5 矩阵乘法

Dive into Matrix Multiplication

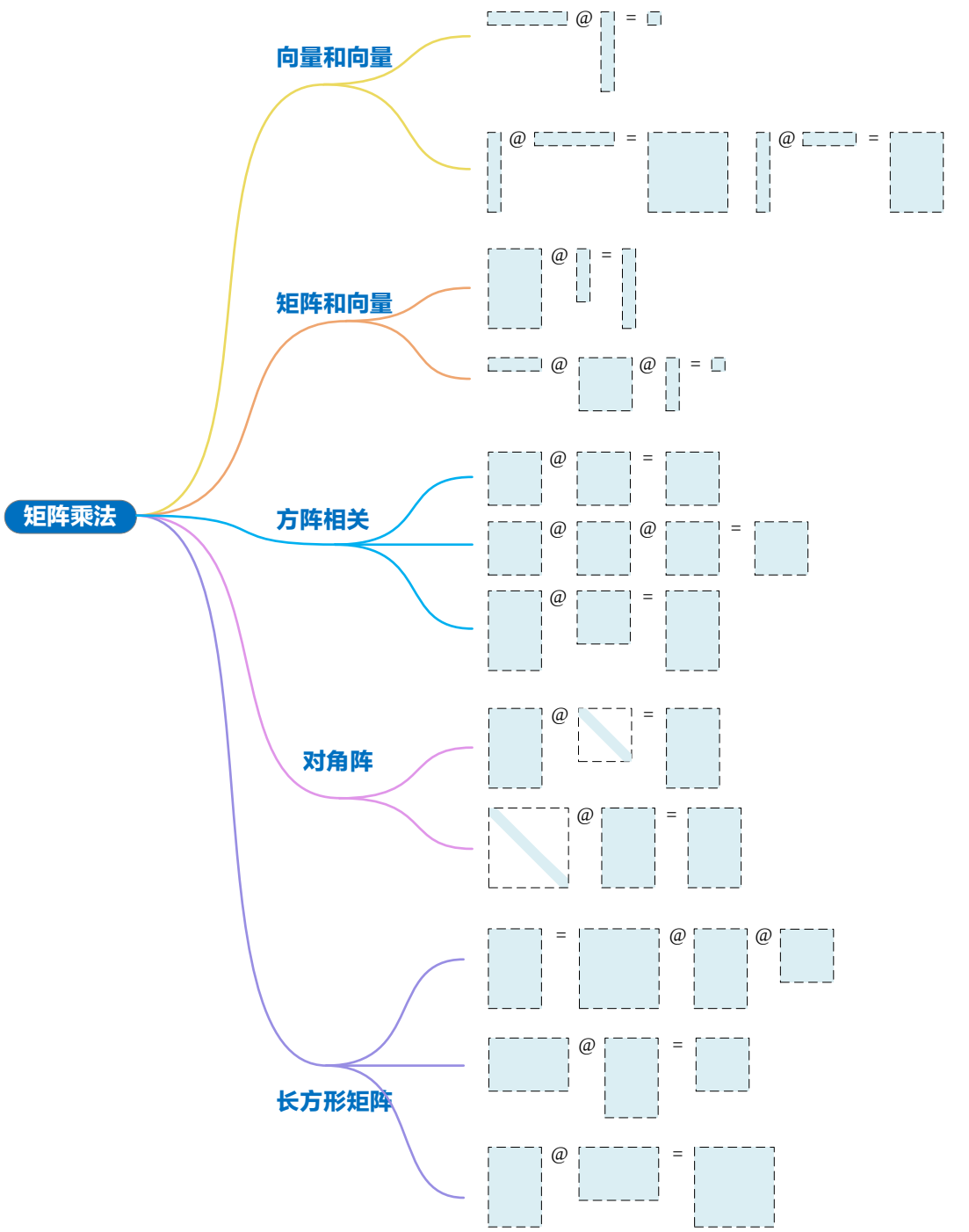
代数、几何、统计、数据交融的盛宴

只要持续进步，千万别泼冷水，哪怕蜗行牛步。

Never discourage anyone who continually makes progress, no matter how slow.

—— 柏拉图 (Plato) | 古希腊哲学家 | 424/423 ~ 348/347 BC

- `numpy.array()` 构造多维矩阵/数组
- `numpy.einsum()` 爱因斯坦求和约定
- `numpy.linalg.inv()` 矩阵逆运算
- `numpy.matrix()` 构造二维矩阵
- `numpy.multiply()` 矩阵逐项积
- `numpy.random.random_integers()` 生成随机整数
- `seaborn.heatmap()` 绘制数据热图



5.1 矩阵乘法：形态丰富多样

矩阵乘法可以说是矩阵运算中最重要的规则，没有之一！

矩阵乘法的规则不难理解；但是，横在大多数读者面前最大的困难是，矩阵乘法的灵活性。这种灵活性体现在矩阵乘法不同视角和矩阵乘法形态的多样性。

本书前文和大家讨论了矩阵乘法的两个视角，本书后续还将在分块矩阵中继续探讨矩阵乘法更多视角。

而本章将介绍常见矩阵乘法形态。阅读时，请大家注意从代数、几何、数据、统计几个角度理解，特别是几何和数据这两个角度。

如果你之前曾经系统学过线性代数，这一章会让你有众里寻他千百度相见恨晚的感觉。我在学习线性代数的时候，就特别希望能找到一本书能够把常见的矩阵乘法形态和应用场景都娓娓道来。

如果你刚刚接触线性代数相关内容，不要被本章大量术语吓到，大家现在不需要记住它们。本章可以视作全书重要知识点的总结。本章的作用就是鸟瞰全景，让大家开开眼界，不需要大家注意运算细节。希望读者学完本书后，再回过头来阅读本章，自己对矩阵乘法的认识绝对会进一步加深。

下面，我们就开始讲解各种形态的矩阵乘法。

5.2 向量和向量

定义两个等长列向量 $\boldsymbol{x}$ 和 $\boldsymbol{y}$：

$$\boldsymbol{x}=\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}, \quad \boldsymbol{y}=\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} \tag{1}$$

$\boldsymbol{x}$ 和 $\boldsymbol{y}$ 向量内积可以写作 $\boldsymbol{x}$ 转置乘 $\boldsymbol{y}$，或者 $\boldsymbol{y}$ 转置乘 $\boldsymbol{x}$：

$$\boldsymbol{x}\cdot\boldsymbol{y}=\boldsymbol{y}\cdot\boldsymbol{x}=\langle\boldsymbol{x},\boldsymbol{y}\rangle=\boldsymbol{x}^{\mathrm{T}}\boldsymbol{y}=\left(\boldsymbol{x}^{\mathrm{T}}\boldsymbol{y}\right)^{\mathrm{T}}=\boldsymbol{y}^{\mathrm{T}}\boldsymbol{x}=x_1y_1+x_2y_2+\cdots x_ny_n=\sum_{i=1}^{n}x_iy_i \tag{2}$$

(2) 告诉我们，$\boldsymbol{x}^{\mathrm{T}}\boldsymbol{y}$ 和 $\boldsymbol{y}^{\mathrm{T}}\boldsymbol{x}$ 相当于向量元素分别相乘，再求和。观察图 1，$\boldsymbol{x}^{\mathrm{T}}\boldsymbol{y}$ 和 $\boldsymbol{y}^{\mathrm{T}}\boldsymbol{x}$ 结果一致，均为标量，相当于 1×1 矩阵。

如果 $\boldsymbol{x}$ 和 $\boldsymbol{y}$ 正交，则两者向量内积为 0：

$$\boldsymbol{x}\cdot\boldsymbol{y}=\boldsymbol{y}\cdot\boldsymbol{x}=\langle\boldsymbol{x},\boldsymbol{y}\rangle=\boldsymbol{x}^{\mathrm{T}}\boldsymbol{y}=\left(\boldsymbol{x}^{\mathrm{T}}\boldsymbol{y}\right)^{\mathrm{T}}=\boldsymbol{y}^{\mathrm{T}}\boldsymbol{x}=0 \tag{3}$$

