11 Matrix Decompositions
矩阵分解

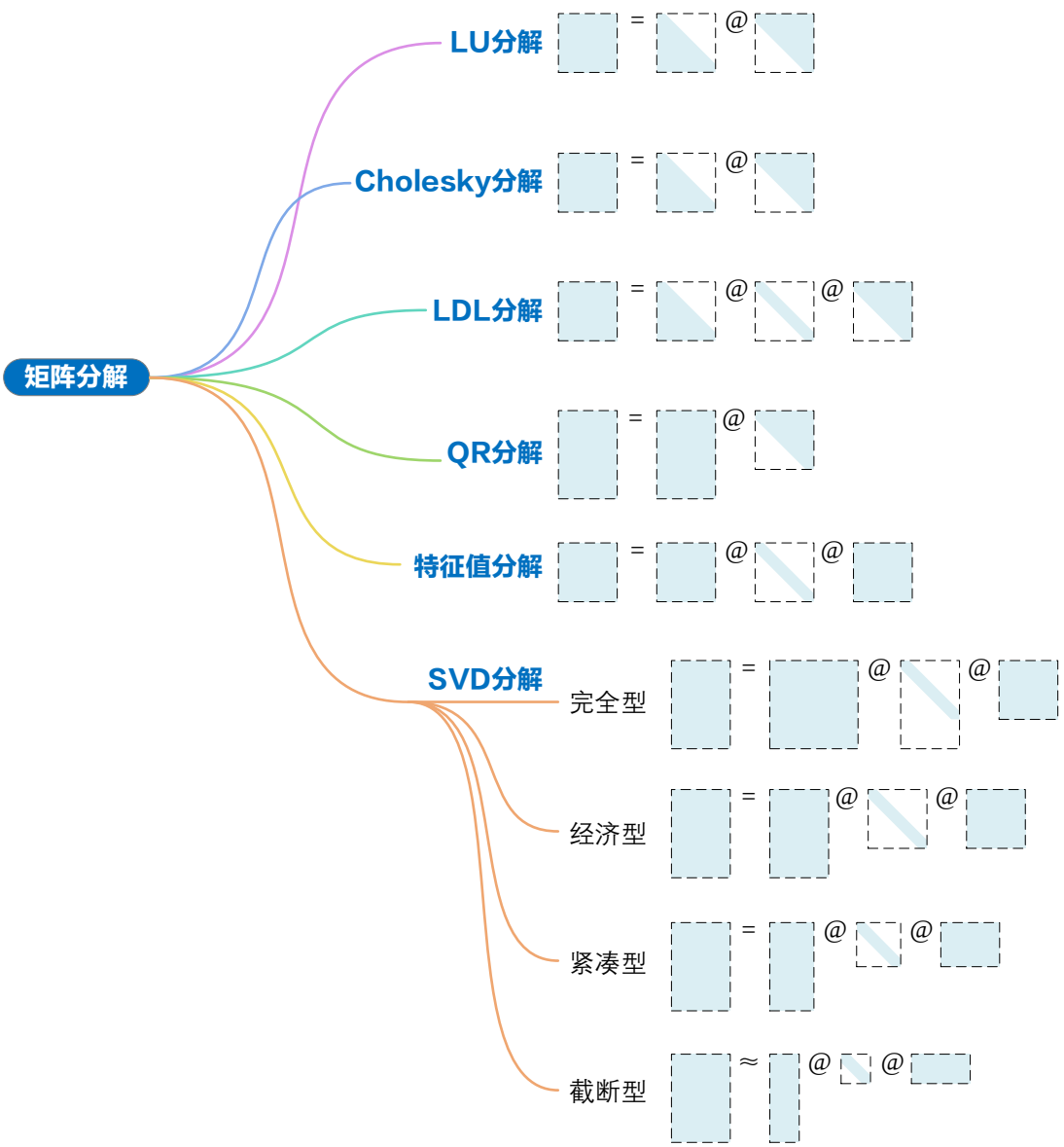
将矩阵拆解为其组成部分，从而简化运算

宇宙是一部鸿篇巨制，只有掌握它的文字和语言的人才能读懂宇宙；而数学便是解密宇宙的语言。

The universe is a grand book which cannot be read until one first learns to comprehend the language and become familiar with the characters in which it is composed. It is written in the language of mathematics.

—— 伽利略·伽利莱 (Galilei Galileo) | 意大利物理学家、数学家及哲学家 | 1564 ~ 1642

- `matplotlib.pyplot.contour()` 绘制等高线图
- `matplotlib.pyplot.contourf()` 绘制填充等高线图
- `numpy.linalg.cholesky()` Cholesky 分解
- `numpy.linalg.eig()` 特征值分解
- `numpy.linalg.qr()` QR 分解
- `numpy.linalg.svd()` 奇异值分解
- `numpy.meshgrid()` 生成网格化数据
- `scipy.linalg.ldl()` LDL 分解
- `scipy.linalg.lu()` LU 分解
- `seaborn.heatmap()` 绘制热图



11.1 矩阵分解：类似因式分解

矩阵分解 (matrix decomposition) 将矩阵解构得到其组成部分，类似代数中的因式分解。

从矩阵乘法角度，矩阵分解将矩阵拆解为若干矩阵的乘积。

从几何角度角度，这些组成部分可能对应缩放、旋转、投影、剪切等等各种几何变换。而原矩阵就是这些几何变换按特定次序的叠加。

数据科学和机器学习很多算法都直接依赖矩阵分解。本章全景介绍以下几种矩阵分解：

- **LU 分解** (lower–upper decomposition, LU decomposition)；
- **Cholesky 分解** (Cholesky decomposition, Cholesky factorization)；
- **LDL 分解** (lower-diagonal-lower transposed decomposition, LDL/LDLT decomposition)；
- **QR 分解** (QR decomposition) 实际上就是本书前文介绍的 Gram-Schmidt 正交化；
- **特征值分解** (eigendecomposition)；
- **SVD 分解** (singular value decomposition)。

在数据分析和机器学习很多算法中，Cholesky 分解、特征值分解和 SVD 分解应用广泛，本书后续将专门讲解这三种矩阵分解。

11.2 LU 分解：上下三角

一说，**LU 分解** (lower–upper decomposition, LU decomposition) 由图灵 (Alan Turing) 于 1948 年发明；另一说，波兰数学家 Tadeusz Banachiewicz 于 1938 年发明。

LU 分解将一个方阵 $\boldsymbol{A}$，分解为一个**上三角矩阵** (upper triangular matrix) $\boldsymbol{L}$ 和一个**下三角矩阵** (lower triangular matrix) $\boldsymbol{U}$ 的乘积，即，

$$\boldsymbol{A} = \boldsymbol{L}\boldsymbol{U} \tag{1}$$

(1) 展开得到：

$$\begin{bmatrix} a_{1,1} & a_{1,2} & \dots & a_{1,m} \\ a_{2,1} & a_{2,2} & \dots & a_{2,m} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m,1} & a_{m,2} & \dots & a_{m,m} \end{bmatrix}_{m\times m} = \begin{bmatrix} l_{1,1} & 0 & \cdots & 0 \\ l_{2,1} & l_{2,2} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ l_{m,1} & l_{m,2} & \cdots & l_{m,m} \end{bmatrix}_{m\times m} \begin{bmatrix} u_{1,1} & u_{1,2} & \cdots & u_{1,m} \\ 0 & u_{2,2} & \cdots & u_{2,m} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & u_{m,m} \end{bmatrix}_{m\times m} \tag{2}$$

图 1 所示为 LU 分解对应的矩阵运算示意图。LU 分解可以被视为**高斯消元法** (Gaussian elimination) 的矩阵形式。

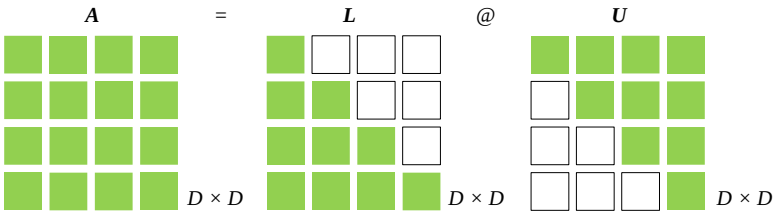


图 1. LU 分解

常用 `scipy.linalg.lu()` 函数进行 LU 分解。注意，`scipy.linalg.lu()` 默认进行 PLU 分解，即，

$$\boldsymbol{A} = \boldsymbol{PLU} \tag{3}$$

其中，$\boldsymbol{P}$ 为**置换矩阵** (permutation matrix)。`scipy.linalg.lu()` 函数得到的矩阵 $\boldsymbol{L}$ 主对角线均为 1。

