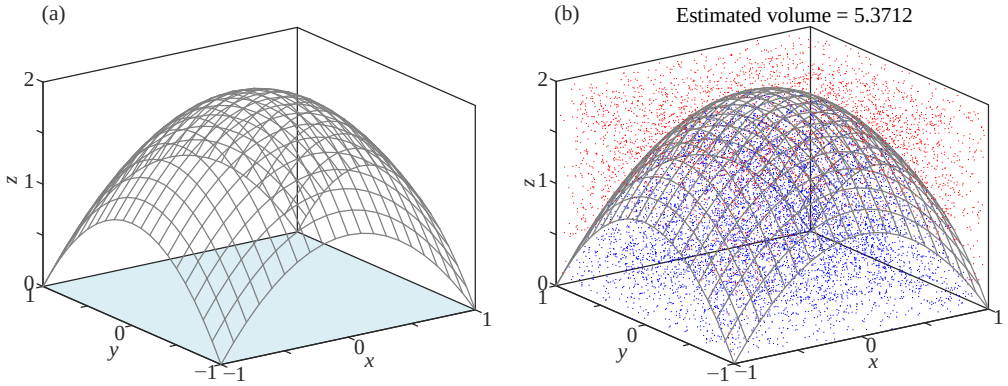


图 3. 利用蒙特卡洛估算体积

Bk5_Ch15_03.py 估算本节体积。

15.5 估算圆周率

《数学要素》一本已经介绍几种方法估算圆周率 π，本节介绍采用蒙特卡罗模拟法估算圆周率。

圆面积和正方形面积之间的比例关系为：

$$\frac{A_{\text{circle}}}{A_{\text{square}}} = \frac{\pi}{4} \tag{8}$$

可以推导得到：

$$\pi = 4 \times \frac{A_{\text{circle}}}{A_{\text{square}}} \tag{9}$$

图 4 所示为一次随机数数量为 500 条件下，圆周率估算结果。图 5 所示为不断增大随机数数量，圆周率估算精确度不断提高。

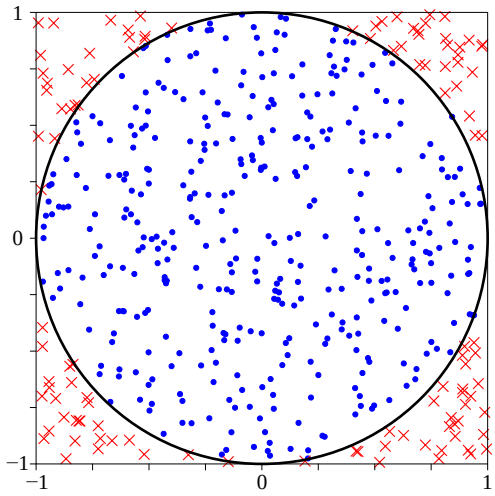


图 4. 蒙特卡罗模拟估算圆周率

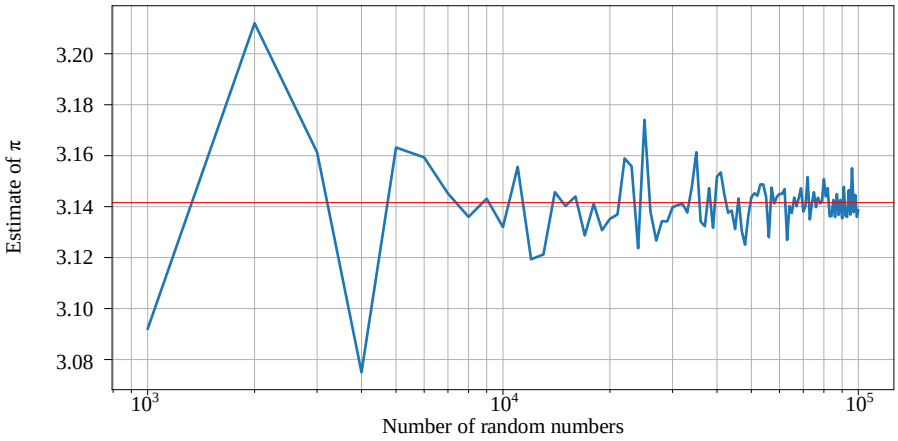


图 5. 不断增大随机数数量，圆周率估算精确度不断提高

Bk5_Ch15_04.py 利用蒙特卡洛模拟估算圆周率。

15.6 布丰投针估算圆周率

布丰投针 (Buffon's needle problem) 也可以用来估算圆周率。

十八世纪，法国博物学家布丰 (Comte de Buffon) 提出著名的布丰投针问题。一个用平行且等距木纹铺成的地板，随意投掷一支长度比木纹间距略小的针，求针和其中一条木纹相交的概率。

如图 6 所示，和平行线相交的针颜色为红色，不和平行线相交的针颜色为蓝色。设平行线距离为 t，针的长度为 l。本节布丰投针问题，我们仅仅考虑“短针”情况，即 $l < t$。

如放大视图所示，x 为针的中心和最近平行线的距离，θ 为针和平行线之间的锐角夹角。

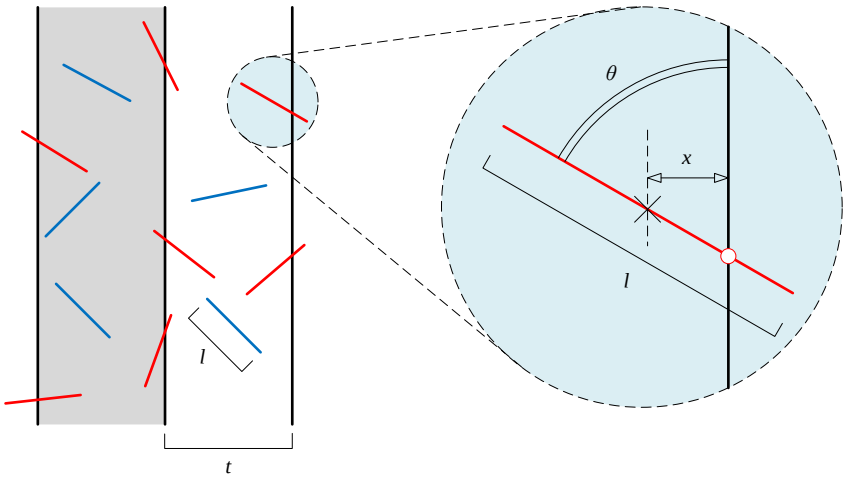


图 6. 布丰投针原理

不难理解，X 作为一个随机变量是在 $[0, t/2]$ 区间的连续均匀分布，概率密度函数为：

$$f_X(x) = \frac{2}{t} \quad x \in [0, t/2] \tag{10}$$

同理，Θ 作为一个随机变量是在 $[0, \pi/2]$ 区间的均匀分布，概率密度函数为：

$$f_\Theta(\theta) = \frac{2}{\pi} \quad \theta \in [0, \pi/2] \tag{11}$$

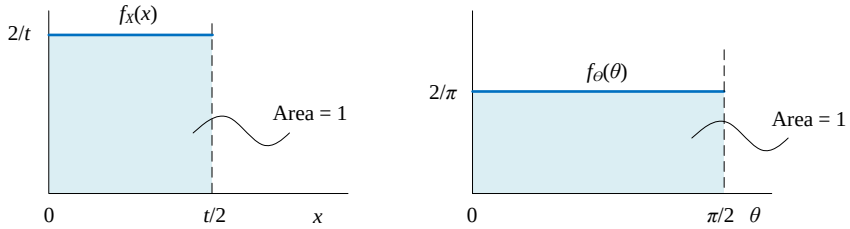


图 7. X 和 Θ 的概率密度函数

显然，X 和 Θ 这两个随机变量相互独立；因此，它们的联合概率密度函数是两者之积，即：

$$f_{\Theta,X}(\theta, x) = \frac{2}{\pi}\frac{2}{t} = \frac{4}{\pi t} \quad \theta \in [0, \pi/2], \quad x \in [0, t/2] \tag{12}$$

给定夹角 θ，满足如下条件，针和平行线相交：

$$x \le \frac{l}{2}\sin\theta, \quad \theta \in [0, \pi/2], \quad x \in [0, t/2] \tag{13}$$

因此，针线相交的概率为如下双重定积分：

$$\Pr(\text{cross}) = \int_0^{\frac{\pi}{2}} \int_0^{\frac{l}{2}\sin\theta} \frac{4}{\pi t} \mathrm{d}x \mathrm{d}\theta = \frac{2l}{\pi t} \tag{14}$$