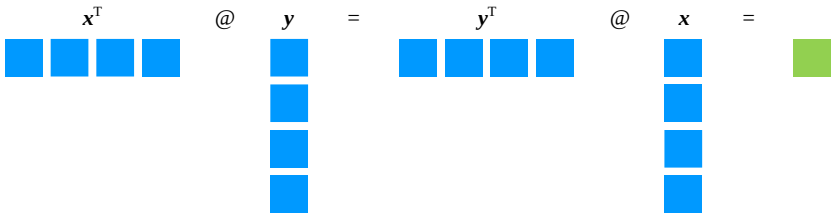


图 1. 标量积

全 1 列向量

全 1 列向量是非常神奇的存在，多元统计中离不开全 1 列向量。下面举两个例子。

如图 2 所示，全 1 列向量 $\boldsymbol{1}$ 乘行向量 $\boldsymbol{a}$，相当于对 $\boldsymbol{a}$ 进行复制，结果是向下叠放的向量 $\boldsymbol{a}$，即：

$$\boldsymbol{1}@\boldsymbol{a}=\begin{bmatrix}1\\1\\\vdots\\1\end{bmatrix}@\boldsymbol{a}=\begin{bmatrix}\boldsymbol{a}\\\boldsymbol{a}\\\vdots\\\boldsymbol{a}\end{bmatrix} \tag{4}$$

复制的份数取决于全 1 列向量 $\boldsymbol{1}$ 的元素个数。

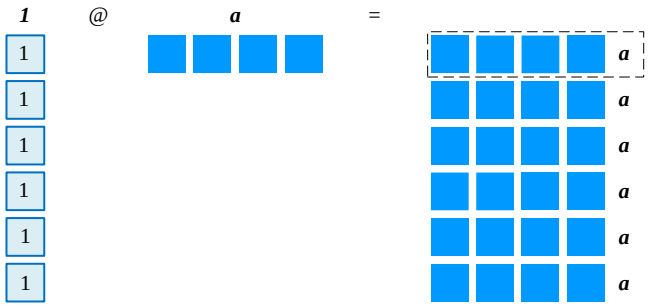


图 2. 复制行向量 $\boldsymbol{a}$

类似地，如所示，列向量 $\boldsymbol{b}$ 乘全 1 向量 $\boldsymbol{1}$ 转置，相当于对 $\boldsymbol{b}$ 进行复制，结果是左右排列的向量 $\boldsymbol{b}$

$$\boldsymbol{b}@\boldsymbol{1}^{\mathrm{T}}=\boldsymbol{b}@\begin{bmatrix}1\\1\\\vdots\\1\end{bmatrix}^{\mathrm{T}}=\begin{bmatrix}\boldsymbol{b} & \boldsymbol{b} & \cdots & \boldsymbol{b}\end{bmatrix} \tag{5}$$

本章后续大家会看到更多全 1 向量 $\boldsymbol{1}$ 相关例子。

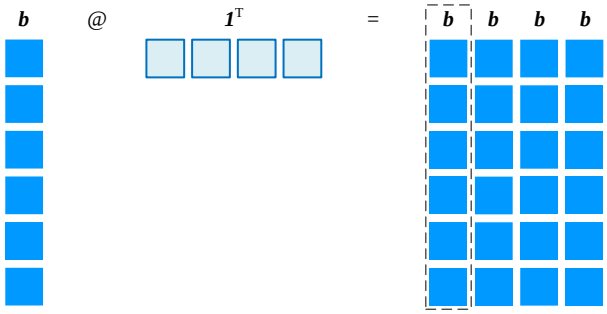


图 3. 复制列向量 $\boldsymbol{b}$

统计

下式为向量 $\boldsymbol{x}$ 元素求和：

$$\boldsymbol{1}\cdot\boldsymbol{x}=\boldsymbol{1}^{\mathrm{T}}\boldsymbol{x}=\boldsymbol{x}^{\mathrm{T}}\boldsymbol{1}=x_1+x_2+\cdots x_n=\sum_{i=1}^{n}x_i \tag{6}$$

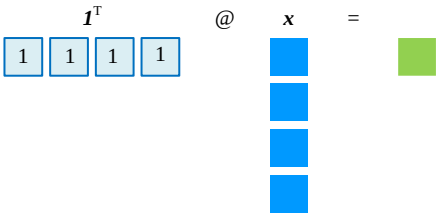


图 4. 求和运算

(6) 除以 n 便是向量 $\boldsymbol{x}$ 元素平均值：

$$\mathrm{E}\left(X\right)=\frac{x_1+x_2+\cdots x_n}{n}=\frac{1}{n}\sum_{i=1}^{n}x_i=\frac{\boldsymbol{1}\cdot\boldsymbol{x}}{n}=\frac{\boldsymbol{1}^{\mathrm{T}}\boldsymbol{x}}{n}=\frac{\boldsymbol{x}^{\mathrm{T}}\boldsymbol{1}}{n} \tag{7}$$

下式为向量 $\boldsymbol{x}$ 元素各自平方后再求和：

$$\boldsymbol{x}\cdot\boldsymbol{x}=\boldsymbol{x}^{\mathrm{T}}\boldsymbol{x}=x_1^2+x_2^2+\cdots x_n^2=\sum_{i=1}^{n}x_i^2 \tag{8}$$

如果 $\boldsymbol{x}$ 为单位向量，则下式成立：

$$\boldsymbol{x}\cdot\boldsymbol{x}=\boldsymbol{x}^{\mathrm{T}}\boldsymbol{x}=\left\|\boldsymbol{x}\right\|_2^2=1 \tag{9}$$

展开上式得到：

$$x_1^2 + x_2^2 + \cdots x_n^2 = 1 \tag{10}$$

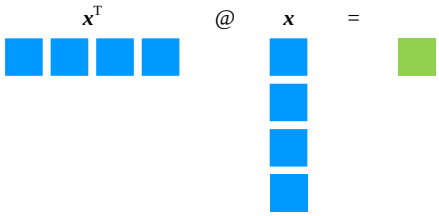


图 5. 平方和运算

计算方差时用到类似 (8) 计算：

$$\mathrm{var}(X) = \frac{1}{n}\sum_{i=1}^{n}(x_i - \mu)^2 \tag{11}$$

上式中，随机数 X 看成列向量 $\boldsymbol{x}$，

$$\mathrm{var}(X) = \frac{1}{n}\left(\boldsymbol{x} - \frac{\boldsymbol{1}^{\mathrm{T}}\boldsymbol{x}}{n}\right)\left(\boldsymbol{x} - \frac{\boldsymbol{1}^{\mathrm{T}}\boldsymbol{y}}{n}\right) \tag{12}$$

前文介绍过，在计算协方差时，我们采用类似 (2) 运算：

$$\mathrm{cov}(X,Y) = \frac{1}{n}\sum_{i=1}^{n}(x_i - \mathrm{E}(X))(y_i - \mathrm{E}(Y)) \tag{13}$$

注意，如果计算样本协方差的话，(13) 分母的 n 则应该改为 $n-1$。上式中，随机数 X 和 Y 可以看做两个列向量 $\boldsymbol{x}$ 和 $\boldsymbol{y}$，也就是说，(13) 可以写成：

$$\mathrm{cov}(X,Y) = \frac{1}{n}\left(\boldsymbol{x} - \frac{\boldsymbol{1}^{\mathrm{T}}\boldsymbol{x}}{n}\right)\cdot\left(\boldsymbol{y} - \frac{\boldsymbol{1}^{\mathrm{T}}\boldsymbol{y}}{n}\right) \tag{14}$$

张量积

列向量 $\boldsymbol{x}$ 和自身张量积得到一个方阵，相当于 $\boldsymbol{x}$ 和 $\boldsymbol{x}^{\mathrm{T}}$ 的乘积：

$$\boldsymbol{x} \otimes \boldsymbol{x} = \boldsymbol{x}\boldsymbol{x}^{\mathrm{T}} = \begin{bmatrix} x_1x_1 & x_1x_2 & \cdots & x_1x_n \\ x_2x_1 & x_2x_2 & \cdots & x_2x_n \\ \vdots & \vdots & \ddots & \vdots \\ x_nx_1 & x_nx_2 & \cdots & x_nx_n \end{bmatrix} \tag{15}$$