前文介绍过，置换矩阵的任意一行或列只有一个 1，剩余均为 0。置换矩阵的作用是交换矩阵的行或列。注意，置换矩阵的逆还是置换矩阵，置换矩阵必定是正交矩阵。

PLU 分解有很高的数值稳定性。更重要的是，所有的方阵都可以进行 PLU 分解。图 2 所示为对方阵 $\boldsymbol{A}$ 进行 PLU 分解运算热图。

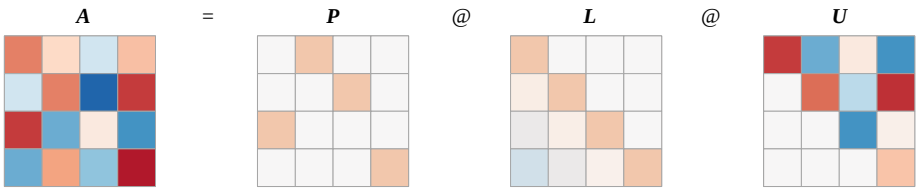
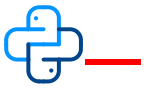


图 2. 对矩阵 $\boldsymbol{A}$ 的 PLU 分解热图



`Bk4_Ch11_01.py` 绘制图 2。

```
# Bk4_Ch11_01.py

import numpy as np
import seaborn as sns
from scipy.linalg import lu
from matplotlib import pyplot as plt

A = np.array([[ 5,  2, -2,  3],
              [-2,  5, -8,  7],
              [ 7, -5,  1, -6],
              [-5,  4, -4,  8]])
```

```
P,L,U = lu(A, permute_l = False)
# P, permutation matrix
# L, lower triangular with unit diagonal elements
# U, upper triangular
# Default do not perform the multiplication P*L

fig, axs = plt.subplots(1, 7, figsize=(12, 3))

plt.sca(axs[0])
ax = sns.heatmap(A,cmap='RdBu_r',vmax = 10,vmin = -10,
                 cbar_kws={"orientation": "horizontal"})
ax.set_aspect("equal")
plt.title('A')

plt.sca(axs[1])
plt.title('=')
plt.axis('off')

plt.sca(axs[2])
ax = sns.heatmap(P,cmap='RdBu_r',vmax = 10,vmin = -10,
                 cbar_kws={"orientation": "horizontal"})
ax.set_aspect("equal")
plt.title('P')

plt.sca(axs[3])
plt.title('@')
plt.axis('off')

plt.sca(axs[4])
ax = sns.heatmap(L,cmap='RdBu_r',vmax = 10,vmin = -10,
                 cbar_kws={"orientation": "horizontal"})
ax.set_aspect("equal")
plt.title('L')

plt.sca(axs[5])
plt.title('@')
plt.axis('off')

plt.sca(axs[6])
ax = sns.heatmap(U,cmap='RdBu_r', vmax = 10,vmin = -10,
                 cbar_kws={"orientation": "horizontal"})
ax.set_aspect("equal")
plt.title('U')
```

11.3 Cholesky 分解：得到数据矩阵的坐标

Cholesky 分解 (Cholesky decomposition) 是 LU 分解的特例。Cholesky 分解把矩阵分解为一个下三角矩阵以及它的转置矩阵的乘积：

$$\boldsymbol{A} = \boldsymbol{L}\boldsymbol{L}^{\mathrm{T}} \tag{4}$$

也就是说：

$$\begin{bmatrix} a_{1,1} & a_{1,2} & \ldots & a_{1,m} \\ a_{2,1} & a_{2,2} & \ldots & a_{2,m} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m,1} & a_{m,2} & \ldots & a_{m,m} \end{bmatrix}_{m\times m} = \begin{bmatrix} l_{1,1} & 0 & \cdots & 0 \\ l_{2,1} & l_{2,2} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ l_{m,1} & l_{m,2} & \cdots & l_{m,m} \end{bmatrix}_{m\times m} \begin{bmatrix} l_{1,1} & l_{2,1} & \cdots & l_{m,1} \\ 0 & l_{2,2} & \cdots & l_{m,2} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & l_{m,m} \end{bmatrix}_{m\times m} \tag{5}$$

当然，Cholesky 分解也可以写成：

$$\boldsymbol{A} = \boldsymbol{R}^{\mathrm{T}}\boldsymbol{R} \tag{6}$$

其中，$\boldsymbol{R} = \boldsymbol{L}^{\mathrm{T}}$。

图 3 所示为 Cholesky 分解矩阵运算示意图。本节标题说 Cholesky 分解“得到数据矩阵的坐标”，这个话题留给下一章。

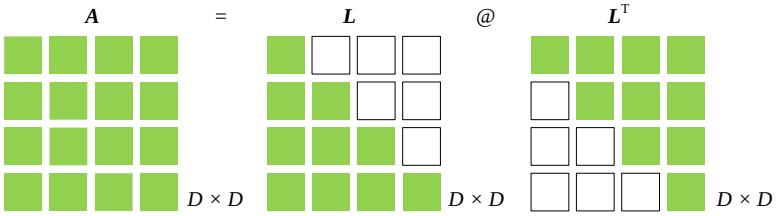


图 3. Cholesky 分解矩阵运算

Numpy 中进行 Cholesky 分解的函数为 `numpy.linalg.cholesky()`。请读者自行编写代码并绘制图 4。

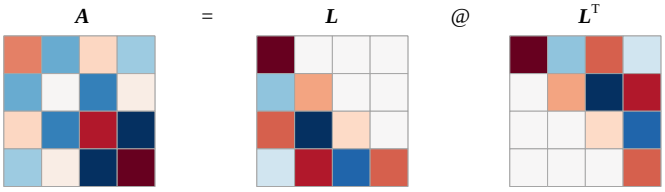


图 4. Cholesky 分解示例

注意，只有正定矩阵 (positive definite matrix) 才能 Cholesky 分解。下一章将浮光掠影介绍正定性及其几何内涵。本书后续将专门讲解正定性。

丛书在讲解**协方差矩阵** (covariance matrix)、数据转换、蒙特卡洛模拟等内容都会使用 Cholesky 分解。下一章将专门从几何角度介绍 Cholesky 分解。

LDL 分解：Cholesky 分解的扩展

Cholesky 分解可以进一步扩展为 **LDL 分解** (LDL decomposition)：

$$\boldsymbol{A} = \boldsymbol{L}\boldsymbol{D}\boldsymbol{L}^{\mathrm{T}} = \boldsymbol{L}\boldsymbol{D}^{1/2}\left(\boldsymbol{D}^{1/2}\right)\boldsymbol{L}^{\mathrm{T}} = \boldsymbol{L}\boldsymbol{D}^{1/2}\left(\boldsymbol{L}\boldsymbol{D}^{1/2}\right)^{\mathrm{T}} \tag{7}$$

其中，$\boldsymbol{L}$ 为下三角矩阵，但是对角线元素均为 1；$\boldsymbol{D}$ 为对角矩阵。

(7) 展开来写得到：

$$\begin{bmatrix} a_{1,1} & a_{1,2} & \dots & a_{1,m} \\ a_{2,1} & a_{2,2} & \dots & a_{2,m} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m,1} & a_{m,2} & \dots & a_{m,m} \end{bmatrix}_{m\times m} = \begin{bmatrix} 1 & 0 & \cdots & 0 \\ l_{2,1} & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ l_{m,1} & l_{m,2} & \cdots & 1 \end{bmatrix}_{m\times m} \begin{bmatrix} d_{1,1} & 0 & \cdots & 0 \\ 0 & d_{2,2} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & d_{m,m} \end{bmatrix}_{m\times m} \begin{bmatrix} 1 & l_{2,1} & \cdots & l_{m,1} \\ 0 & 1 & \cdots & l_{m,2} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{bmatrix}_{m\times m} \tag{8}$$