假设抛 n 根针，其中有 c 根和平行线相交，概率值 Pr(cross) 可以通过下式估算：

$$\Pr(\text{cross}) \approx \frac{c}{n} \tag{15}$$

联立 (14) 和 (15)，可以得到：

$$\frac{2l}{\pi t} \approx \frac{c}{n} \tag{16}$$

从而推导得到，圆周率的估算值：

$$\pi \approx \frac{2l}{t}\frac{n}{c} \tag{17}$$

图 8 所示为某次试验投掷 2000 根针，612 根和平行线相交 (红色线)；式样中，针的长度 l = 1，平行线间隔 t = 2。

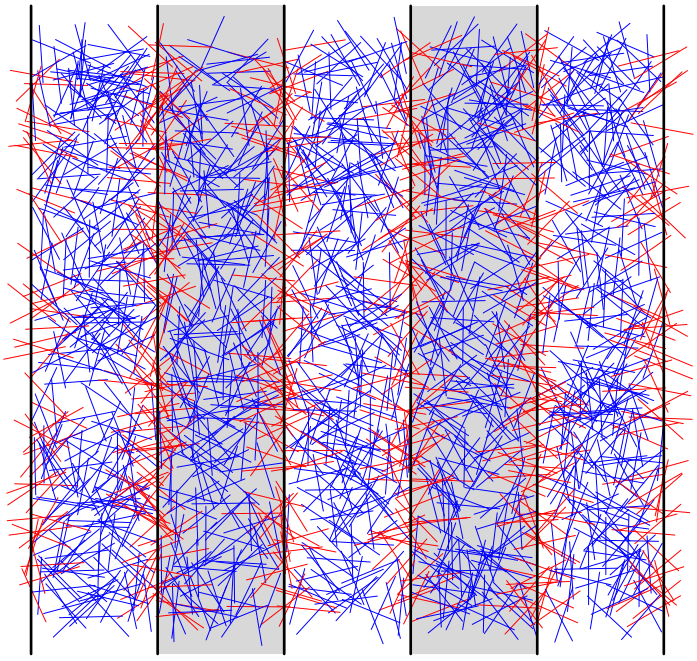


图 8. 投掷 2000 根针，612 根和平行线相交，针的长度 l = 1，平行线间隔 t = 2

实际上，根据 (13)，我们知道针和平行线相交的概率 Pr(cross) 可以进一步简化。在图 9 阴影区域产生满足均匀分布的随机数，随机数落入蓝色区域的概率就是 Pr(cross)。这样，我们可以根据这一思路编程解决这个简化版的布丰投针估算圆周率问题。

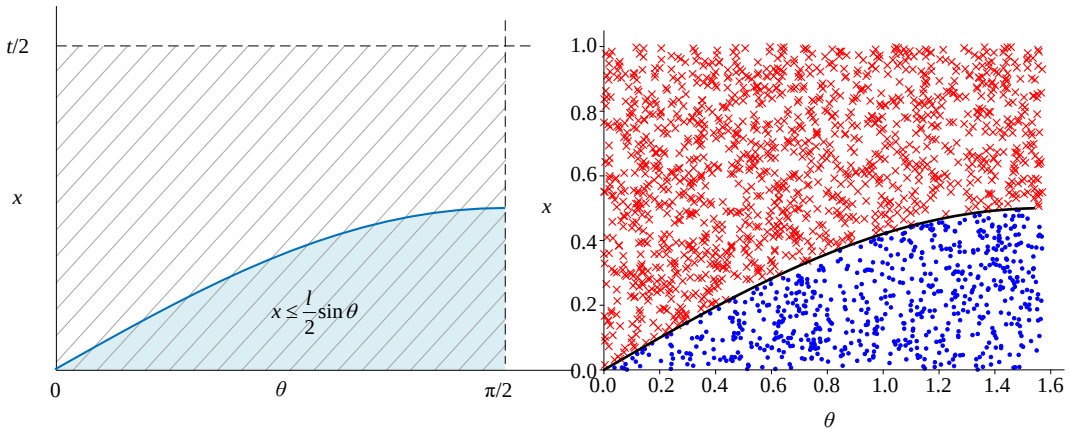


图 9. 针和平行线相交的概率，蒙特卡洛模拟试验结果

Bk5_Ch15_05.py 完成简化版布丰投针蒙特卡洛模拟试验。

15.7 接受–拒绝抽样法

随机变量 X 的概率密度函数为 $f_X(x)$，但是 $f_X(x)$ 不可以直接抽样。也就是说，不能直接产生满足 $f_X(x)$ 的随机数。我们可以采用本节介绍的接受-拒绝抽样法 (accept-reject sampling method)。接受-拒绝抽样法适合于概率密度函数复杂、不能直接抽样的情况。

接受-拒绝抽样法的基本思想是，生成一个辅助分布 (proposal distribution)，并利用这个分布来生成随机数。然后，计算目标概率分布在该点处的概率密度，并将其除以辅助分布在该点处的概率密度，得到接受率 (acceptance ratio)。随机生成一个介于 0 和 1 之间的均匀分布的随机数，如果这个随机数小于接受率，则接受这个样本，否则拒绝。重复此过程，直到生成足够多的样本。

接受-拒绝抽样法的优点是简单易用、适用范围广，可以应用于各种不同的概率分布。它的缺点是样本生成的效率可能较低，因为需要进行接受和拒绝的判断。在实践中，辅助分布的选择对于样本生成的效率和精度非常重要。

给定如图 10 所示的随机变量 X 的概率密度函数为 $f_X(x)$。显然没有“现成”的随机数发生器能够直接生成满足 $f_X(x)$ 的随机数。

我们首先生成如图 11 所示的连续均匀分布随机数。简单来说，如图 12 所示，接受-拒绝抽样法就是在图 11 中“剪裁”得到形似图 10 的部分，并“接受”这些随机数。图 13 所示为用直方图可视化“拒