图 6 所示为 (15) 计算过程。

图 6. 张量积运算

另外，请读者注意如图 7 所示的两种形状的张量积和矩阵乘法关系。

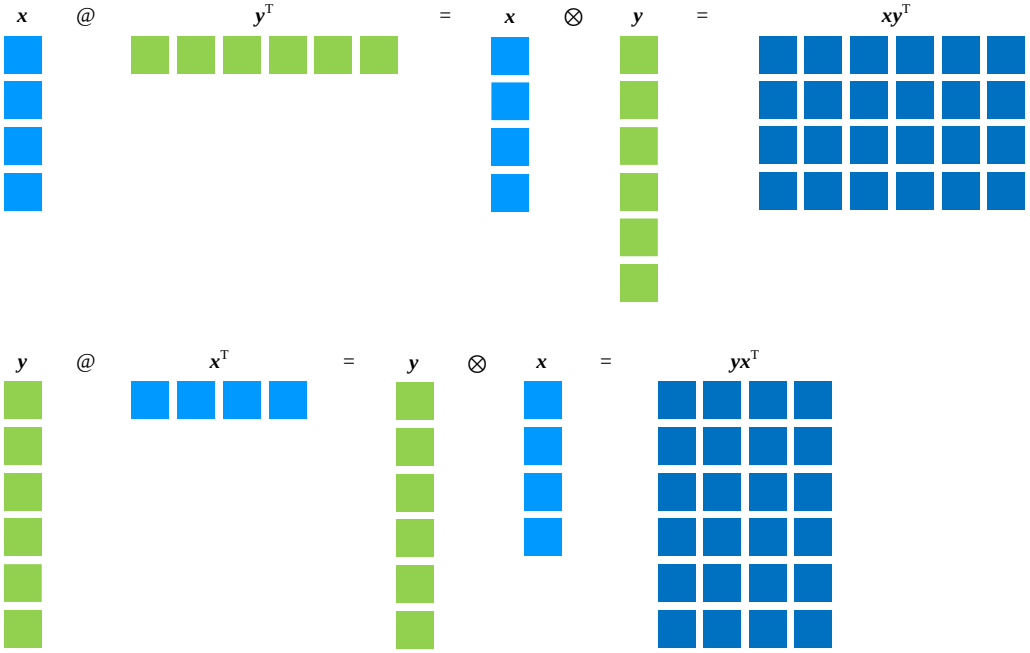


图 7. 另外两种形状的张量积

5.3 再聊全 1 列向量

本节主要介绍全 1 列向量在求和方面的作用。有关 Σ 求和，本系列丛书《数学要素》中讲过。本节主要从矩阵乘法角度再深入探讨。

每列元素求和

如图 8 所示，全 $\boldsymbol{1}$ 列向量转置左乘数据矩阵 $\boldsymbol{X}$，相当于对 $\boldsymbol{X}$ 每一列元素求和。计算结果为行向量，行向量的每个元素是 $\boldsymbol{X}$ 对应列元素之和：

$$\left(\mathbf{1}_{n\times 1}\right)^{\mathrm{T}}\boldsymbol{X}=\begin{bmatrix}1 & 1 & \cdots & 1\end{bmatrix}_{1\times n}\begin{bmatrix}x_{1,1} & x_{1,2} & \cdots & x_{1,D}\\ x_{2,1} & x_{2,2} & \cdots & x_{2,D}\\ \vdots & \vdots & \ddots & \vdots\\ x_{n,1} & x_{n,2} & \cdots & x_{n,D}\end{bmatrix}_{n\times D}=\begin{bmatrix}\sum_{i=1}^{n}x_{i,1} & \sum_{i=1}^{n}x_{i,2} & \cdots & \sum_{i=1}^{n}x_{i,D}\end{bmatrix}_{1\times D} \tag{16}$$

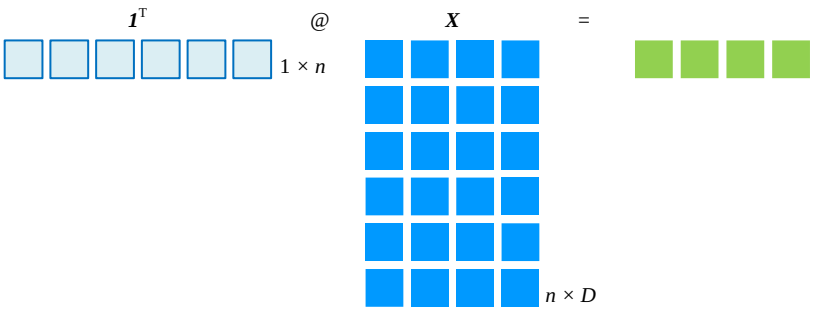


图 8. 列方向求和

(16) 左右除以 n，便得到每一列元素均值构成的行向量 E($\boldsymbol{X}$)：

$$\mathrm{E}\left(\boldsymbol{X}\right)=\frac{\mathbf{1}^{\mathrm{T}}\boldsymbol{X}}{n}=\begin{bmatrix}\frac{\sum_{i=1}^{n}x_{i,1}}{n} & \frac{\sum_{i=1}^{n}x_{i,2}}{n} & \cdots & \frac{\sum_{i=1}^{n}x_{i,D}}{n}\end{bmatrix}=\begin{bmatrix}\mu_1 & \mu_2 & \cdots & \mu_D\end{bmatrix} \tag{17}$$

E($\boldsymbol{X}$) 常被称作数据的质心 (centroid)。注意，本系列丛书中 E($\boldsymbol{X}$) 为行向量。而 $\boldsymbol{\mu_X}$ 为列向量，和 E($\boldsymbol{X}$) 就差在转置上，具体为：

$$\boldsymbol{\mu}_{\boldsymbol{X}}=\mathrm{E}\left(\boldsymbol{X}\right)^{\mathrm{T}}=\begin{bmatrix}\mu_1\\ \mu_2\\ \vdots\\ \mu_D\end{bmatrix}=\frac{\boldsymbol{X}^{\mathrm{T}}\mathbf{1}}{n} \tag{18}$$

E($\boldsymbol{X}$) 一般常配合原始数据矩阵 $\boldsymbol{X}$ 一起出现，而 $\boldsymbol{\mu_X}$ 多用在分布相关讨论中。

复制

上一节提到，全 1 列向量有复制的功能。

很多应用场合需要将 (17) 复制 n 份，得到一个和原矩阵形状相同的矩阵。下式可以完成这个计算：

$$\boldsymbol{1}@\mathrm{E}\left(\boldsymbol{X}\right)=\frac{\boldsymbol{1}\boldsymbol{1}^{\mathrm{T}}\boldsymbol{X}}{n}=\begin{bmatrix}\mu_1 & \mu_2 & \cdots & \mu_D\\ \mu_1 & \mu_2 & \cdots & \mu_D\\ \vdots & \vdots & \ddots & \vdots\\ \mu_1 & \mu_2 & \cdots & \mu_D\end{bmatrix}_{n\times D} \tag{19}$$

举个例子，对 $\boldsymbol{X}$ 去均值 (demean 或 centralize) 就是 $\boldsymbol{X}$ 的每个元素减去 $\boldsymbol{X}$ 对应列方向数据均值，可以通过下式得到去均值 $\boldsymbol{X}$:

$$\boldsymbol{X}-\frac{\boldsymbol{1}\boldsymbol{1}^{\mathrm{T}}\boldsymbol{X}}{n}=\begin{bmatrix}x_{1,1}-\mu_1 & x_{1,2}-\mu_2 & \cdots & x_{1,D}-\mu_D\\ x_{2,1}-\mu_1 & x_{2,2}-\mu_2 & \cdots & x_{2,D}-\mu_D\\ \vdots & \vdots & \ddots & \vdots\\ x_{n,1}-\mu_1 & x_{n,2}-\mu_2 & \cdots & x_{n,D}-\mu_D\end{bmatrix}_{n\times D} \tag{20}$$