图 5 所示为 LDL 分解矩阵运算示意图。

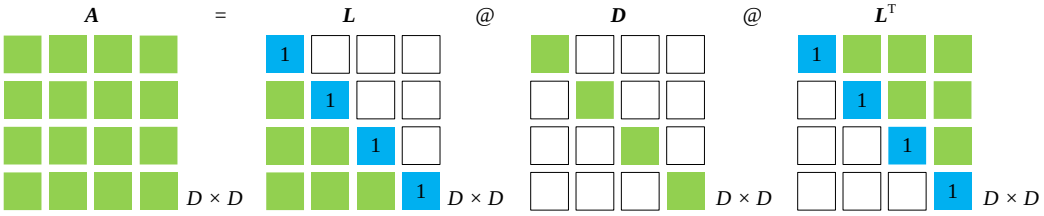


图 5. LDL 分解矩阵运算示意图

LDL 分解的函数为 `scipy.linalg.ldl()`，注意这个函数的返回结果也包括置换矩阵。图 6 所示为 LDL 分解运算热图。请读者根据前文代码自行绘制图 6。

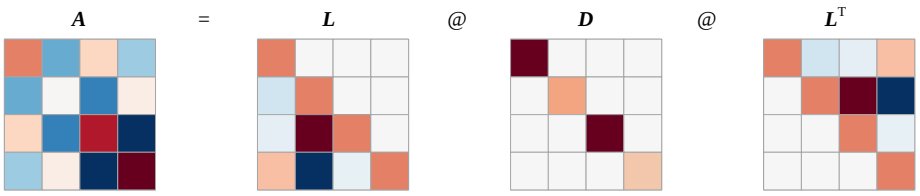


图 6. LDL 分解示例热图

11.4 QR 分解：正交化

QR 分解 (QR decomposition, QR factorization) 和上一节介绍的格拉姆-斯密特正交化联系紧密。QR 分解有两种常见形式：

- **完全型** (complete)，$\boldsymbol{Q}$ 为方阵；
- **缩略型** (reduced)。

图 7 所示为对 $n \times D$ 数据矩阵 $\boldsymbol{X}$ 进行完全型 QR 分解示意图，对应的等式为：

$$\boldsymbol{X}_{n\times D} = \boldsymbol{Q}_{n\times n}\boldsymbol{R}_{n\times D} \tag{9}$$

其中，$\boldsymbol{Q}$ 为方阵，形状为 $n \times n$；$\boldsymbol{R}$ 和 $\boldsymbol{X}$ 形状一致。注意，任何实数矩阵都可以 QR 分解。

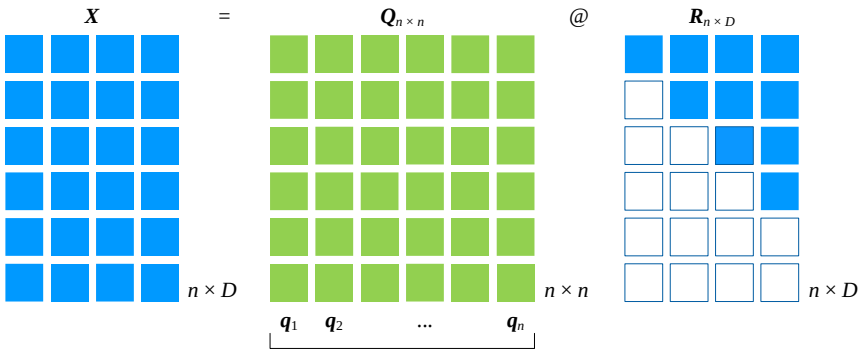


图 7. 完全型 QR 分解示意图

图 8 所示为对某个细高数据矩阵 $\boldsymbol{X}$ 进行完全型 QR 分解运算热图。

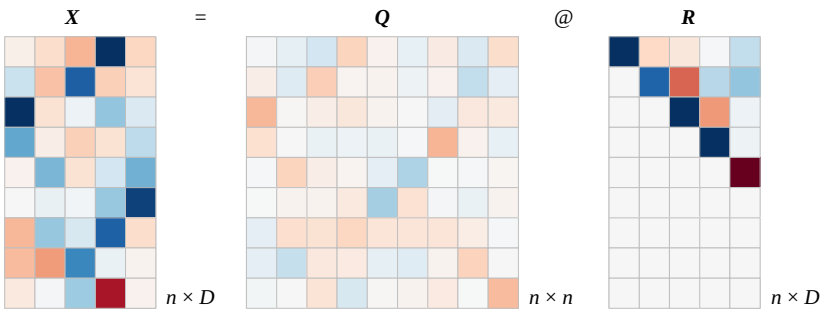


图 8. 完全型 QR 分解热图

$\boldsymbol{Q}$ 为正交矩阵，也就是说：

$$\boldsymbol{Q}_{n\times n}\boldsymbol{Q}_{n\times n}^{\mathrm{T}}=\boldsymbol{Q}_{n\times n}^{\mathrm{T}}\boldsymbol{Q}_{n\times n}=\boldsymbol{I}_{n\times n} \tag{10}$$

图 9 所示为 (10) 运算对应热图。根据本书前文内容，$\boldsymbol{Q}$ 的列向量组 $[\boldsymbol{q}_1, \boldsymbol{q}_2, \ldots, \boldsymbol{q}_D]$ 是一个规范正交基。

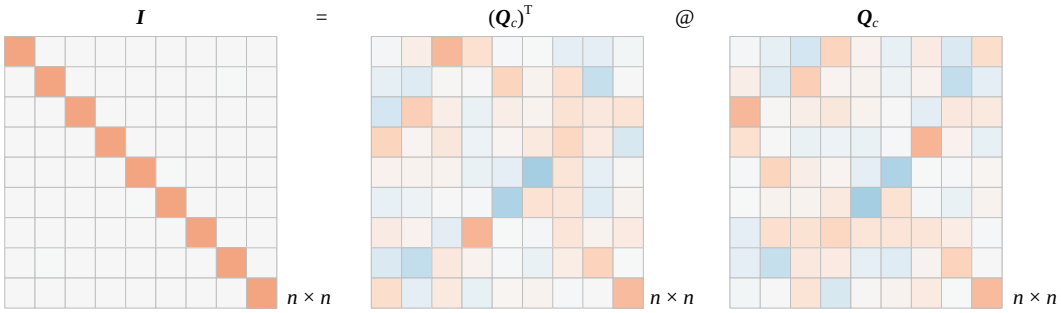


图 9. $\boldsymbol{Q}_c$ 为正交矩阵

把 $\boldsymbol{Q}$ 展开写成 $[\boldsymbol{q}_1, \boldsymbol{q}_2, \ldots, \boldsymbol{q}_D]$，$\boldsymbol{X}$ 的第一列向量 $\boldsymbol{x}_1$ 可以通过下式得到：

$$\boldsymbol{x}_1 = \begin{bmatrix} \boldsymbol{q}_1 & \boldsymbol{q}_2 & \cdots & \boldsymbol{q}_D \end{bmatrix} \begin{bmatrix} r_{1,1} \\ r_{2,1} \\ \vdots \\ r_{D,1} \end{bmatrix} = r_{1,1}\boldsymbol{q}_1 + \underbrace{r_{2,1}\boldsymbol{q}_2}_{=0} + \cdots \underbrace{r_{D,1}\boldsymbol{q}_D}_{=0} = r_{1,1}\boldsymbol{q}_1 \tag{11}$$

上式相当于 $\boldsymbol{x}_1$ 在 规范正交基 $[\boldsymbol{q}_1, \boldsymbol{q}_2, \ldots, \boldsymbol{q}_D]$ 张成的空间坐标为 $(r_{1,1}, r_{2,1}, \ldots, r_{D,1})$，即 $(r_{1,1}, 0, \ldots, 0)$。也就是，$\boldsymbol{x}_1$ 和 $\boldsymbol{q}_1$ 在同一条直线上，方向同向或反向。这个本书前文介绍的格拉姆-施密特正交化第一步一致。

$\boldsymbol{q}_1$ 是单位向量，也就是说：

$$r_{1,1} = \pm \left\| \boldsymbol{x}_1 \right\| \tag{12}$$

这一点已经说明 QR 分解结果不唯一。但是，如果 $\boldsymbol{X}$ 列满秩，且 $\boldsymbol{R}$ 的对角元素为正实数的情况下是 QR 分解唯一。