绝”和“接受”部分的随机数。大家很容易发现，图中浅蓝色部分矩形构成形状形似图 10 中的 $f_X(x)$。我们将在本书第 22 章用到接受-拒绝抽样法。

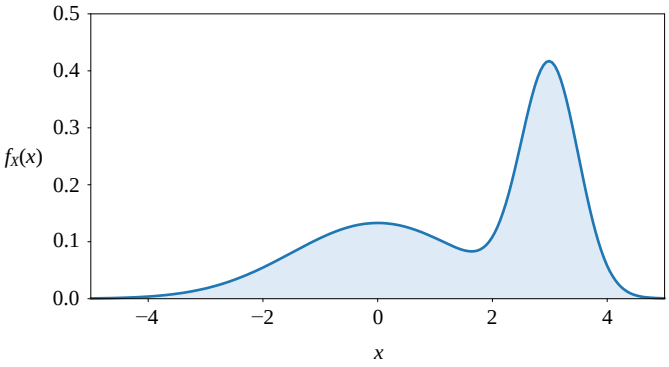


图 10. 随机变量 X 的概率密度函数

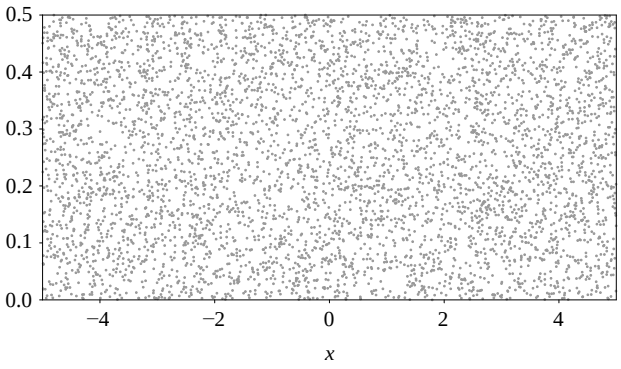


图 11. 生成连续均匀分布随机数

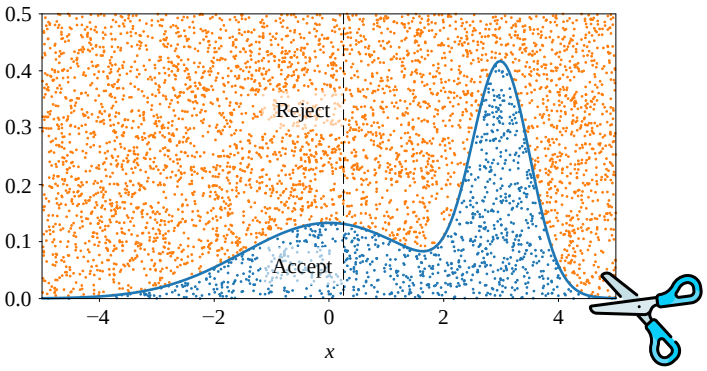


图 12. “剪裁”连续均匀分布随机数

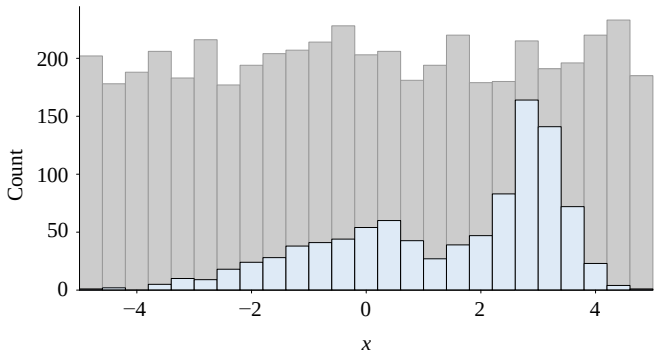


图 13. “接受” vs “拒绝” 部分随机数

15.8 二项分布随机漫步

“鸢尾花书”《数学要素》第 20 章讲过在二叉树规定的网格行走的例子。

如图 14 所示，登山者在二叉树始点或中间节点时，他都会面临“向上”或“向下”抉择。如果登山者，通过抛硬币来决定每一步的行走路径——正面，向右上走；反面，向右下走。

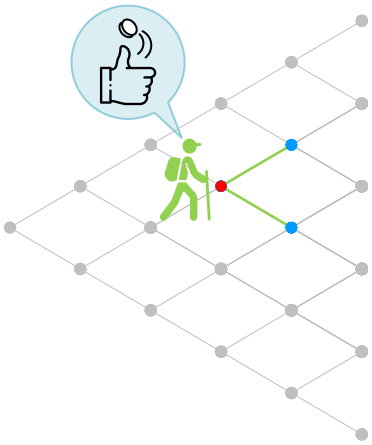


图 14. 二叉树路径与可能性，图片来自《数学要素》

图 15 所示为若干条二叉树随机行走路径，模拟时向上行走的概率 $p = 0.5$。乍一看图 15 很难发现任何规律。但是不断增大随机行走的路径数 n，如图 16 所示，我们发现登山者到达终点的位置呈现类似二项分布规律。观察图 16 (c)，我们发现当 $p = 0.5$ 时，登山者大概率会到达二叉树网格终点中部。