上式可以整理为:

$$\boldsymbol{X}-\frac{\boldsymbol{1}\boldsymbol{1}^{\mathrm{T}}\boldsymbol{X}}{n}=\boldsymbol{I}\boldsymbol{X}-\frac{\boldsymbol{1}\boldsymbol{1}^{\mathrm{T}}\boldsymbol{X}}{n}=\left(\boldsymbol{I}-\frac{\boldsymbol{1}\boldsymbol{1}^{\mathrm{T}}}{n}\right)\boldsymbol{X} \tag{21}$$

其中，$\boldsymbol{I}$ 是单位矩阵，对角线元素都是 1，其余为 0。

如图 9 所示，从几何视角来看，去均值相当于将数据的质心平移到原点。

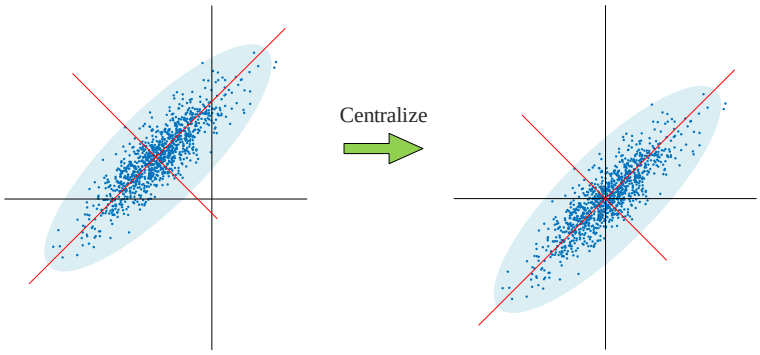


图 9. 去均值的几何视角

观察 (20)，可以发现式中有一个张量积——全 1 列向量的张量积 $\boldsymbol{1}\otimes\boldsymbol{1}$。因此，(20) 可以写成:

$$\boldsymbol{X}-\frac{\boldsymbol{1}\boldsymbol{1}^{\mathrm{T}}\boldsymbol{X}}{n}=\boldsymbol{X}-\frac{\boldsymbol{1}\otimes\boldsymbol{1}}{n}\boldsymbol{X} \tag{22}$$

张量积 $\boldsymbol{1}\otimes\boldsymbol{1}$ 是个 $n\times n$ 方阵，矩阵的元素都是 1。张量积 $\boldsymbol{1}\otimes\boldsymbol{1}$ 再除以 n 得到的方阵每个元素都是 $1/n$。

每行元素求和

如图 10 所示，据矩阵 $\boldsymbol{X}$ 乘全 1 列向量 $\boldsymbol{1}$，相当于对 $\boldsymbol{X}$ 每一行元素求和，结果为列向量：

$$\boldsymbol{X}_{n\times D}\boldsymbol{1}_{D\times 1}=\begin{bmatrix} x_{1,1} & x_{1,2} & \cdots & x_{1,D} \\ x_{2,1} & x_{2,2} & \cdots & x_{2,D} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n,1} & x_{n,2} & \cdots & x_{n,D} \end{bmatrix}_{n\times D}\begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix}_{D\times 1}=\begin{bmatrix} \sum_{j=1}^{D} x_{1,j} \\ \sum_{j=1}^{D} x_{2,j} \\ \vdots \\ \sum_{j=1}^{D} x_{n,j} \end{bmatrix}_{n\times 1} \tag{23}$$

注意，(17) 和 (23) 两式中的全 $\boldsymbol{1}$ 向量长度不同。

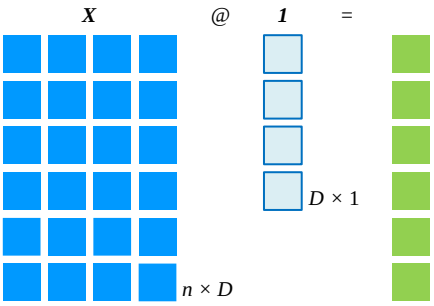


图 10. 行方向求和

所有元素的和

图 11 所示，数据矩阵 $\boldsymbol{X}$ 分别左乘 $\boldsymbol{1}^{\mathrm{T}}$、右乘全 $\boldsymbol{1}$ 向量，结果为 $\boldsymbol{X}$ 所有元素求和：

$$\boldsymbol{1}^{\mathrm{T}}\boldsymbol{X}\boldsymbol{1}=\begin{bmatrix} \sum_{i=1}^{n} x_{i,1} & \sum_{i=1}^{n} x_{i,2} & \cdots & \sum_{i=1}^{n} x_{i,D} \end{bmatrix}\begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix}=\sum_{j=1}^{D}\sum_{i=1}^{n} x_{i,j} \tag{24}$$

注意，上式中两个全 $\boldsymbol{1}$ 列向量长度也不同，具体形状如图 11 所示。

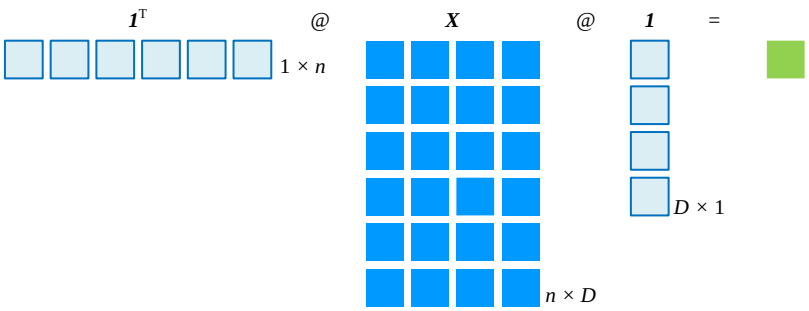


图 11. 矩阵所有元素求和

再强调一点，希望读者在看到一些规则的代数式时，要联想到对应的矩阵运算。本章后续还会继续给出更多示例，以便强化代数和矩阵运算的联系。

5.4 矩阵乘向量：线性方程组

矩阵 $\boldsymbol{A}$ 为 n 行、D 列：

$$\boldsymbol{A}_{n\times D}=\begin{bmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,D} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,D} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n,1} & a_{n,2} & \cdots & a_{n,D} \end{bmatrix} \tag{25}$$

$\boldsymbol{x}$ 为 D 个未知量 x_1、x_1 … x_D 构成的列向量，$\boldsymbol{b}$ 为常数 b_1、b_1 … b_n 构成的列向量：

$$\boldsymbol{x}_{D\times 1}=\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_D \end{bmatrix},\quad \boldsymbol{b}_{n\times 1}=\begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix} \tag{26}$$

$\boldsymbol{Ax}=\boldsymbol{b}$ 可以写成：

$$\underbrace{\begin{bmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,D} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,D} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n,1} & a_{n,2} & \cdots & a_{n,D} \end{bmatrix}}_{\boldsymbol{A}_{n\times D}}\underbrace{\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_D \end{bmatrix}}_{\boldsymbol{x}_{D\times 1}}=\underbrace{\begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix}}_{\boldsymbol{b}_{n\times 1}} \tag{27}$$

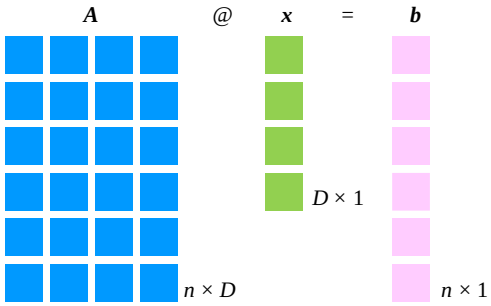


图 12. 长方阵乘列向量

(27) 展开得到**线性方程组** (system of linear equations)：

$$\begin{cases} a_{1,1}x_1 + a_{1,2}x_2 + \cdots + a_{1,D}x_D = b_1 \\ a_{2,1}x_1 + a_{2,2}x_2 + \cdots + a_{2,D}x_D = b_2 \\ \vdots \\ a_{n,1}x_1 + a_{n,2}x_2 + \cdots + a_{n,D}x_D = b_n \end{cases} \tag{28}$$

举个例子，本系列丛书《数学要素》一册中鸡兔同笼问题，就可以写成：

$$\begin{bmatrix} 1 & 1 \\ 2 & 4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 35 \\ 94 \end{bmatrix} \quad \Rightarrow \quad \begin{cases} x_1 + x_2 = 35 \\ 2x_1 + 4x_2 = 94 \end{cases} \tag{29}$$