类似地，$\boldsymbol{X}$ 的第二列向量 $\boldsymbol{x}_2$ 写成：

$$\boldsymbol{x}_2 = \begin{bmatrix} \boldsymbol{q}_1 & \boldsymbol{q}_2 & \cdots & \boldsymbol{q}_D \end{bmatrix} \begin{bmatrix} r_{1,2} \\ r_{2,2} \\ \vdots \\ r_{D,2} \end{bmatrix} = r_{1,2}\boldsymbol{q}_1 + r_{2,2}\boldsymbol{q}_2 + \underbrace{r_{3,2}\boldsymbol{q}_3}_{=0} + \cdots \underbrace{r_{D,2}\boldsymbol{q}_D}_{=0} = r_{1,2}\boldsymbol{q}_1 + r_{2,2}\boldsymbol{q}_2 \tag{13}$$

$\boldsymbol{X}_2$ 在规范正交基 $[\boldsymbol{q}_1, \boldsymbol{q}_2, \ldots, \boldsymbol{q}_D]$ 张成的空间坐标为 $(r_{1,2}, r_{2,2}, r_{3,2}, \ldots, r_{D,1})$，即 $(r_{1,2}, r_{2,2}, 0, \ldots, 0)$。

缩略型

图 7 对应的完全型 QR 分解可以进一步简化。将 (9) 中 $\boldsymbol{R}$ 上下切一刀，让上方子块为方阵，这样 (9) 可以写成分块矩阵乘法：

$$\boldsymbol{X} = \begin{bmatrix} \boldsymbol{Q}_{n\times D} & \boldsymbol{Q}_{n\times(n-D)} \end{bmatrix} \begin{bmatrix} \boldsymbol{R}_{D\times D} \\ \boldsymbol{O}_{(n-D)\times D} \end{bmatrix} = \boldsymbol{Q}_{n\times D}\boldsymbol{R}_{D\times D} + \boldsymbol{Q}_{n\times(n-D)}\boldsymbol{O}_{(n-D)\times D} = \boldsymbol{Q}_{n\times D}\boldsymbol{R}_{D\times D} \tag{14}$$

其中 $\boldsymbol{Q}_{n\times D}$ 和 $\boldsymbol{X}$ 矩阵形状相同，而 $\boldsymbol{R}_{D\times D}$ 为上三角矩阵。

图 10 所示为 QR 分解从完全型到缩略型简化过程。

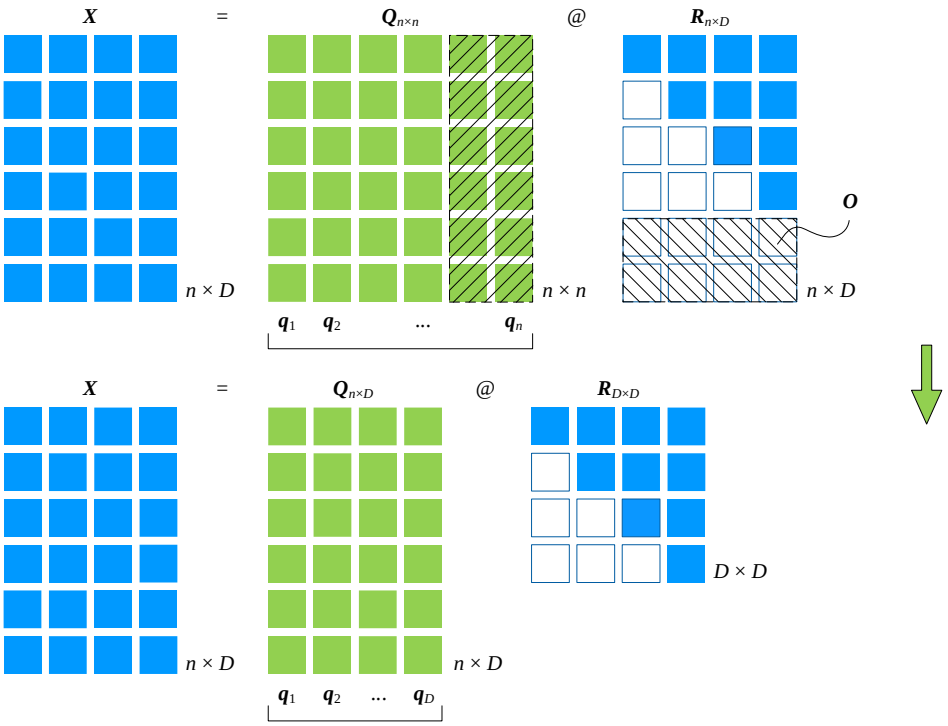


图 10. QR 分解从完全型到缩略型简化过程

图 11 所示为对矩阵 $\boldsymbol{X}$ 进行缩略型 QR 分解运算热图。

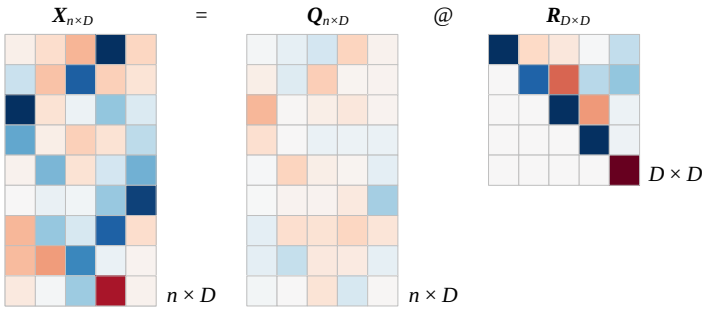


图 11. 缩略型 QR 分解热图

列向量两两正交

虽然 (14) 中矩阵 $\boldsymbol{Q}_{n\times D}$ 不是一个方阵，但列向量也两两正交，因为，

$$\left(\boldsymbol{Q}_{n\times D}\right)^{\mathrm{T}}\boldsymbol{Q}_{n\times D}=\boldsymbol{I}_{D\times D} \tag{15}$$

把 $\boldsymbol{Q}$ 展开写成 $[\boldsymbol{q}_1, \boldsymbol{q}_2, \ldots, \boldsymbol{q}_D]$，代入上式得到：

$$
\begin{aligned}
\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{Q} &= \begin{bmatrix} \boldsymbol{q}_1^{\mathrm{T}} \\ \boldsymbol{q}_2^{\mathrm{T}} \\ \vdots \\ \boldsymbol{q}_D^{\mathrm{T}} \end{bmatrix} \begin{bmatrix} \boldsymbol{q}_1 & \boldsymbol{q}_2 & \cdots & \boldsymbol{q}_D \end{bmatrix} \\
&= \begin{bmatrix} \boldsymbol{q}_1^{\mathrm{T}}\boldsymbol{q}_1 & \boldsymbol{q}_1^{\mathrm{T}}\boldsymbol{q}_2 & \cdots & \boldsymbol{q}_1^{\mathrm{T}}\boldsymbol{q}_D \\ \boldsymbol{q}_2^{\mathrm{T}}\boldsymbol{q}_1 & \boldsymbol{q}_2^{\mathrm{T}}\boldsymbol{q}_2 & \cdots & \boldsymbol{q}_2^{\mathrm{T}}\boldsymbol{q}_D \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{q}_D^{\mathrm{T}}\boldsymbol{q}_1 & \boldsymbol{q}_D^{\mathrm{T}}\boldsymbol{q}_2 & \cdots & \boldsymbol{q}_D^{\mathrm{T}}\boldsymbol{q}_D \end{bmatrix} = \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{bmatrix}
\end{aligned} \tag{16}
$$

其中，$\boldsymbol{q}_j$向量为 n 行。图 12 所示为 $\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{Q}$ 运算对应的热图。