图 17 和图 18 对应登山者向上行走的概率 $p = 0.6$。图 19 图 20 对应登山者向上行走的概率 $p = 0.4$。请大家自行分析这四幅图。

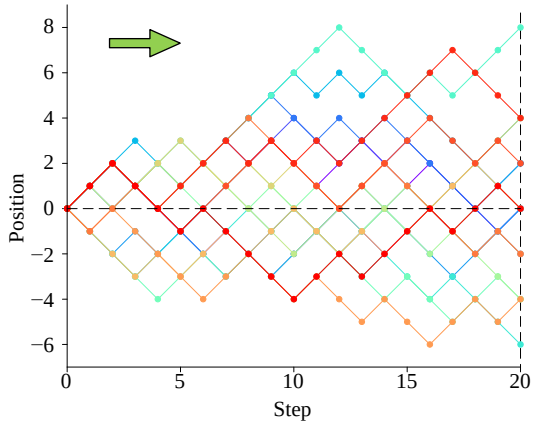


图 15. 二叉树随机行走路径，向上行走的概率 $p = 0.5$

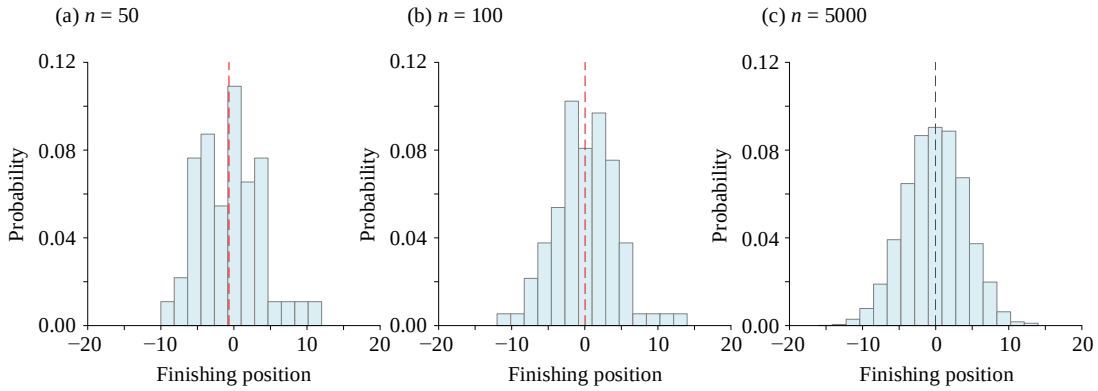


图 16. 第 20 步时随机漫步位置分布，$p = 0.5$

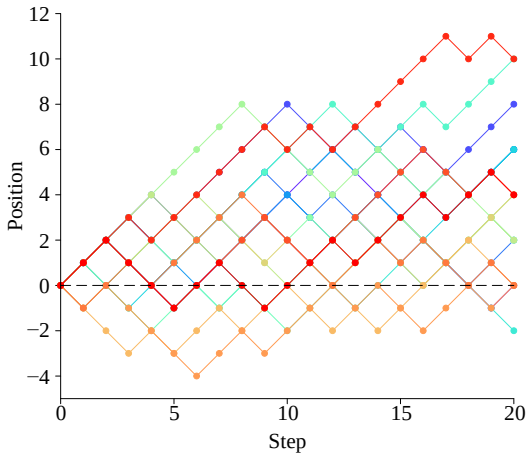


图 17. 二叉树随机行走路径，向上行走的概率 $p = 0.6$

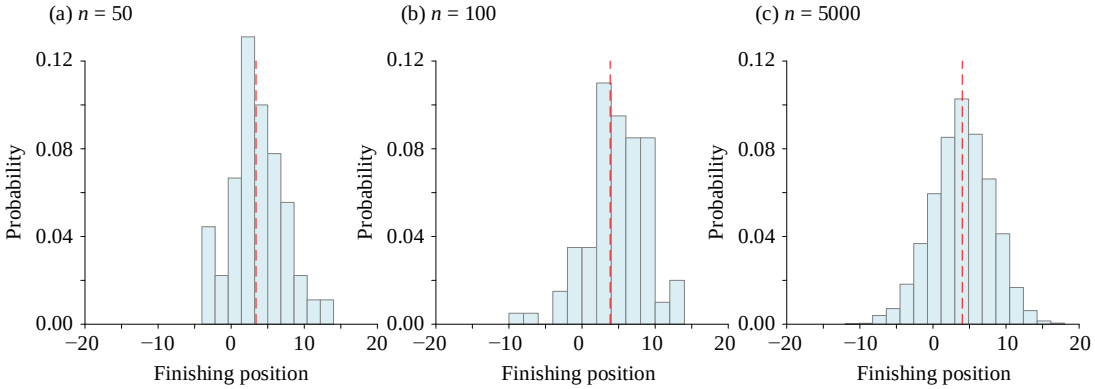


图 18. 第 20 步时随机漫步位置分布，$p = 0.6$

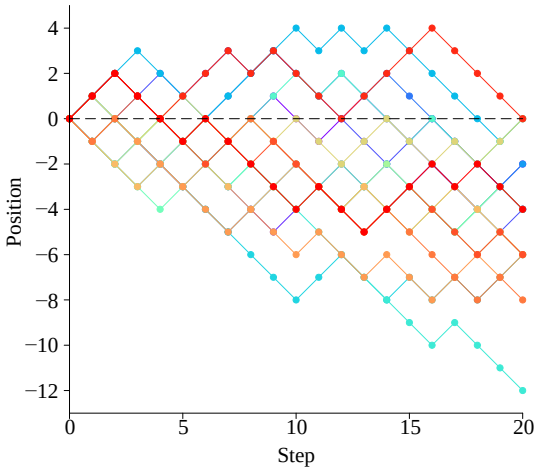


图 19. 二叉树随机行走路径，向上行走的概率 $p = 0.4$

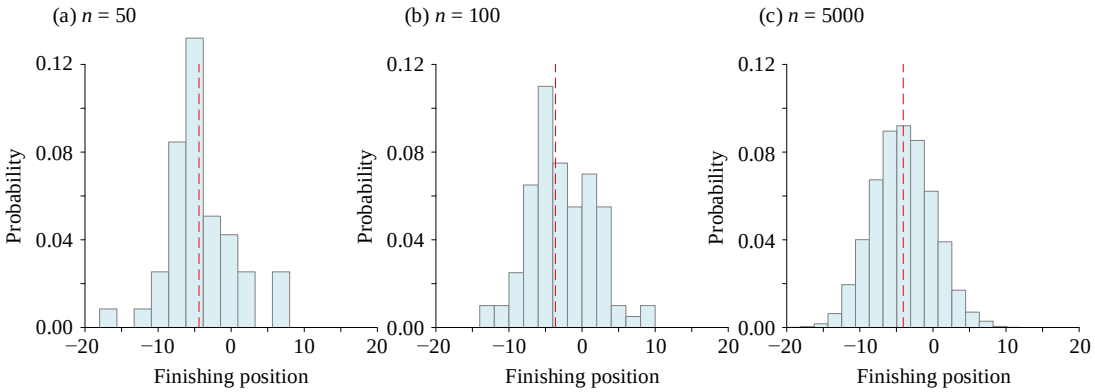


图 20. 向上行走的概率 $p = 0.4$

Bk5_Ch15_06.py 完成本节二叉树随机漫步试验。

15.9 两个服从高斯分布的随机变量相加

X_1 和 X_2 分别服从正态分布，具体如下：

$$\begin{cases} X_1 \sim N\left(\mu_1, \sigma_1^2\right) \\ X_2 \sim N\left(\mu_2, \sigma_2^2\right) \end{cases} \tag{18}$$

图 21 所示为 X_1 和 X_2 的随机数分布情况。

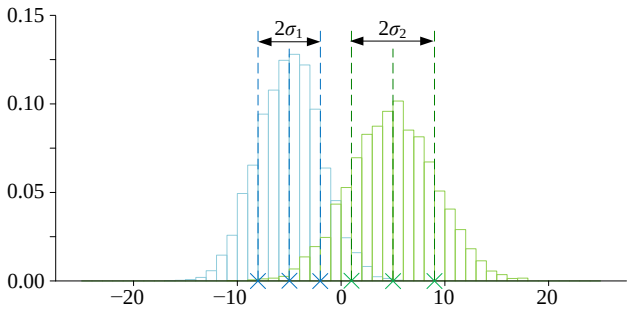


图 21. X_1 和 X_2 的随机数分布情况

X_1 和 X_2 分布之和，即 $Y = X_1 + X_2$，也服从正态分布：

$$Y \sim N\left(\mu_Y, \sigma_Y^2\right) \tag{19}$$

其中，

$$\begin{aligned} \mu_Y &= \mu_1 + \mu_2 \\ \sigma_Y^2 &= \sigma_1^2 + \sigma_2^2 + \rho_{1,2}\sigma_1\sigma_2 \end{aligned} \tag{20}$$

图 22 所示为相关性系数 $\rho_{1,2}$ 影响 $X_1 + X_2$ 随机数分布。请大家利用几何视角分析图中不同子图结果。

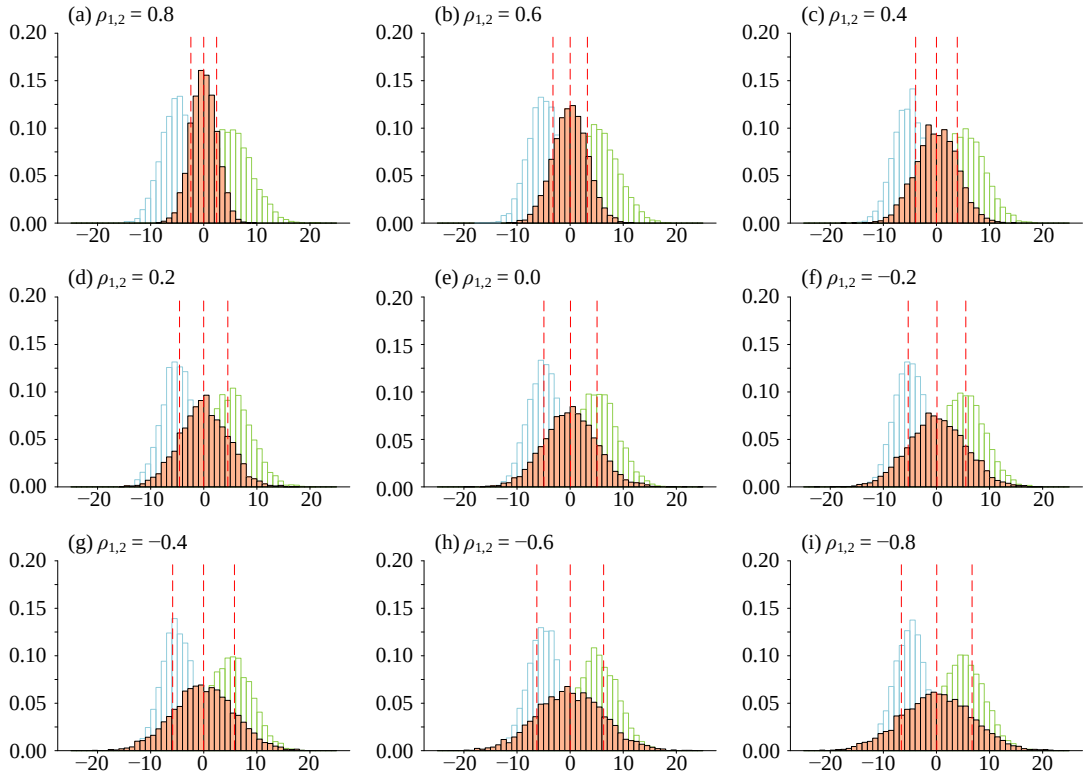


图 22. 相关性系数如何影响 $X_1 + X_2$ 随机数分布

Bk5_Ch15_07.py 绘制图 22。

15.10 产生满足特定相关性的随机数

Cholesky 分解

如图 23 所示，我们在《矩阵力量》第 14 章中学过如何完成“单位圆 (缩放) → 正椭圆 (剪切) → 旋转椭圆”几何变换。经过本书上一板块的学习，大家已经清楚单位圆代表 $N(\boldsymbol{0}, \boldsymbol{I})$，而旋转椭圆代表 $N(\boldsymbol{0}, \boldsymbol{\Sigma})$。再经过平移，我们就可以到 $N(\boldsymbol{\mu}, \boldsymbol{\Sigma})$。

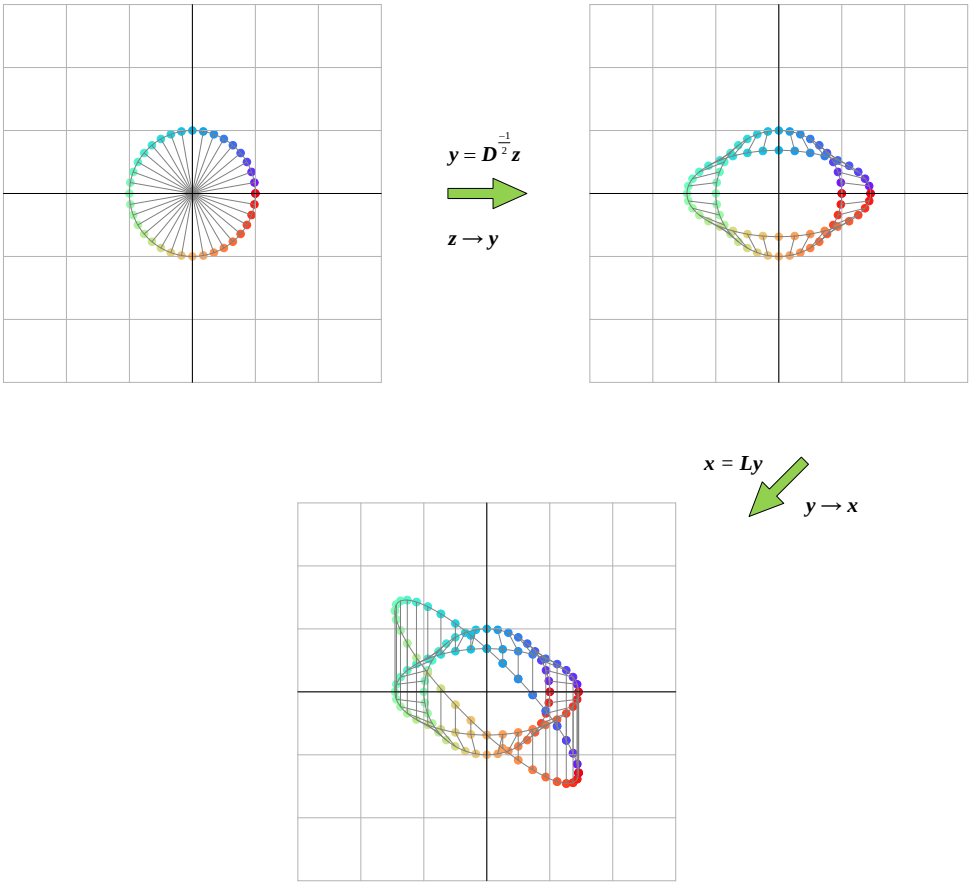


图 23. 单位圆 (缩放) → 正椭圆 (剪切) → 旋转椭圆，来自《矩阵力量》第 14 章

图 23 用到的数学工具是 LDL 分解。实际上，利用 Cholesky 分解我们可以通过一次矩阵乘法便完成“缩放 + 剪切”。这便是利用 Cholesky 分解结果产生满足特定相关性随机数的技术路线。