解的个数

上式有唯一一组解，有唯一解方程组被称作恰定方程组。

有无穷多解的**欠定方程组** (underdetermined system)。

解不存在的方程组被称作**超定方程组** (overdetermined system)。

本系列丛书《数学要素》一册介绍过**最小二乘法** (ordinary least squares, OLS) 和超定方程组之间的关系。图 12 也是线性回归问题的矩阵运算形式。

拿来《数学要素》的一组图，用二元一次方程组解释解的三种情况。

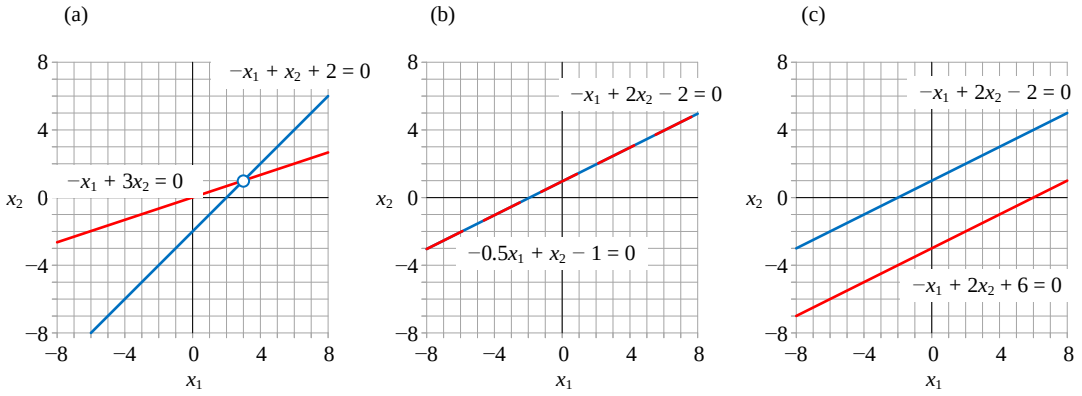


图 13. 两个二元一次方程组有一个解、无数解、没有解

图 13 (a) 给出的两条直线相交于一点，也就是二元一次方程组有一个解。写成线性方程组：

$$\begin{bmatrix} -1 & 1 \\ -1 & 3 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} -2 \\ 0 \end{bmatrix} \tag{30}$$

图 13 (b) 给出的两条直线相重合，也就是二元一次方程组无数解，写成线性方程组：

$$\underbrace{\begin{bmatrix} -1 & 2 \\ -0.5 & 1 \end{bmatrix}}_{\boldsymbol{A}} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 2 \\ 1 \end{bmatrix} \tag{31}$$

图 13 (c) 给出的两条直线平行，也就是二元一次方程组没有解。写成线性方程组：

$$\underbrace{\begin{bmatrix} -1 & 2 \\ -1 & 2 \end{bmatrix}}_{\boldsymbol{A}} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 2 \\ -6 \end{bmatrix} \tag{32}$$

线性组合视角

下面用另外一个视角看 $\boldsymbol{Ax} = \boldsymbol{b}$。

本书前文反复提到，矩阵 $\boldsymbol{A}$ 可以看做由一排列向量构造而成：

$$\boldsymbol{A}_{n\times D} = \begin{bmatrix} \boldsymbol{a}_1 & \boldsymbol{a}_2 & \cdots & \boldsymbol{a}_D \end{bmatrix} \tag{33}$$

(27) 可以写成：

$$\begin{bmatrix} \boldsymbol{a}_1 & \boldsymbol{a}_2 & \cdots & \boldsymbol{a}_D \end{bmatrix}_{1\times D} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_D \end{bmatrix}_{D\times 1} = \boldsymbol{b}_{n\times 1} \tag{34}$$

展开得到：

$$x_1\boldsymbol{a}_1 + x_2\boldsymbol{a}_2 + \cdots + x_D\boldsymbol{a}_D = \boldsymbol{b}_{n\times 1} \tag{35}$$

即，

$$x_1 \underbrace{\begin{bmatrix} a_{1,1} \\ a_{2,1} \\ \vdots \\ a_{n,1} \end{bmatrix}}_{\boldsymbol{a}_1} + x_2 \underbrace{\begin{bmatrix} a_{1,2} \\ a_{2,2} \\ \vdots \\ a_{n,2} \end{bmatrix}}_{\boldsymbol{a}_2} + \cdots + x_D \underbrace{\begin{bmatrix} a_{1,D} \\ a_{2,D} \\ \vdots \\ a_{n,D} \end{bmatrix}}_{\boldsymbol{a}_D} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix} \tag{36}$$

当 x_1、$x_1 \ldots x_D$ 取具体值时，上式是**线性组合** (linear combination)。用腊八粥举个例子，上式相当于不同比例的原料混合，x_i 就是比例，$\boldsymbol{a}_i$ 就是不同的原料。线性组合这个概念非常重要，本书后续将专门介绍。

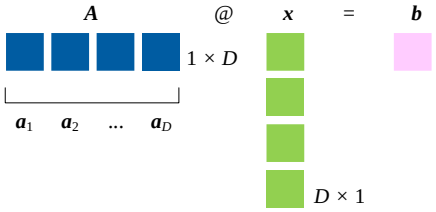


图 14. 线性组合视角看线性方程组

5.5 向量乘矩阵乘向量：二次型

二次型 (quadratic form) 的矩阵算式为：

$$\boldsymbol{x}^{\mathrm{T}}\boldsymbol{Q}\boldsymbol{x}=q \tag{37}$$

其中，$\boldsymbol{Q}$ 为对称阵，q 为实数。矩阵运算过程如图 15 所示。

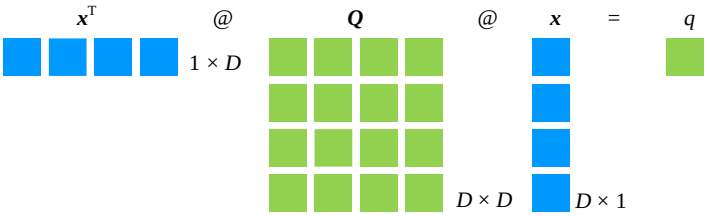


图 15. $\boldsymbol{x}^{\mathrm{T}}\boldsymbol{Q}\boldsymbol{x}=q$ 矩阵运算

$\boldsymbol{Q}$ 和 $\boldsymbol{x}$ 分别为：

$$\boldsymbol{x}=\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_D \end{bmatrix}, \quad \boldsymbol{Q}=\begin{bmatrix} q_{1,1} & q_{1,2} & \cdots & q_{1,D} \\ q_{2,1} & q_{2,2} & \cdots & q_{2,D} \\ \vdots & \vdots & \ddots & \vdots \\ q_{D,1} & q_{D,2} & \cdots & q_{D,D} \end{bmatrix} \tag{38}$$

将 (38) 代入 (37)，展开得到：

$$\boldsymbol{x}^{\mathrm{T}}\boldsymbol{Q}\boldsymbol{x}=\sum_{i=1}^{D}q_{i,i}x_i^2+\sum_{i=1}^{D}\sum_{j=1}^{D}q_{i,j}x_ix_j=q \tag{39}$$

举个例子

举个简单例子，比如 $\boldsymbol{x}$ 和 $\boldsymbol{Q}$ 分别为：

$$\boldsymbol{x}=\begin{bmatrix}x_1\\x_2\end{bmatrix},\quad \boldsymbol{Q}=\begin{bmatrix}a & b\\c & d\end{bmatrix} \tag{40}$$

代入 (37) 得到：

$$\begin{bmatrix}x_1 & x_2\end{bmatrix}\begin{bmatrix}a & b\\c & d\end{bmatrix}\begin{bmatrix}x_1\\x_2\end{bmatrix}=ax_1^2+(b+c)x_1x_2+dx_2^2=q \tag{41}$$