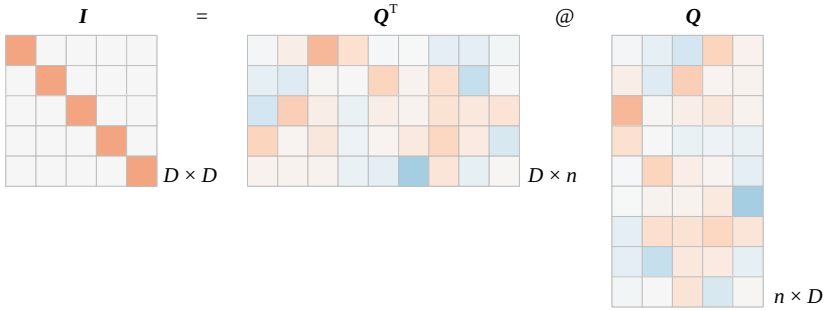


图 12. $\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{Q}$ 运算对应的热图

几何视角

QR 分解完成对数据矩阵 $\boldsymbol{X}$ 的正交化。$\boldsymbol{X}$ 的列向量 $[\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_D]$ 可能并非两两正交，经过 QR 分解得到的 $[\boldsymbol{q}_1, \boldsymbol{q}_2, \ldots, \boldsymbol{q}_D]$ 两两正交，且每个向量为单位向量。如图 13 所示，$[\boldsymbol{q}_1, \boldsymbol{q}_2, \ldots, \boldsymbol{q}_D]$ 本身就是一个规范正交基，它的重要特点是 $\boldsymbol{q}_1$ 平行于 $\boldsymbol{x}_1$，剩余的 $\boldsymbol{q}_j$ 是通过逐步正交投影得到的。

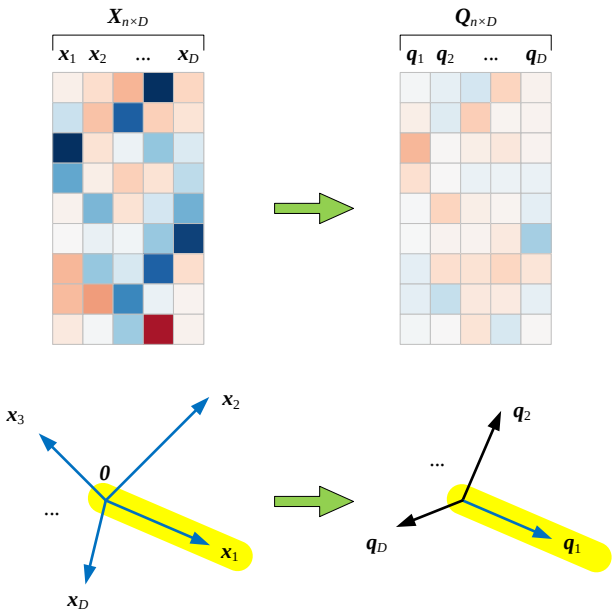
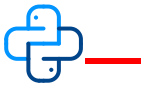


图 13. QR 分解背后的几何意义

当然，对原始数据矩阵 $\boldsymbol{X}$ 的正交化并不唯一，不同正交化方法得到的规范正交基也不同。本书后面还会介绍其他正交化方法，请大家注意区分结果的差异。



Bk4_Ch11_02.py 绘制本节热图。

```
# Bk4_Ch11_02.py

import numpy as np
import seaborn as sns
from matplotlib import pyplot as plt

# generate original data matrix, X
np.random.default_rng()
X = np.random.randn(9, 5)

#%% QR decomposition,  complete version

Q_complete, R_complete = np.linalg.qr(X, mode = 'complete')


fig, axs = plt.subplots(1, 5, figsize=(12, 3))

plt.sca(axs[0])
ax = sns.heatmap(X,cmap='RdBu_r',vmax = 2.5,vmin = -2.5,
                 cbar_kws={"orientation": "horizontal"})
ax.set_aspect("equal")
plt.title('X')

plt.sca(axs[1]); plt.title('='); plt.axis('off')

plt.sca(axs[2])
ax = sns.heatmap(Q_complete,cmap='RdBu_r',vmax = 2.5,vmin = -2.5,
                 cbar_kws={"orientation": "horizontal"})
ax.set_aspect("equal")
plt.title('Qc')

plt.sca(axs[3]); plt.title('@'); plt.axis('off')

plt.sca(axs[4])
ax = sns.heatmap(R_complete,cmap='RdBu_r',vmax = 2.5,vmin = -2.5,
                 cbar_kws={"orientation": "horizontal"})
ax.set_aspect("equal")
plt.title('Rc')

# properties of Q (reduced)

fig, axs = plt.subplots(1, 5, figsize=(12, 3))

plt.sca(axs[0])
ax = sns.heatmap(Q_complete.T@Q_complete,cmap='RdBu_r',vmax = 2.5,vmin = -2.5,
                 cbar_kws={"orientation": "horizontal"})
ax.set_aspect("equal")
plt.title('Qc.T@Qc')

plt.sca(axs[1]); plt.title('='); plt.axis('off')

plt.sca(axs[2])
ax = sns.heatmap(Q_complete.T,cmap='RdBu_r',vmax = 2.5,vmin = -2.5,
                 cbar_kws={"orientation": "horizontal"})
```

```
ax.set_aspect("equal")
plt.title('Qc.T')

plt.sca(axs[3]); plt.title('@'); plt.axis('off')

plt.sca(axs[4])
ax = sns.heatmap(Q_complete,cmap='RdBu_r',vmax = 2.5,vmin = -2.5,
                 cbar_kws={"orientation": "horizontal"})
ax.set_aspect("equal")
plt.title('Qc')

#%% QR decomposition,  reduced version

Q, R = np.linalg.qr(X)
# default: reduced

fig, axs = plt.subplots(1, 5, figsize=(12, 3))

plt.sca(axs[0])
ax = sns.heatmap(X,cmap='RdBu_r',vmax = 2.5,vmin = -2.5,
                 cbar_kws={"orientation": "horizontal"})
ax.set_aspect("equal")
plt.title('X')

plt.sca(axs[1]); plt.title('='); plt.axis('off')

plt.sca(axs[2])
ax = sns.heatmap(Q,cmap='RdBu_r',vmax = 2.5,vmin = -2.5,
                 cbar_kws={"orientation": "horizontal"})
ax.set_aspect("equal")
plt.title('Q')

plt.sca(axs[3]); plt.title('@'); plt.axis('off')

plt.sca(axs[4])
ax = sns.heatmap(R,cmap='RdBu_r',vmax = 2.5,vmin = -2.5,
                 cbar_kws={"orientation": "horizontal"})
ax.set_aspect("equal")
plt.title('R')

# properties of Q (reduced)

fig, axs = plt.subplots(1, 5, figsize=(12, 3))

plt.sca(axs[0])
ax = sns.heatmap(Q.T@Q,cmap='RdBu_r',vmax = 2.5,vmin = -2.5,
                 cbar_kws={"orientation": "horizontal"})
ax.set_aspect("equal")
plt.title('Q.T@Q')

plt.sca(axs[1]); plt.title('='); plt.axis('off')

plt.sca(axs[2])
ax = sns.heatmap(Q.T,cmap='RdBu_r',vmax = 2.5,vmin = -2.5,
                 cbar_kws={"orientation": "horizontal"})
ax.set_aspect("equal")
plt.title('Q.T')

plt.sca(axs[3]); plt.title('@'); plt.axis('off')

plt.sca(axs[4])
ax = sns.heatmap(Q,cmap='RdBu_r',vmax = 2.5,vmin = -2.5,
                 cbar_kws={"orientation": "horizontal"})
ax.set_aspect("equal")
plt.title('Q')
```

11.5 特征值分解：刻画矩阵引发的几何变换

特征值分解是矩阵分解中的一道“大菜”，本节仅仅介绍其皮毛。本书后续会有两章内容专门讲解特征值分解及其应用。

枯燥的定义

对于方阵 $\boldsymbol{A}$，如果存在**非零向量** (non-zero vector) $\boldsymbol{v}$ 使得：

$$\boldsymbol{A}\boldsymbol{v} = \lambda\boldsymbol{v} \tag{17}$$

$\boldsymbol{v}$ 就是的 $\boldsymbol{A}$ 的特征向量 (eigen vector)，标量 λ 被称作**特征值** (eigen value)。

(17) 可以写作：

$$\left(\boldsymbol{A} - \lambda\boldsymbol{I}\right)\boldsymbol{v} = 0 \tag{18}$$