如图 24 所示，首先生成满足 $N(\boldsymbol{0}, \boldsymbol{I}_{D\times D})$ 的随机数矩阵 $\boldsymbol{Z}$。

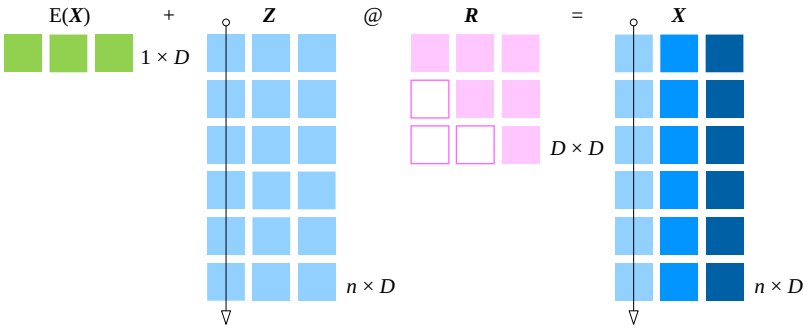


图 24. 产生满足特定相关性随机数的矩阵运算

然后，对协方差矩阵 $\boldsymbol{\Sigma}$ 进行 Cholesky 分解，得到下三角矩阵 $\boldsymbol{R}^{\mathrm{T}}$ 和上三角矩阵 $\boldsymbol{R}$ 乘积：

$$\boldsymbol{\Sigma} = \boldsymbol{R}^{\mathrm{T}}\boldsymbol{R} \tag{21}$$

注意，要求 $\boldsymbol{\Sigma}$ 为正定；否则，不能进行 Cholesky 分解。

矩阵 $\boldsymbol{R}$ 中含有图 23 中“剪切”、“缩放”两个成分。

$\boldsymbol{Z}$ 服从 $N(\boldsymbol{0}, \boldsymbol{I}_{D\times D})$，经过如下运算得到的多元随机数 $\boldsymbol{X}$ 服从 $N(\mathrm{E}(\boldsymbol{X}), \boldsymbol{\Sigma}_{D\times D})$：

$$\underset{N(\mathrm{E}(\boldsymbol{X}),\boldsymbol{\Sigma})}{\boldsymbol{X}} = \underset{N(\boldsymbol{0},\boldsymbol{I})}{\boldsymbol{Z}}\ \underbrace{\boldsymbol{R}}_{\text{Scale + shear}} + \underbrace{\mathrm{E}(\boldsymbol{X})}_{\text{Translate}} \tag{22}$$

其中

$$\boldsymbol{X} = \begin{bmatrix} x_1 & x_2 & \cdots & x_D \end{bmatrix},\quad \boldsymbol{Z} = \begin{bmatrix} z_1 & z_2 & \cdots & z_D \end{bmatrix},\quad \mathrm{E}(\boldsymbol{X}) = \begin{bmatrix} \mu_1 & \mu_2 & \cdots & \mu_D \end{bmatrix} \tag{23}$$

分别计算 $\boldsymbol{Z}$ 和 $\boldsymbol{X}$ 的协方差矩阵。$\boldsymbol{Z}$ 的协方差矩阵：

$$\boldsymbol{\Sigma}_{\boldsymbol{Z}} = \frac{\boldsymbol{Z}^{\mathrm{T}}\boldsymbol{Z}}{n-1} = \frac{1}{n-1}\begin{bmatrix} \boldsymbol{z}_1^{\mathrm{T}} \\ \boldsymbol{z}_2^{\mathrm{T}} \\ \vdots \\ \boldsymbol{z}_D^{\mathrm{T}} \end{bmatrix}\begin{bmatrix} \boldsymbol{z}_1 & \boldsymbol{z}_2 & \cdots & \boldsymbol{z}_D \end{bmatrix} = \frac{1}{n-1}\begin{bmatrix} \boldsymbol{z}_1^{\mathrm{T}}\boldsymbol{z}_1 & \boldsymbol{z}_1^{\mathrm{T}}\boldsymbol{z}_2 & \cdots & \boldsymbol{z}_1^{\mathrm{T}}\boldsymbol{z}_D \\ \boldsymbol{z}_2^{\mathrm{T}}\boldsymbol{z}_1 & \boldsymbol{z}_2^{\mathrm{T}}\boldsymbol{z}_2 & \cdots & \boldsymbol{z}_2^{\mathrm{T}}\boldsymbol{z}_D \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{z}_D^{\mathrm{T}}\boldsymbol{z}_1 & \boldsymbol{z}_D^{\mathrm{T}}\boldsymbol{z}_2 & \cdots & \boldsymbol{z}_D^{\mathrm{T}}\boldsymbol{z}_D \end{bmatrix} = \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{bmatrix}_{D\times D} \tag{24}$$

对 $\boldsymbol{X}$ 求协方差：

$$\begin{aligned} \boldsymbol{\Sigma}_{\boldsymbol{X}} &= \frac{\left(\boldsymbol{X} - \mathrm{E}(\boldsymbol{X})\right)^{\mathrm{T}}\left(\boldsymbol{X} - \mathrm{E}(\boldsymbol{X})\right)}{n-1} \\ &\underset{\boldsymbol{\Sigma}_{\boldsymbol{Z}}}{=} \frac{(\boldsymbol{Z}\boldsymbol{R})^{\mathrm{T}}\boldsymbol{Z}\boldsymbol{R}}{n-1} = \boldsymbol{R}^{\mathrm{T}}\frac{\boldsymbol{Z}^{\mathrm{T}}\boldsymbol{Z}}{n-1}\boldsymbol{R} = \boldsymbol{R}^{\mathrm{T}}\boldsymbol{R} = \boldsymbol{\Sigma} \end{aligned} \tag{25}$$

二维随机数

下面，我们先看 $D = 2$ 这个特殊情况。

二维随机变量 $\boldsymbol{\chi}$ 满足如下二维高斯分布：

$$\boldsymbol{\chi} = \begin{bmatrix} X_1 \\ X_2 \end{bmatrix} \sim N\left(\underbrace{\begin{bmatrix} \mu_1 \\ \mu_2 \end{bmatrix}}_{\boldsymbol{\mu}}, \underbrace{\begin{bmatrix} \sigma_1^2 & \rho\sigma_1\sigma_2 \\ \rho\sigma_1\sigma_2 & \sigma_2^2 \end{bmatrix}}_{\boldsymbol{\Sigma}}\right) \tag{26}$$

而 Z_1 和 Z_2 服从标准正态分布，且不相关。也就是说 (Z_1, Z_2) 服从 $N(\boldsymbol{0}, \boldsymbol{I}_{2\times 2})$：

$$\boldsymbol{\varsigma} = \begin{bmatrix} Z_1 \\ Z_2 \end{bmatrix} \sim N\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \underbrace{\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}}_{\boldsymbol{\Sigma}}\right) \tag{27}$$

对 (26) 协方差矩阵 Cholesky 分解：

$$\begin{bmatrix} \sigma_1^2 & \rho\sigma_1\sigma_2 \\ \rho\sigma_1\sigma_2 & \sigma_2^2 \end{bmatrix} = \underbrace{\begin{bmatrix} \sigma_1 & 0 \\ \rho\sigma_2 & \sigma_2\sqrt{1-\rho^2} \end{bmatrix}}_{L} \underbrace{\begin{bmatrix} \sigma_1 & \rho\sigma_2 \\ 0 & \sigma_2\sqrt{1-\rho^2} \end{bmatrix}}_{R} \tag{28}$$