可以发现，(41) 对应本系列丛书中《数学要素》介绍过各种二次曲线，比如正圆、椭圆、抛物线或双曲线，如图 16 所示。

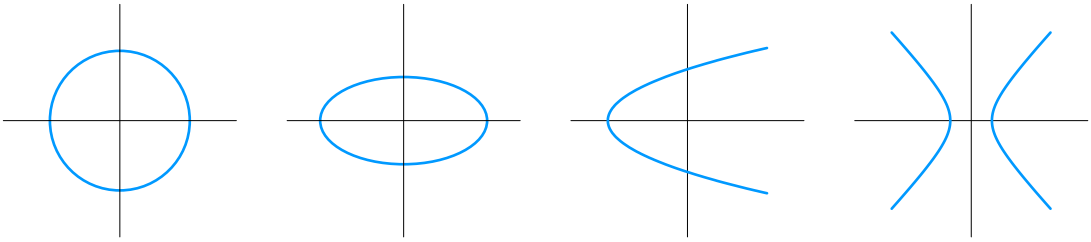


图 16. 四种二次曲线

将 (41) 写成二元函数形式 $f(x_1, x_2)$：

$$f(x_1,x_2)=\begin{bmatrix}x_1 & x_2\end{bmatrix}\begin{bmatrix}a & b\\c & d\end{bmatrix}\begin{bmatrix}x_1\\x_2\end{bmatrix}=ax_1^2+(b+c)x_1x_2+dx_2^2 \tag{42}$$

$f(x_1, x_2) = q$，相当于曲面某个高度的等高线。上式便是我们常见的二次型的示例。(42) 对应着如图 17 所示的几种曲面。

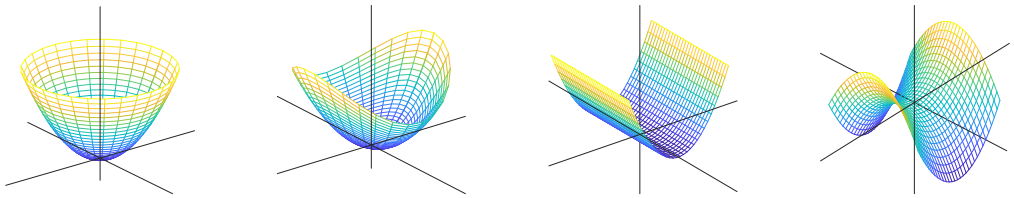


图 17. 常见二次型曲面

二次型在多元微积分、正定性、多元正态分布、协方差矩阵、数据映射和优化方法中都有举足轻重的分量。本书后续将会深入探讨。

5.6 方阵乘方阵：矩阵分解

和方阵有关的矩阵乘法中，方阵乘方阵最为简单。

图 18 所示这种方阵乘法广泛应用于 LU 分解、Cholesky 分解、特征值分解等场合。本节不做展开介绍，本书后续会专门介绍不同类别矩阵分解。

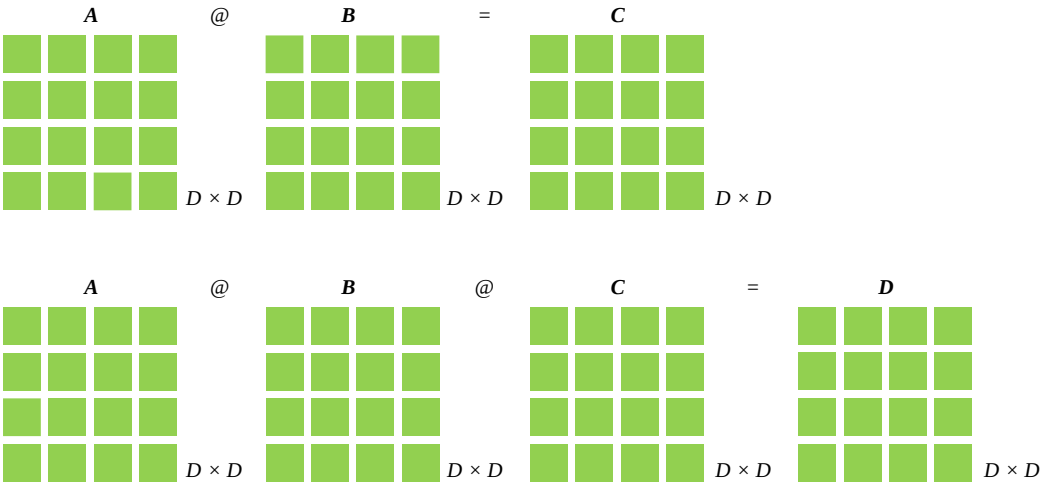


图 18. 方阵乘方阵

特别地，如果方阵 $\boldsymbol{A}$，下式成立：

$$\boldsymbol{A}^2 = \boldsymbol{A} \tag{43}$$

则称 $\boldsymbol{A}$ 为**幂等矩阵** (idempotent matrix)。我们会在本书统计部分和最小二乘法线性回归中再谈及幂等矩阵。

丛书每册均有涉及线性回归这个话题，本书采用的是线性代数和向量几何视角，《概率统计》使用统计分析视角，而《数据科学》则是从数据分析视角介绍线性回归。

5.7 对角阵：批量缩放

如果形状相同的方阵 $\boldsymbol{A}$ 和 $\boldsymbol{B}$ 都为对角阵，两者乘积还是一个对角阵：

$$\boldsymbol{A}_{D\times D}\boldsymbol{B}_{D\times D} = \begin{bmatrix} a_1 & & & \\ & a_2 & & \\ & & \ddots & \\ & & & a_D \end{bmatrix} \begin{bmatrix} b_1 & & & \\ & b_2 & & \\ & & \ddots & \\ & & & b_D \end{bmatrix} = \begin{bmatrix} a_1 b_1 & & & \\ & a_2 b_2 & & \\ & & \ddots & \\ & & & a_D b_D \end{bmatrix} \tag{44}$$

对角阵 $\boldsymbol{\Lambda}$ 的逆也是一个对角阵：

$$\boldsymbol{\Lambda}_{D\times D}\left(\boldsymbol{\Lambda}_{D\times D}\right)^{-1}=\begin{bmatrix}\lambda_1 & & & \\ & \lambda_2 & & \\ & & \ddots & \\ & & & \lambda_D\end{bmatrix}\begin{bmatrix}1/\lambda_1 & & & \\ & 1/\lambda_2 & & \\ & & \ddots & \\ & & & 1/\lambda_D\end{bmatrix}=\begin{bmatrix}1 & & & \\ & 1 & & \\ & & \ddots & \\ & & & 1\end{bmatrix}=\boldsymbol{I}_{D\times D} \tag{45}$$

右乘

矩阵 $\boldsymbol{A}$ 乘对角方阵阵 $\boldsymbol{\Lambda}$：

$$\begin{aligned}\boldsymbol{A}_{n\times D}\boldsymbol{\Lambda}_{D\times D}&=\begin{bmatrix}\boldsymbol{a}_1 & \boldsymbol{a}_2 & \cdots & \boldsymbol{a}_D\end{bmatrix}\begin{bmatrix}\lambda_1 & 0 & \cdots & 0\\ 0 & \lambda_2 & \cdots & 0\\ \vdots & \vdots & \ddots & \vdots\\ 0 & 0 & \cdots & \lambda_D\end{bmatrix}\\&=\begin{bmatrix}\lambda_1\boldsymbol{a}_1 & \lambda_2\boldsymbol{a}_2 & \cdots & \lambda_D\boldsymbol{a}_D\end{bmatrix}\end{aligned} \tag{46}$$

观察 (46) 发现，$\boldsymbol{\Lambda}$ 的对角线元素相当于缩放系数，分别对矩阵 $\boldsymbol{A}$ 的每一列数值进行不同比例缩放。