其中，$\boldsymbol{I}$ 是**单位阵** (identity matrix)。

特征向量 $\boldsymbol{v}$ 代表方向，通常是列向量；特征值 λ 是在这个方向上的比例，特征值是标量。

注意，并不是所有方阵都可以特征值分解，只有可对角化矩阵 (diagonalizable matrix) 才能进行特征值分解。如果一个方阵 $\boldsymbol{A}$ 相似于对角矩阵，也就是说，如果存在一个可逆矩阵 P 使得矩阵乘积 $\boldsymbol{P}^{-1}\boldsymbol{A}\boldsymbol{P}$ 结果为对角矩阵，则 $\boldsymbol{A}$ 就被称为可对角化 (diagonalizable)。

大家是否还记得，本书前文讲解几何变换时提到，几何上，我们更喜欢看到对角阵，因为在空间中对角阵代表“立方体”。

二维方阵

假设某个二维方阵 $\boldsymbol{A}$，有两个特征值和特征向量：

$$\begin{aligned}\boldsymbol{A}\boldsymbol{v}_1 &= \lambda_1\boldsymbol{v}_1\\ \boldsymbol{A}\boldsymbol{v}_2 &= \lambda_2\boldsymbol{v}_2\end{aligned} \tag{19}$$

两个特征向量可以构成矩阵 $\boldsymbol{V}$，用两个特征值构造对角阵 $\boldsymbol{\Lambda}$，上式可以写成：

$$\boldsymbol{A}\underbrace{\begin{bmatrix}\boldsymbol{v}_1 & \boldsymbol{v}_2\end{bmatrix}}_{\boldsymbol{V}} = \underbrace{\begin{bmatrix}\boldsymbol{v}_1 & \boldsymbol{v}_2\end{bmatrix}}_{\boldsymbol{V}}\underbrace{\begin{bmatrix}\lambda_1 & 0\\ 0 & \lambda_2\end{bmatrix}}_{\boldsymbol{\Lambda}} \tag{20}$$

即，

$$\boldsymbol{A}\boldsymbol{V} = \boldsymbol{V}\boldsymbol{\Lambda} \tag{21}$$

(21) 可以进一步写成：

$$\boldsymbol{A}=\boldsymbol{V}\boldsymbol{\Lambda}\boldsymbol{V}^{-1} \tag{22}$$

上式就叫做矩阵 $\boldsymbol{A}$ 的**特征分解** (eigen-decomposition)。$\boldsymbol{\Lambda}$ 被称作特征值矩阵，$\boldsymbol{V}$ 被称作特征向量矩阵。

多维方阵

对于 $D \times D$ 方阵 $\boldsymbol{A}$，如果存在如下一系列等式：

$$\begin{cases} \boldsymbol{A}\boldsymbol{v}_1 = \lambda_1 \boldsymbol{v}_1 \\ \boldsymbol{A}\boldsymbol{v}_2 = \lambda_2 \boldsymbol{v}_2 \\ \quad \vdots \\ \boldsymbol{A}\boldsymbol{v}_D = \lambda_D \boldsymbol{v}_D \end{cases} \tag{23}$$

整理上式得到：

$$\begin{bmatrix} \boldsymbol{A}\boldsymbol{v}_1 & \boldsymbol{A}\boldsymbol{v}_2 & \cdots & \boldsymbol{A}\boldsymbol{v}_D \end{bmatrix} = \begin{bmatrix} \lambda_1\boldsymbol{v}_1 & \lambda_2\boldsymbol{v}_2 & \cdots & \lambda_D\boldsymbol{v}_D \end{bmatrix} \tag{24}$$

即，

$$\boldsymbol{A}\underbrace{\begin{bmatrix} \boldsymbol{v}_1 & \boldsymbol{v}_2 & \cdots & \boldsymbol{v}_D \end{bmatrix}}_{\boldsymbol{V}} = \underbrace{\begin{bmatrix} \boldsymbol{v}_1 & \boldsymbol{v}_2 & \cdots & \boldsymbol{v}_D \end{bmatrix}}_{\boldsymbol{V}} \underbrace{\begin{bmatrix} \lambda_1 & & & \\ & \lambda_2 & & \\ & & \ddots & \\ & & & \lambda_D \end{bmatrix}}_{\boldsymbol{\Lambda}} \tag{25}$$

特征多项式

方阵 $\boldsymbol{A}$ **特征多项式** (characteristic polynomial) 可以这样获得：

$$p(\lambda) = |\boldsymbol{A} - \lambda \boldsymbol{I}| \tag{26}$$

$\boldsymbol{A}$ 的**特征方程** (characteristic equation) 为：

$$|\boldsymbol{A} - \lambda \boldsymbol{I}| = 0 \tag{27}$$

特征方程可以用来求解矩阵的特征值，从而进一步求解对应的特征向量。

手算特征值分解

给定如下矩阵 $\boldsymbol{A}$：

$$\boldsymbol{A} = \begin{bmatrix} 1.25 & -0.75 \\ -0.75 & 1.25 \end{bmatrix} \tag{28}$$

方阵 $\boldsymbol{A}$ 的特征方程为：

$$\begin{aligned} p(\lambda) &= |\boldsymbol{A} - \lambda \boldsymbol{I}| = \begin{vmatrix} 1.25-\lambda & -0.75 \\ -0.75 & 1.25-\lambda \end{vmatrix} \\ &= \lambda^2 - 2.5\lambda + 1 = (\lambda - 2)(\lambda - 0.5) = 0 \end{aligned} \tag{29}$$

求解 (29) 所示一元二次方程，得到 $p(\lambda)$ 的两个根分别为：

$$\lambda_1 = 0.5, \quad \lambda_2 = 2 \tag{30}$$

对于 $\lambda_1 = 0.5$,

$$(\boldsymbol{A} - \lambda_1 \boldsymbol{I})\boldsymbol{v}_1 = \left\{ \begin{bmatrix} 1.25 & -0.75 \\ -0.75 & 1.25 \end{bmatrix} - \begin{bmatrix} 0.5 & 0 \\ 0 & 0.5 \end{bmatrix} \right\} \begin{bmatrix} v_{1,1} \\ v_{1,2} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \tag{31}$$

得到如下等式：

$$v_{1,1} - v_{1,2} = 0 \tag{32}$$

满足如上等式的向量都是特征向量，选择第一象限的单位向量为特征向量 $\boldsymbol{v}_1$

$$\boldsymbol{v}_1 = \begin{bmatrix} \sqrt{2}/2 \\ \sqrt{2}/2 \end{bmatrix} \tag{33}$$

这一步可以看出特征向量不唯一。

对于 $\lambda_2 = 2$,

$$(\boldsymbol{A} - \lambda_2 \boldsymbol{I})\boldsymbol{v}_2 = \left\{ \begin{bmatrix} 1.25 & -0.75 \\ -0.75 & 1.25 \end{bmatrix} - \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix} \right\} \begin{bmatrix} v_{2,1} \\ v_{2,2} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \tag{34}$$

得到如下等式：

$$v_{2,1} + v_{2,2} = 0 \tag{35}$$

同样，满足如上等式的向量都是特征向量，选择第二象限的单位向量为特征向量 $\boldsymbol{v}_2$

$$\boldsymbol{v}_2 = \begin{bmatrix} -\sqrt{2}/2 \\ \sqrt{2}/2 \end{bmatrix} \tag{36}$$

经过上述分析，大家已经清楚，某个特征值的特征向量并不唯一。

图 14 所示为候选特征向量之间关系。

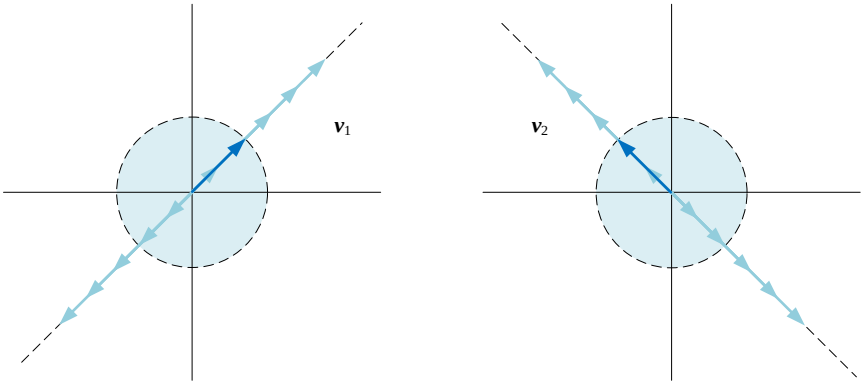


图 14. 候选特征向量

这样我们可以得到特征向量矩阵 $\boldsymbol{V}$:

$$\boldsymbol{V}=\begin{bmatrix}\sqrt{2}/2 & -\sqrt{2}/2\\ \sqrt{2}/2 & \sqrt{2}/2\end{bmatrix} \tag{37}$$