也就是说，χ 的 ς 关系为：

$$\chi = L\varsigma + \mu \tag{29}$$

请大家思考为什么 (22) 采用上三角矩阵 R，而上式采用下三角矩阵 L。

D = 2 时，展开 (29) 得到：

$$\begin{bmatrix} X_1 \\ X_2 \end{bmatrix} = \begin{bmatrix} \sigma_1 & 0 \\ \rho\sigma_2 & \sigma_2\sqrt{1-\rho^2} \end{bmatrix} \begin{bmatrix} Z_1 \\ Z_2 \end{bmatrix} + \begin{bmatrix} \mu_1 \\ \mu_2 \end{bmatrix} \tag{30}$$

即

$$\begin{cases} X_1 = \sigma_1 Z_1 + \mu_1 \\ X_2 = \rho\sigma_2 Z_1 + \sigma_2\sqrt{1-\rho^2} Z_2 + \mu_2 \end{cases} \tag{31}$$

下面给出一个具体示例。

图 25 (a) 给出的二维随机数满足：

$$\begin{bmatrix} Z_1 \\ Z_2 \end{bmatrix} \sim N\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \right) \tag{32}$$

图 25 (b) 给出的二维随机数满足：

$$\begin{bmatrix} X_1 \\ X_2 \end{bmatrix} \sim N\left(\begin{bmatrix} 2 \\ 4 \end{bmatrix}, \begin{bmatrix} 4 & 2 \\ 2 & 2 \end{bmatrix} \right) \tag{33}$$

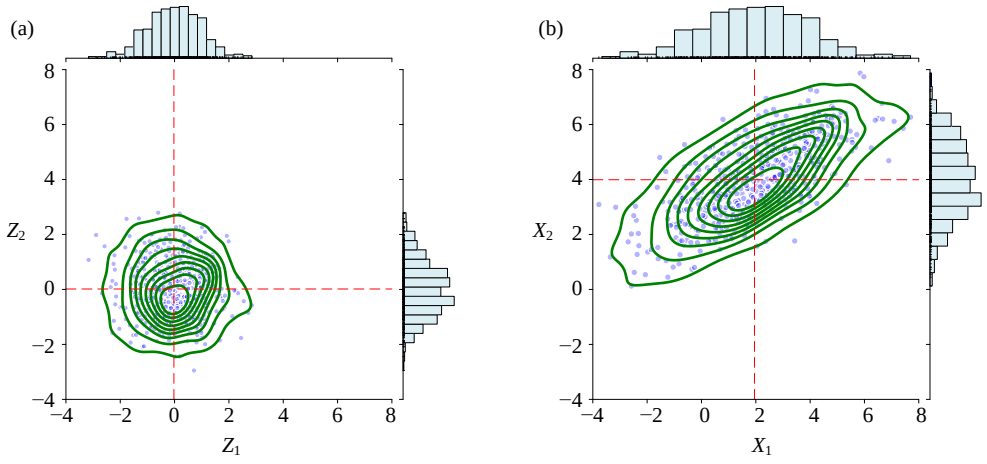
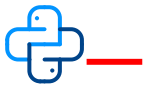


图 25. 将服从 IID 二维标准正态分布随机数转化为满足特定质心和协方差要求的随机数



Bk5_Ch15_08.py 生成图 25。代码中用到了 Cholesky 分解。

多维随机数

图 26 所示为采用多元高斯分布随机数发生器生成的随机数。这组随机数的均值、协方差矩阵和鸢尾花数据相同。请大家利用本节前文介绍的技术原理，首先生成满足 $N(\boldsymbol{0}, \boldsymbol{I}_{D\times D})$的随机数矩阵 $\boldsymbol{Z}$，然后再生成满足 $N(\mathrm{E}(\boldsymbol{X}), \boldsymbol{\Sigma}_{D\times D})$ 的随机数。

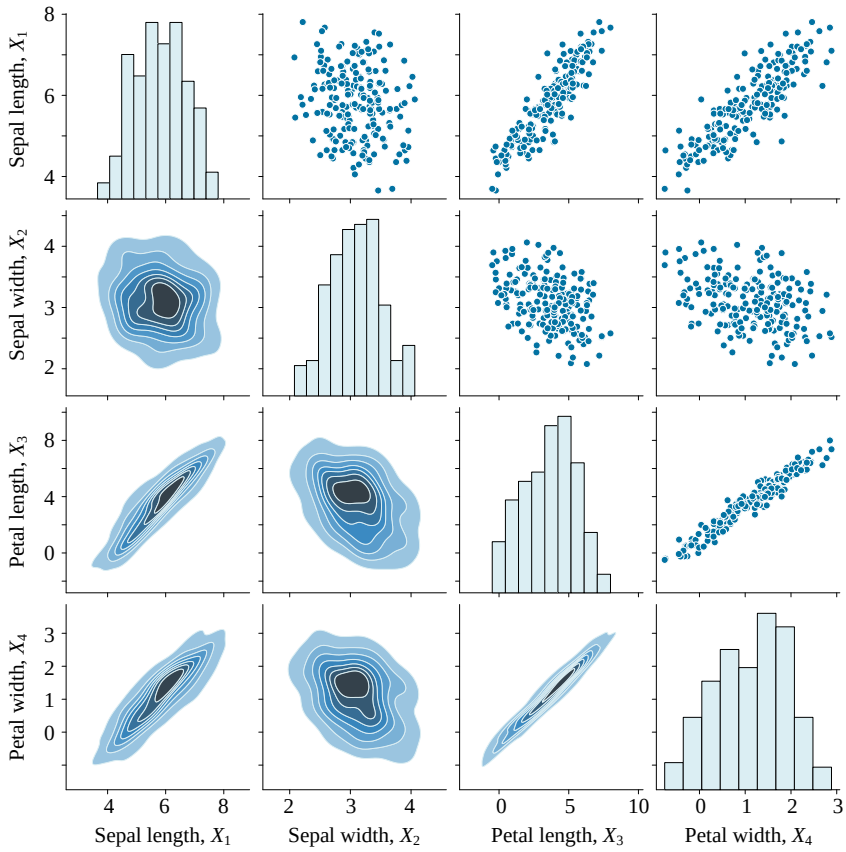
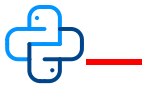


图 26. 四元高斯随机数



Bk5_Ch15_09.py 产生图 26 结果。

特征值分解

《矩阵力量》第 14 章还介讲过图 27 这幅图。图中，单位圆首先经过缩放得到正椭圆，然后正椭圆经过旋转得到旋转椭圆。这实际上是另外一条获得特定相关性随机数的技术路径。