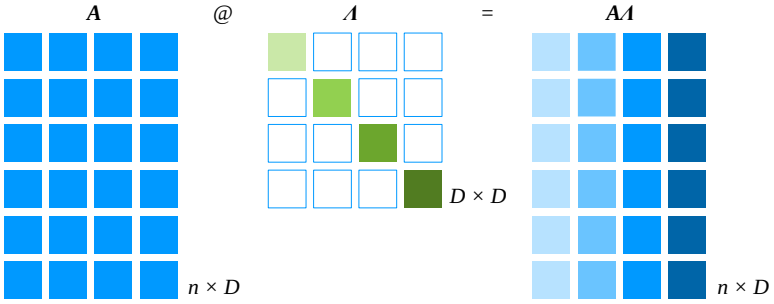


图 19. $\boldsymbol{A}$ 乘对角方阵 $\boldsymbol{\Lambda}$

左乘

对角阵 $\boldsymbol{\Lambda}$ 左乘矩阵 $\boldsymbol{A}$：

$$\boldsymbol{\Lambda}_{n\times n}\boldsymbol{A}_{n\times D}=\begin{bmatrix}\lambda_1 & 0 & \cdots & 0\\ 0 & \lambda_2 & \cdots & 0\\ \vdots & \vdots & \ddots & \vdots\\ 0 & 0 & \cdots & \lambda_n\end{bmatrix}_{n\times n}\begin{bmatrix}\boldsymbol{a}^{(1)}\\ \boldsymbol{a}^{(2)}\\ \vdots\\ \boldsymbol{a}^{(n)}\end{bmatrix}_{n\times 1}=\begin{bmatrix}\lambda_1\boldsymbol{a}^{(1)}\\ \lambda_2\boldsymbol{a}^{(2)}\\ \vdots\\ \lambda_n\boldsymbol{a}^{(n)}\end{bmatrix}_{n\times 1} \tag{47}$$

观察 (47)，可以发现 $\boldsymbol{\Lambda}$ 的对角线元素分别对矩阵 $\boldsymbol{A}$ 的每一行数值进行批量缩放。

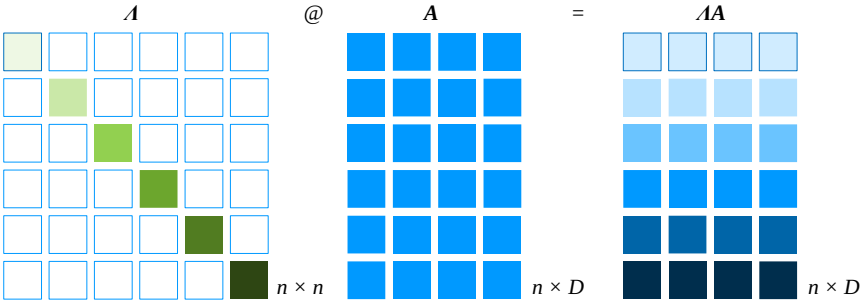


图 20. 对角阵 $\boldsymbol{\Lambda}$ 乘方阵 $\boldsymbol{A}$

乘列向量

特别地，如果对角阵 $\boldsymbol{\Lambda}$ 乘列向量 $\boldsymbol{x}$，相当于对列向量每个元素以不同比例分别缩放：

$$\boldsymbol{\Lambda}_{D\times D}\boldsymbol{x}=\begin{bmatrix}\lambda_1 & 0 & \cdots & 0\\ 0 & \lambda_2 & \cdots & 0\\ \vdots & \vdots & \ddots & \vdots\\ 0 & 0 & \cdots & \lambda_D\end{bmatrix}_{D\times D}\begin{bmatrix}x_1\\ x_2\\ \vdots\\ x_D\end{bmatrix}_{D\times 1}=\begin{bmatrix}\lambda_1 x_1\\ \lambda_2 x_2\\ \vdots\\ \lambda_D x_D\end{bmatrix}_{D\times 1} \tag{48}$$

乘行向量

类似地，如果行向量 $\boldsymbol{x}^{(1)}$ 乘对角阵 $\boldsymbol{\Lambda}$，相当于对行向量每个元素以不同比例分别缩放：

$$\boldsymbol{x}^{(1)}\boldsymbol{\Lambda}_{D\times D}=\begin{bmatrix}x_{1,1} & x_{1,2} & \cdots & x_{1,D}\end{bmatrix}\begin{bmatrix}\lambda_1 & 0 & \cdots & 0\\ 0 & \lambda_2 & \cdots & 0\\ \vdots & \vdots & \ddots & \vdots\\ 0 & 0 & \cdots & \lambda_D\end{bmatrix}_{D\times D}=\begin{bmatrix}\lambda_1 x_{1,1} & \lambda_2 x_{1,2} & \cdots & \lambda_D x_{1,D}\end{bmatrix} \tag{49}$$

左右都乘

再看下例，对角阵 $\boldsymbol{\Lambda}$ 分别左乘、右乘方阵 $\boldsymbol{B}$：

$$\begin{aligned}\boldsymbol{\Lambda}\boldsymbol{B}\boldsymbol{\Lambda}&=\begin{bmatrix}\lambda_1 & 0 & \cdots & 0\\ 0 & \lambda_2 & \cdots & 0\\ \vdots & \vdots & \ddots & \vdots\\ 0 & 0 & \cdots & \lambda_D\end{bmatrix}\begin{bmatrix}b_{1,1} & b_{1,2} & \cdots & b_{1,D}\\ b_{2,1} & b_{2,2} & \cdots & b_{2,D}\\ \vdots & \vdots & \ddots & \vdots\\ b_{D,1} & b_{D,2} & \cdots & b_{D,D}\end{bmatrix}\begin{bmatrix}\lambda_1 & 0 & \cdots & 0\\ 0 & \lambda_2 & \cdots & 0\\ \vdots & \vdots & \ddots & \vdots\\ 0 & 0 & \cdots & \lambda_D\end{bmatrix}\\ &=\begin{bmatrix}\lambda_1\lambda_1 b_{1,1} & \lambda_1\lambda_2 b_{1,2} & \cdots & \lambda_1\lambda_D b_{1,D}\\ \lambda_2\lambda_1 b_{2,1} & \lambda_2\lambda_2 b_{2,2} & \cdots & \lambda_2\lambda_D b_{2,D}\\ \vdots & \vdots & \ddots & \vdots\\ \lambda_D\lambda_1 b_{D,1} & \lambda_D\lambda_2 b_{D,2} & \cdots & \lambda_D\lambda_D b_{D,D}\end{bmatrix}\end{aligned} \tag{50}$$

看到 (50) 结果形式，大家是否想到了协方差矩阵。λ_i 相当于均方差，$b_{i,j}$ 相当于相关性系数。

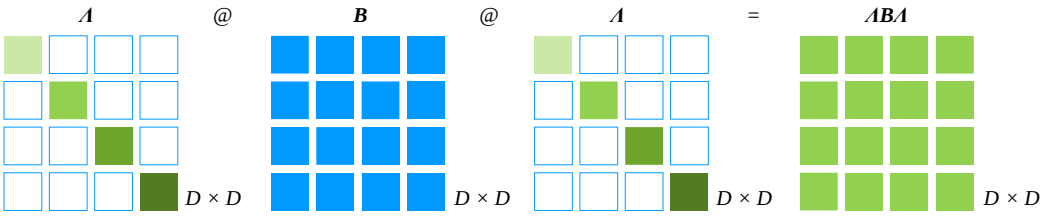


图 21. 对角阵 $\boldsymbol{\Lambda}$ 分别左乘、右乘方阵 $\boldsymbol{B}$

二次型特例

再看一个例子：

$$\boldsymbol{x}^{\mathrm{T}}\boldsymbol{\Lambda}_{D\times D}\boldsymbol{x}=\begin{bmatrix}x_1\\x_2\\\vdots\\x_D\end{bmatrix}^{\mathrm{T}}\begin{bmatrix}\lambda_1&0&\cdots&0\\0&\lambda_2&\cdots&0\\\vdots&\vdots&\ddots&\vdots\\0&0&\cdots&\lambda_D\end{bmatrix}\begin{bmatrix}x_1\\x_2\\\vdots\\x_D\end{bmatrix}=\lambda_1x_1^2+\lambda_2x_2^2+\cdots+\lambda_Dx_D^2=\sum_{j=1}^{D}\lambda_jx_j^2 \tag{51}$$