$\boldsymbol{V}$ 的逆为:

$$\boldsymbol{V}^{-1}=\begin{bmatrix}\sqrt{2}/2 & \sqrt{2}/2\\ -\sqrt{2}/2 & \sqrt{2}/2\end{bmatrix} \tag{38}$$

大家可能已经发现:

$$\boldsymbol{V}^{\mathrm{T}}=\boldsymbol{V}^{-1} \tag{39}$$

这是因为 (28) 中 $\boldsymbol{A}$ 为对称矩阵。这是特征值分解的一种特殊情况。

对称矩阵

如果 $\boldsymbol{A}$ 为对称矩阵，则 (22) 可以写作:

$$\boldsymbol{A}=\boldsymbol{V}\boldsymbol{\Lambda}\boldsymbol{V}^{\mathrm{T}} \tag{40}$$

$\boldsymbol{V}$ 为正交矩阵，即满足:

$$\boldsymbol{V}\boldsymbol{V}^{\mathrm{T}}=\boldsymbol{I} \tag{41}$$

几何视角

对于一个细高的长方形矩阵 $\boldsymbol{X}$ 来说，它本身肯定不能进行特征值分解。但是，它的两个格拉姆矩阵 $\boldsymbol{X}^{\mathrm{T}}\boldsymbol{X}$ 和 $\boldsymbol{X}\boldsymbol{X}^{\mathrm{T}}$ 都是对称阵！也就是说，如图 15 所示，$\boldsymbol{X}^{\mathrm{T}}\boldsymbol{X}$ 和 $\boldsymbol{X}\boldsymbol{X}^{\mathrm{T}}$ 都可以进行特征值分解，而且分解得到的特征向量矩阵 $\boldsymbol{V}$ 和 $\boldsymbol{U}$ 都是正交矩阵，也就是对 $\boldsymbol{X}$ 正交化。之所以用 $\boldsymbol{V}$ 和 $\boldsymbol{U}$ 分别表达特征向量矩阵，是为了和下一章奇异值分解相呼应。

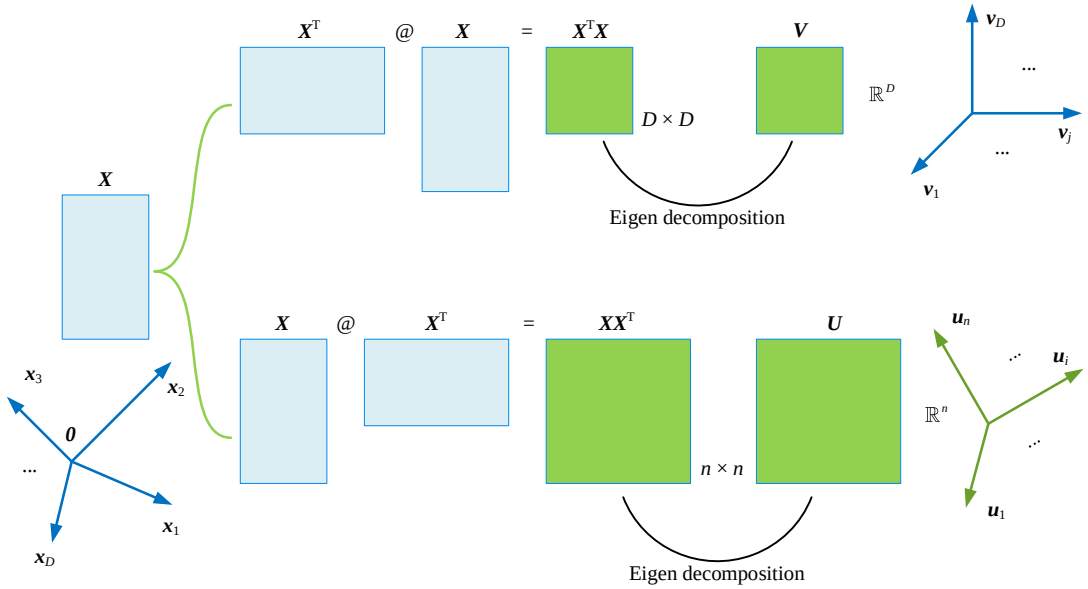


图 15. 对 Gram 矩阵特征值分解

特征值分解在数据科学和机器学习领域应用非常广泛，本书后续将深入讲解特征值分解。

11.6 奇异值分解：适用于任何实数矩阵

如果特征值分解是“大菜”，奇异值分解绝对就是矩阵分解中的“头牌”！

本书后续也会用两章内容专门讲解奇异值分解和应用。本书最后三章还会梳理特征值分解和奇异值分解之间关系，以及它们和数据、空间等概念的关系，把大家对特征值分解的认识提高一个全新高度。

本节则蜻蜓点水介绍一些奇异值分解最基本概念，并让大家感受一些手算奇异值分解的滋味。

定义

对矩阵 $\boldsymbol{X}_{n\times D}$ **奇异值分解** (Singular Value Decomposition, SVD)，得到：

$$\boldsymbol{X}_{n\times D} = \boldsymbol{U}\boldsymbol{S}\boldsymbol{V}^{\mathrm{T}} \tag{42}$$

$\boldsymbol{S}$ 主对角线元素 s_i 为**奇异值** (singular value)。一些教材用 $\boldsymbol{\Sigma}$ 代表奇异值矩阵，而丛书专门用 $\boldsymbol{\Sigma}$ 作为协方差矩阵记号。

$\boldsymbol{U}$ 的列向量称作**左奇异值向量** (left singular vector)。

$\boldsymbol{V}$ 的列向量称作**右奇异值向量** (left singular vector)。

一般情况，分解结果 $\boldsymbol{U}$ 和 $\boldsymbol{V}$ 为方阵，$\boldsymbol{S}$ 和 $\boldsymbol{X}$ 的形状相同，具体如图 16 所示。

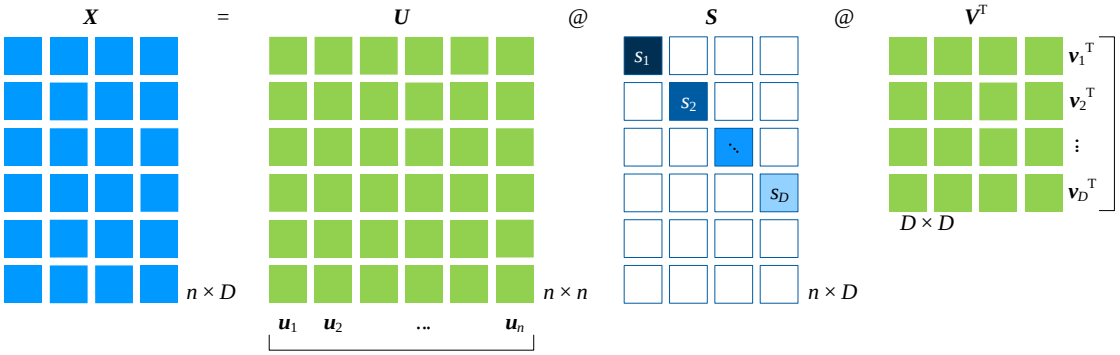


图 16. SVD 分解示意图

注意，任何实数矩阵都可以 SVD 分解。“任何”二字奠定了奇异值分解宇宙第一矩阵分解的地位！不管是方阵，细高、矮胖长方阵，对于 SVD 分解来说，兵来将挡水来土掩。

而 $\boldsymbol{U}$ 如果为方阵的话，它是正交矩阵。类似地，$\boldsymbol{V}$ 如果为方阵的话，它也是正交矩阵。这也就是说，几何视角下，$\boldsymbol{U}$ 和 $\boldsymbol{V}$ 都是规范正交基！如图 17 所示，这相当于一个 SVD 完成了图 15 中两个特征值分解。