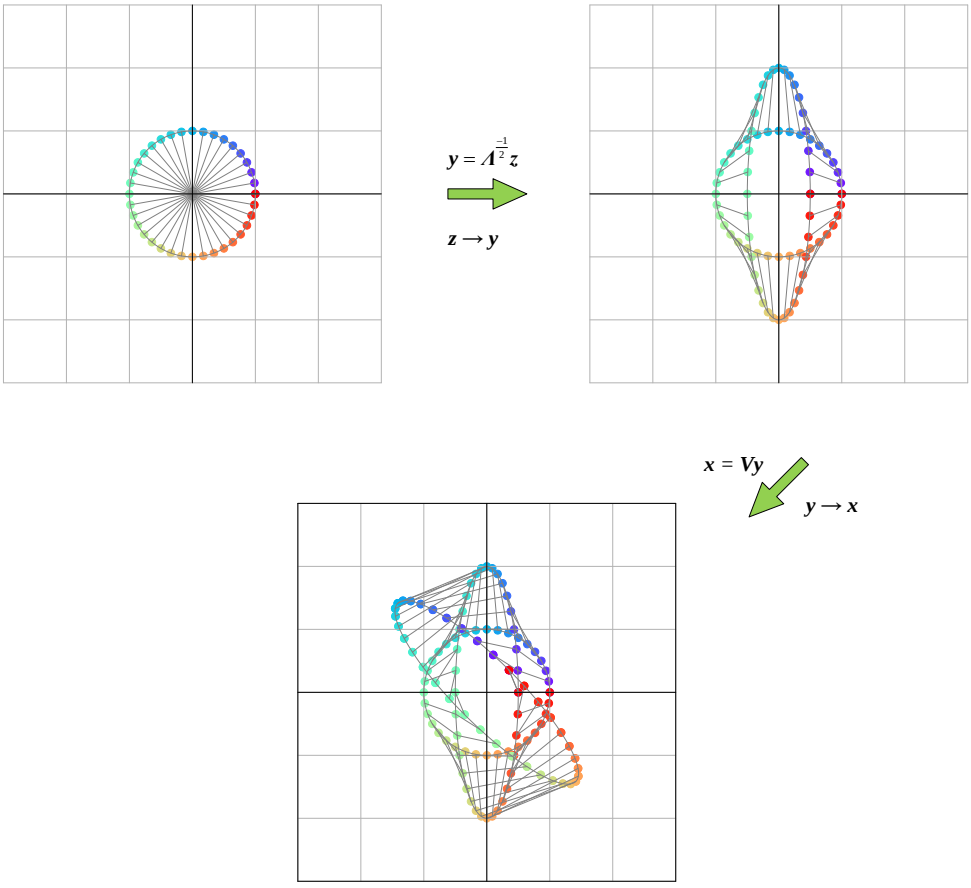


图 27. 单位圆 (缩放) → 正椭圆 (旋转) → 旋转椭圆

对协方差矩阵 $\boldsymbol{\Sigma}$ 进行特征值分解，然后写成“平方式”：

$$\boldsymbol{\Sigma} = \boldsymbol{V}\boldsymbol{\Lambda}\boldsymbol{V}^{\mathrm{T}} = \left(\boldsymbol{\Lambda}^{\frac{1}{2}}\boldsymbol{V}^{\mathrm{T}}\right)^{\mathrm{T}}\boldsymbol{\Lambda}^{\frac{1}{2}}\boldsymbol{V}^{\mathrm{T}} \tag{34}$$

如图 28 所示，随机数矩阵 $\boldsymbol{Z}$ 满足 $N(\boldsymbol{0}, \boldsymbol{I}_{D\times D})$，先经过 $\boldsymbol{\Lambda}^{\frac{1}{2}}$ 缩放，再经过 $\boldsymbol{V}^{\mathrm{T}}$ 旋转，最后通过 $\mathrm{E}(\boldsymbol{X})$ 平移获数据矩阵 $\boldsymbol{X}$：

$$\underset{N(\mathrm{E}(\boldsymbol{X}),\boldsymbol{\Sigma})}{\boldsymbol{X}} = \underset{N(\boldsymbol{0},\boldsymbol{I})}{\boldsymbol{Z}}\ \underset{\text{Scale}}{\boldsymbol{\Lambda}^{\frac{1}{2}}}\ \underset{\text{Rotate}}{\boldsymbol{V}^{\mathrm{T}}} + \underset{\text{Translate}}{\mathrm{E}(\boldsymbol{X})} \tag{35}$$

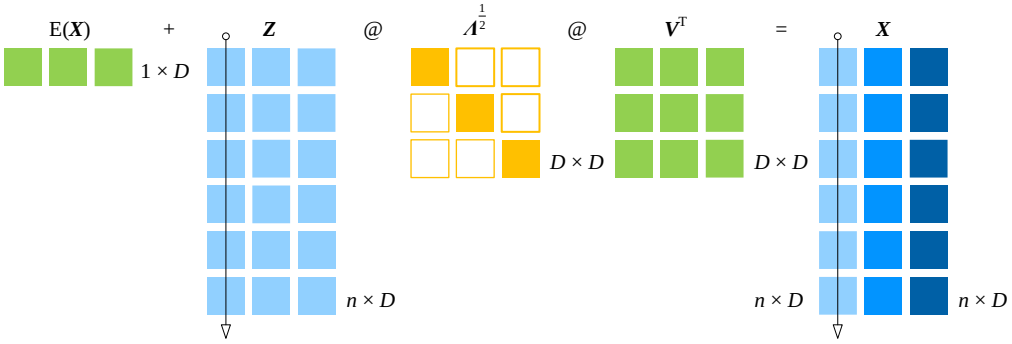


图 28. 产生满足特定相关性随机数的矩阵运算

对 $\boldsymbol{X}$ 求协方差：

$$\begin{aligned}\boldsymbol{\Sigma}_X &= \frac{\left(\boldsymbol{X}-\mathrm{E}(\boldsymbol{X})\right)^{\mathrm{T}}\left(\boldsymbol{X}-\mathrm{E}(\boldsymbol{X})\right)}{n-1} \\ &= \frac{\left(\boldsymbol{Z}\boldsymbol{\Lambda}^{\frac{1}{2}}\boldsymbol{V}^{\mathrm{T}}\right)^{\mathrm{T}}\boldsymbol{Z}\boldsymbol{\Lambda}^{\frac{1}{2}}\boldsymbol{V}^{\mathrm{T}}}{n-1} = \left(\boldsymbol{\Lambda}^{\frac{1}{2}}\boldsymbol{V}^{\mathrm{T}}\right)^{\mathrm{T}}\frac{\overbrace{\boldsymbol{Z}^{\mathrm{T}}\boldsymbol{Z}}^{\boldsymbol{\Sigma}_Z}}{n-1}\boldsymbol{\Lambda}^{\frac{1}{2}}\boldsymbol{V}^{\mathrm{T}} = \boldsymbol{V}\boldsymbol{\Lambda}^{\frac{1}{2}}\boldsymbol{\Lambda}^{\frac{1}{2}}\boldsymbol{V}^{\mathrm{T}} = \boldsymbol{\Sigma}\end{aligned} \tag{36}$$

请大家利用这条技术路径生成图 25 和图 26。

一组特殊的平行四边形

对比 (22) 和 (35)，大家可能已经发现 $\boldsymbol{R}$ 相当于 $\boldsymbol{\Lambda}^{\frac{1}{2}}\boldsymbol{V}^{\mathrm{T}}$。而 $\boldsymbol{R}$ 和 $\boldsymbol{\Lambda}^{\frac{1}{2}}\boldsymbol{V}^{\mathrm{T}}$ 相当于协方差矩阵 $\boldsymbol{\Sigma}$ 的“平方根”。这说明协方差矩阵 $\boldsymbol{\Sigma}$ 的“平方根”不唯一。《矩阵力量》反复强调过这一点。

这意味着，凡是能够写成如下形式的矩阵 $\boldsymbol{B}$ 都是协方差矩阵 $\boldsymbol{\Sigma}$ 的“平方根”：

$$\boldsymbol{\Sigma} = \boldsymbol{B}^{\mathrm{T}}\boldsymbol{B} \tag{37}$$

比如，

$$\begin{aligned}\boldsymbol{\Sigma} &= \boldsymbol{R}^{\mathrm{T}}\boldsymbol{R} \\ \boldsymbol{\Sigma} &= \left(\boldsymbol{\Lambda}^{\frac{1}{2}}\boldsymbol{V}^{\mathrm{T}}\right)^{\mathrm{T}}\boldsymbol{\Lambda}^{\frac{1}{2}}\boldsymbol{V}^{\mathrm{T}}\end{aligned} \tag{38}$$

而上两式代表完全不同的几何变换。如图 29 (a) 所示，我们能够明显地看到 Cholesky 分解中的剪切操作。图 29 (b) 则明显地看出来旋转。

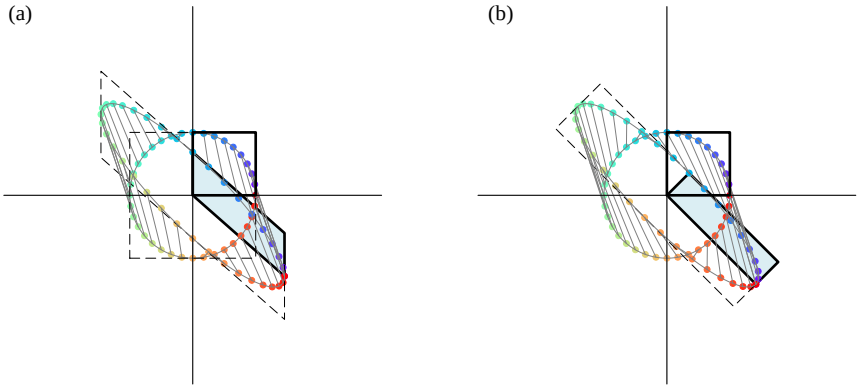


图 29. 对比 Cholesky 分解和特征值分解

更一般地，用数据矩阵 $\boldsymbol{Z}$ 代表正圆，即 $N(\boldsymbol{0}, \boldsymbol{I})$。$\boldsymbol{Z}$ 先经过 $\boldsymbol{U}$ 旋转，然后再用 $\boldsymbol{R}$ 完成“缩放 + 剪切”，最后用 $\mathrm{E}(\boldsymbol{X})$ 平移，得到数据矩阵 $\boldsymbol{X}$，这个过程对应的矩阵运算为：

$$\underset{N(\boldsymbol{0},\boldsymbol{\Sigma})}{\boldsymbol{X}} = \underset{N(\boldsymbol{0},\boldsymbol{I})}{\boldsymbol{Z}} \underset{\text{Rotate}}{\boldsymbol{U}} \underset{\text{Scale + shear}}{\boldsymbol{R}} + \underset{\text{Translate}}{\mathrm{E}(\boldsymbol{X})} \tag{39}$$

注意，$\boldsymbol{U}$ 提供旋转操作，因此 $\boldsymbol{U}$ 是正交矩阵，满足 $\boldsymbol{U}^{\mathrm{T}}\boldsymbol{U} = \boldsymbol{U}\boldsymbol{U}^{\mathrm{T}} = \boldsymbol{I}$。

计算 $\boldsymbol{X}$ 协方差矩阵，结果还是 $\boldsymbol{\Sigma}$：

$$\begin{aligned} \boldsymbol{\Sigma}_{\boldsymbol{X}} &= \frac{\left(\boldsymbol{X} - \mathrm{E}(\boldsymbol{X})\right)^{\mathrm{T}}\left(\boldsymbol{X} - \mathrm{E}(\boldsymbol{X})\right)}{n-1} \\ &= \frac{(\boldsymbol{Z}\boldsymbol{U}\boldsymbol{R})^{\mathrm{T}}\boldsymbol{Z}\boldsymbol{U}\boldsymbol{R}}{n-1} = (\boldsymbol{U}\boldsymbol{R})^{\mathrm{T}}\underbrace{\frac{\boldsymbol{Z}^{\mathrm{T}}\boldsymbol{Z}}{n-1}}_{\boldsymbol{\Sigma}_{\boldsymbol{Z}}}\boldsymbol{U}\boldsymbol{R} = \boldsymbol{R}^{\mathrm{T}}\boldsymbol{U}^{\mathrm{T}}\boldsymbol{U}\boldsymbol{R} = \boldsymbol{\Sigma} \end{aligned} \tag{40}$$

也就是说，给定不同的旋转矩阵 $\boldsymbol{U}$，我们就可以获得不同的 $\boldsymbol{\Sigma}$ 平方根 $\boldsymbol{U}\boldsymbol{R}$。也就相当于，这些完全不同的 $\boldsymbol{U}\boldsymbol{R}$ 都可以获得满足特定相关性条件随机数。

图 30 左上角第一幅子图实际上就是图 29 (b) 特征分解对应的几何变换。

图 30 所示为一系列不同旋转矩阵 $\boldsymbol{U}$，在这些 $\boldsymbol{U}$ 的作用下，我们最终都获得了相同的椭圆。但是仔细观察，会发现“彩灯”的运动轨迹完全不同。

旋转矩阵 $\boldsymbol{U}$ 作用于单位圆，不改变单位圆的解析式。但是，$\boldsymbol{U}$ 却改变了“彩灯”的位置。这实际上也回答了《矩阵力量》第 14 章有关“彩灯”位置的问题。

图 30 中一系列平行四边形都和旋转椭圆相切。相比旋转椭圆，这些平行四边形更能体现 $\boldsymbol{U}\boldsymbol{R}$ 的几何变换。

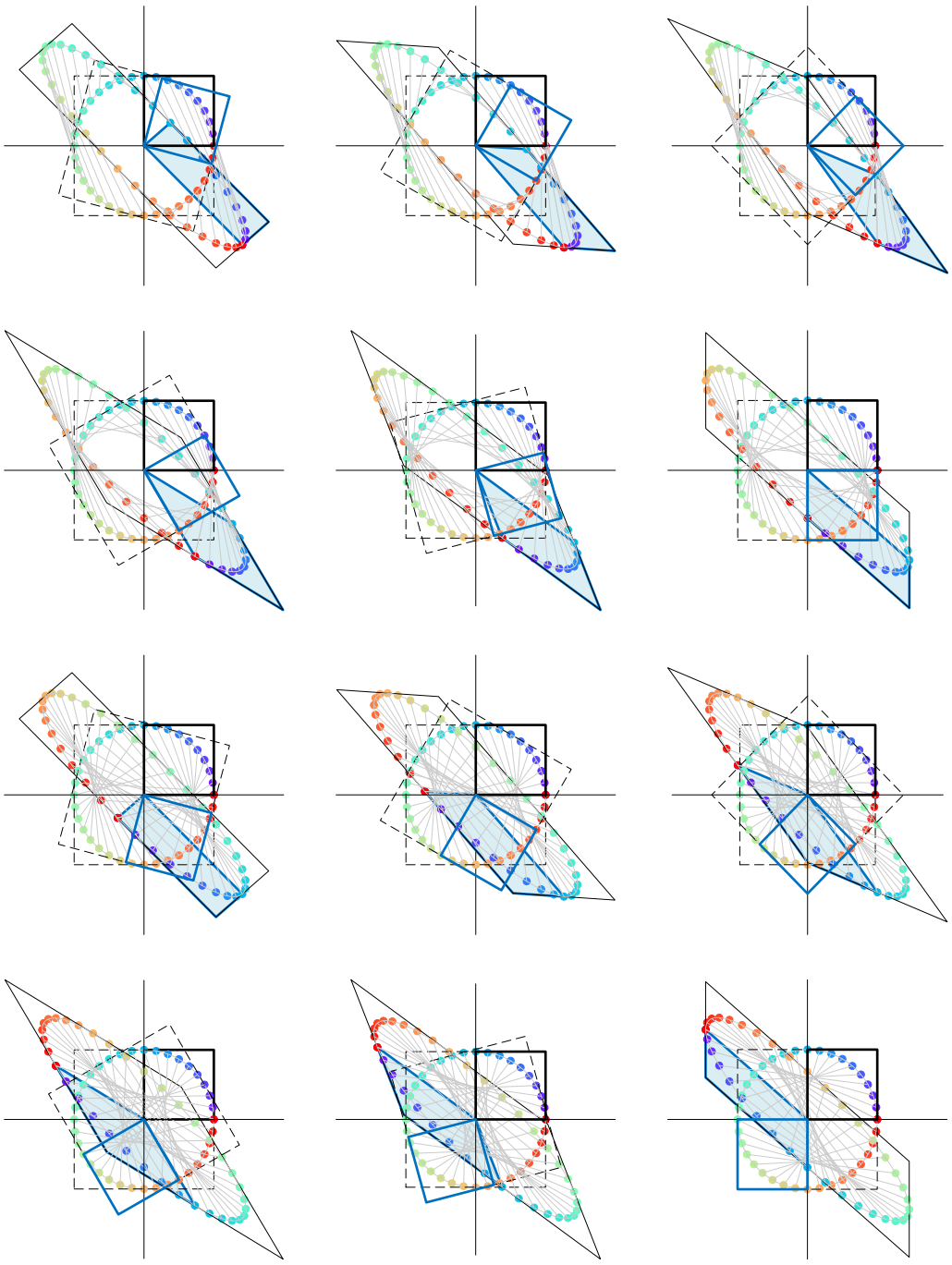
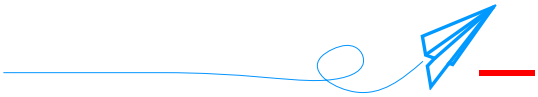


图 30. 不同的旋转矩阵

我们用 Streamlit 制作了一个应用，可视化图 30 不同角度的子图。请大家参考 Streamlit_Bk5_Ch15_10.py。



蒙特卡罗模拟的基本思想是利用随机抽样的方法来生成一组服从特定概率分布的随机数，然后用这些随机数代替原始问题中的未知量，计算问题的输出结果。通过对大量随机数进行抽样和统计，可以获得问题的近似解，从而分析问题的性质和特点。

蒙特卡罗模拟广泛应用于金融、物理、工程、生物、环境、社会科学等领域，例如金融风险评估、物理系统建模、生物统计、环境影响评价、社会网络分析等。它是一种高度灵活和通用的计算方法，可以适用于各种不同的问题和应用场景。本书后续会用马尔科夫链蒙特卡罗模拟完成贝叶斯推断。此外，《数据有道》将会继续这一话题。