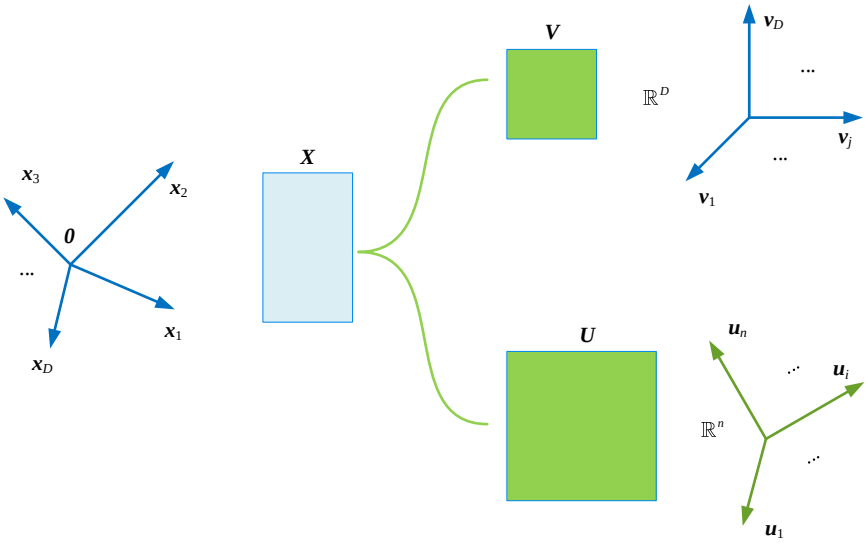


图 17. 对 $\boldsymbol{X}$ 矩阵 SVD 分解获得两个规范正交基

这说明，SVD 分解也是对原始数据矩阵进行正交化的工具，本章前文提到 QR 分解和特征值分解都可以得到规范正交基，这些矩阵分解之间的区别和联系是什么？得到的规范正交基有什么不同？它们和向量空间又有怎样关系？这是本书后续要回答的问题。

手算奇异值分解

这里举例说明如何徒手 SVD 求解，这绝不要求大家掌握。需要大家掌握的是 SVD 背后的思想，他和其他矩阵运算的联系，以及它的应用。

给定矩阵 $\boldsymbol{X}$:

$$\boldsymbol{X}=\begin{bmatrix}0 & 1\\ 1 & 1\\ 1 & 0\end{bmatrix} \tag{43}$$

为求解 $\boldsymbol{V}$，先计算 $\boldsymbol{X}$ 的转置和自身乘积 $\boldsymbol{X}^{\mathrm{T}}\boldsymbol{X}$:

$$\boldsymbol{X}^{\mathrm{T}}\boldsymbol{X}=\begin{bmatrix}0 & 1\\ 1 & 1\\ 1 & 0\end{bmatrix}^{\mathrm{T}}\begin{bmatrix}0 & 1\\ 1 & 1\\ 1 & 0\end{bmatrix}=\begin{bmatrix}2 & 1\\ 1 & 2\end{bmatrix} \tag{44}$$

相信大家已经在上式中看到了本书前文介绍的格拉姆 Gram 矩阵。本书前文说过，细高或矮胖的长方形矩阵在进行矩阵运算时并不友好，我们通常需要将它们“平方”，写成 $\boldsymbol{X}^{\mathrm{T}}\boldsymbol{X}$ 这种 Gram 矩阵的形式，以便完成进一步运算。

此外，本书前文还提到 Gram 矩阵的几何内涵，下一章大家可以看到 Cholesky 分解和 Gram 矩阵之间碰撞出的火花。

进一步计算得到 $\boldsymbol{X}^{\mathrm{T}}\boldsymbol{X}$ 特征值和特征向量:

$$\begin{cases}\lambda_1=3\\ \boldsymbol{v}_1=\begin{bmatrix}\frac{\sqrt{2}}{2}\\ \frac{\sqrt{2}}{2}\end{bmatrix}\end{cases}\quad\begin{cases}\lambda_2=1\\ \boldsymbol{v}_2=\begin{bmatrix}-\frac{\sqrt{2}}{2}\\ \frac{\sqrt{2}}{2}\end{bmatrix}\end{cases} \tag{45}$$

然后再求解 $\boldsymbol{X}\boldsymbol{X}^{\mathrm{T}}$:

$$\boldsymbol{X}\boldsymbol{X}^{\mathrm{T}}=\begin{bmatrix}0 & 1\\ 1 & 1\\ 1 & 0\end{bmatrix}\begin{bmatrix}0 & 1\\ 1 & 1\\ 1 & 0\end{bmatrix}^{\mathrm{T}}=\begin{bmatrix}1 & 1 & 0\\ 1 & 2 & 1\\ 0 & 1 & 1\end{bmatrix} \tag{46}$$

注意区分，$\boldsymbol{X}^{\mathrm{T}}\boldsymbol{X}$ 形状为 2×2，$\boldsymbol{X}\boldsymbol{X}^{\mathrm{T}}$ 形状为 3×3。两者都是 Gram 矩阵，前者关注 $\boldsymbol{X}$ 的列向量，后者关注 $\boldsymbol{X}$ 的行向量。

计算 $\boldsymbol{X}\boldsymbol{X}^{\mathrm{T}}$ 特征值和特征向量:

$$\begin{cases} \lambda_1 = 3 \\ \boldsymbol{u}_1 = \begin{bmatrix} \frac{1}{\sqrt{6}} \\ \frac{2}{\sqrt{6}} \\ \frac{1}{\sqrt{6}} \end{bmatrix} \end{cases} \quad \begin{cases} \lambda_2 = 1 \\ \boldsymbol{u}_2 = \begin{bmatrix} \frac{\sqrt{2}}{2} \\ 0 \\ -\frac{\sqrt{2}}{2} \end{bmatrix} \end{cases} \quad \begin{cases} \lambda_3 = 0 \\ \boldsymbol{u}_3 = \begin{bmatrix} \frac{\sqrt{3}}{3} \\ -\frac{\sqrt{3}}{3} \\ \frac{\sqrt{3}}{3} \end{bmatrix} \end{cases} \tag{47}$$

奇异值矩阵如下：

$$\boldsymbol{S} = \begin{bmatrix} s_1 & 0 \\ 0 & s_2 \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} \sqrt{\lambda_1} & 0 \\ 0 & \sqrt{\lambda_2} \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} \sqrt{3} & 0 \\ 0 & 1 \\ 0 & 0 \end{bmatrix} \tag{48}$$

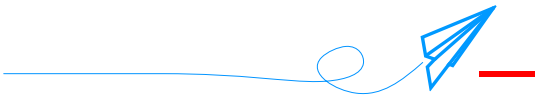
注意，对角线元素为 λ_1 和 λ_2 平方根。这一点是特征值分解和 SVD 分解的一个重要的区别，也是一个重要的联系。请大家格外留意。

因此，对 $\boldsymbol{X}$ 进行 SVD 分解可以得到：

$$\boldsymbol{X} = \boldsymbol{U}\boldsymbol{S}\boldsymbol{V}^{\mathrm{T}} = \begin{bmatrix} \frac{1}{\sqrt{6}} & \frac{\sqrt{2}}{2} & \frac{\sqrt{3}}{3} \\ \frac{2}{\sqrt{6}} & 0 & -\frac{\sqrt{3}}{3} \\ \frac{1}{\sqrt{6}} & -\frac{\sqrt{2}}{2} & \frac{\sqrt{3}}{3} \end{bmatrix} \begin{bmatrix} \sqrt{3} & 0 \\ 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} \\ \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \end{bmatrix}^{\mathrm{T}} \tag{49}$$

上式仅仅是 SVD 分解的一种形式，本书后续还将介绍 SVD 分解常见的四种形式。

再次强调，本书绝不要求大家掌握如何手算 SVD 分解，我们强调的是理解和应用。



本章开启了本书一个全新的板块——矩阵分解。以下四幅图总结本书的主要内容。矩阵分解看着让人眼花缭乱，但是万变不离其宗，矩阵分解的内核还是矩阵乘法！

矩阵分解让我们从一个全新的高度领略到了矩阵乘法的魅力。几何视角，数据视角，这两点绝对是学好矩阵分解的利器，怎么强调都不为过。

下面五章将展开讲解 Cholesky 分解、特征值分解和奇异值分解。本书最后三章会结合几何、数据、空间、应用等概念，再次升华矩阵分解！