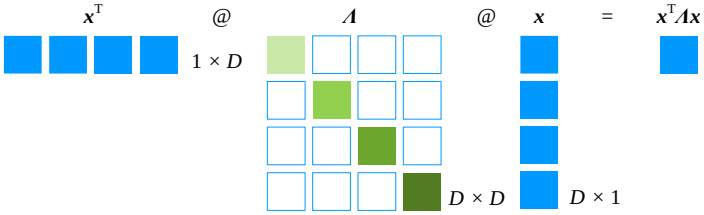


图 22. $\boldsymbol{x}^{\mathrm{T}}\boldsymbol{\Lambda}\boldsymbol{x}$ 对应的矩阵运算

上式实际上是本章前文介绍的二次型的特例。看到类似 (51) 形式运算，希望大家能联想到正椭圆、正椭球、正椭圆抛物面，比如：

$$\begin{bmatrix}x_1\\x_2\end{bmatrix}^{\mathrm{T}}\begin{bmatrix}1/4&\\&1\end{bmatrix}\begin{bmatrix}x_1\\x_2\end{bmatrix}=\frac{1}{4}x_1^2+x_2^2=1 \tag{52}$$

特别地，当对角阵对角线元素数值相同，我们得到的是正圆、正球体，比如：

$$\begin{bmatrix}x_1\\x_2\end{bmatrix}^{\mathrm{T}}\begin{bmatrix}2&\\&2\end{bmatrix}\begin{bmatrix}x_1\\x_2\end{bmatrix}=2x_1^2+2x_2^2=1 \tag{53}$$

几何视角

再次强调，如果在矩阵运算时遇到对角阵，请试着从几何缩放角度来看。比如图 23 所示的，单位圆经过缩放得到椭圆，这个过程便可以用对角阵进行变换。几何视角是本书后续要介绍的重要内容之一。

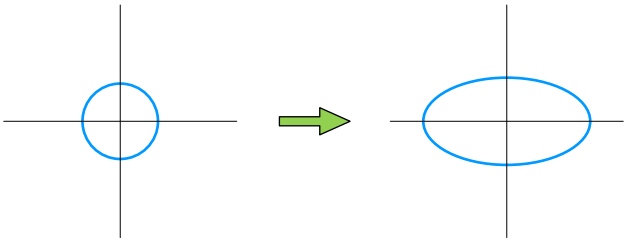


图 23. 单位圆到椭圆的缩放过程

5.8 置换矩阵：调换元素顺序

行向量 $\boldsymbol{a}$ 乘以副对角矩阵，副对角线上元素都为 1，得到左右翻转的行向量：

$$\begin{bmatrix} a_1 & a_2 & \cdots & a_D \end{bmatrix}_{1\times D} \begin{bmatrix} & & & 1 \\ & & \iddots & \\ & 1 & & \\ 1 & & & \end{bmatrix}_{D\times D} = \begin{bmatrix} a_D & a_{D-1} & \cdots & a_1 \end{bmatrix} \tag{54}$$

实际上，(54) 中完成左右翻转的方阵是**置换矩阵** (permutation matrix) 的一种特殊形式。

置换矩阵是由 0 和 1 组成的方阵；置换矩阵的每一行每一列都恰好只有一个 1，其余元素均为 0。置换矩阵的作用是调换元素顺序。

举个例子：

$$\begin{bmatrix} a_1 & a_2 & a_3 & a_4 \end{bmatrix} \begin{bmatrix} & 1 & & \\ & & & 1 \\ 1 & & & \\ & & 1 & \end{bmatrix} = \begin{bmatrix} a_3 & a_1 & a_4 & a_2 \end{bmatrix} \tag{55}$$

调整列向量顺序

置换矩阵同样可以作用于矩阵，将 (55) 中行向量元素替换成列向量，即，

$$\boldsymbol{a}_1 = \begin{bmatrix} a_{1,1} \\ a_{2,1} \\ a_{3,1} \\ a_{4,1} \end{bmatrix}, \quad \boldsymbol{a}_2 = \begin{bmatrix} a_{1,2} \\ a_{2,2} \\ a_{3,2} \\ a_{4,2} \end{bmatrix}, \quad \boldsymbol{a}_3 = \begin{bmatrix} a_{1,3} \\ a_{2,3} \\ a_{3,3} \\ a_{4,3} \end{bmatrix}, \quad \boldsymbol{a}_4 = \begin{bmatrix} a_{1,4} \\ a_{2,4} \\ a_{3,4} \\ a_{4,4} \end{bmatrix} \tag{56}$$

可以得到：

$$\begin{bmatrix} \begin{bmatrix} a_{1,1} \\ a_{2,1} \\ a_{3,1} \\ a_{4,1} \end{bmatrix} & \begin{bmatrix} a_{1,2} \\ a_{2,2} \\ a_{3,2} \\ a_{4,2} \end{bmatrix} & \begin{bmatrix} a_{1,3} \\ a_{2,3} \\ a_{3,3} \\ a_{4,3} \end{bmatrix} & \begin{bmatrix} a_{1,4} \\ a_{2,4} \\ a_{3,4} \\ a_{4,4} \end{bmatrix} \end{bmatrix} \begin{bmatrix} & 1 & & \\ & & & 1 \\ 1 & & & \\ & & 1 & \end{bmatrix} = \begin{bmatrix} \begin{bmatrix} a_{1,3} \\ a_{2,3} \\ a_{3,3} \\ a_{4,3} \end{bmatrix} & \begin{bmatrix} a_{1,1} \\ a_{2,1} \\ a_{3,1} \\ a_{4,1} \end{bmatrix} & \begin{bmatrix} a_{1,4} \\ a_{2,4} \\ a_{3,4} \\ a_{4,4} \end{bmatrix} & \begin{bmatrix} a_{1,2} \\ a_{2,2} \\ a_{3,2} \\ a_{4,2} \end{bmatrix} \end{bmatrix} \tag{57}$$

大家看到置换矩阵右乘矩阵 $\boldsymbol{A}$，让 $\boldsymbol{A}$ 的列向量顺序改变。

调整行向量顺序

这个置换矩阵左乘矩阵 $\boldsymbol{A}$，可以改变 $\boldsymbol{A}$ 的行向量的排序：

$$\begin{bmatrix} & 1 & & \\ & & & 1 \\ 1 & & & \\ & & 1 & \end{bmatrix} \begin{bmatrix} \boldsymbol{a}^{(1)} \\ \boldsymbol{a}^{(2)} \\ \boldsymbol{a}^{(3)} \\ \boldsymbol{a}^{(4)} \end{bmatrix} = \begin{bmatrix} \boldsymbol{a}^{(2)} \\ \boldsymbol{a}^{(4)} \\ \boldsymbol{a}^{(1)} \\ \boldsymbol{a}^{(3)} \end{bmatrix} \tag{58}$$

置换矩阵可以用来简化一些矩阵运算。

5.9 矩阵乘向量：映射到一维

$n \times D$ 形状的矩阵 $\boldsymbol{X}$ 乘 $D \times 1$ 列向量 $\boldsymbol{v}$ 得到 $n \times 1$ 列向量 $\boldsymbol{z}$：

$$\boldsymbol{X}_{n \times D} \boldsymbol{v}_{D \times 1} = \boldsymbol{z}_{n \times 1} \tag{59}$$

如图 24 所示，矩阵 $\boldsymbol{X}$ 有 D 列，对应 D 个特征。而结果 $\boldsymbol{z}$ 只有一列，也就是一个特征。

(59) 可以称作为**映射** (mapping)，这个运算过程相当于降维，如图 25 所示。

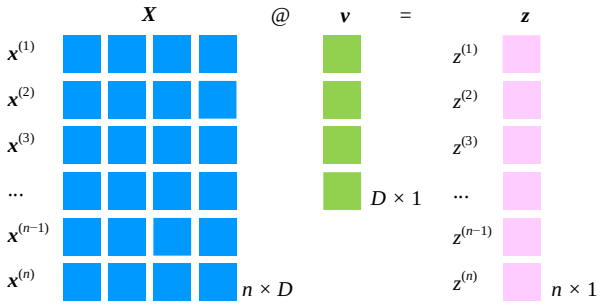


图 24. 矩阵乘法 $\boldsymbol{Xv} = \boldsymbol{z}$

$\boldsymbol{X}\boldsymbol{v} = \boldsymbol{z}$ 可以展开写成：

$$\boldsymbol{X}\boldsymbol{v} = \begin{bmatrix} \boldsymbol{x}^{(1)} \\ \boldsymbol{x}^{(2)} \\ \vdots \\ \boldsymbol{x}^{(n)} \end{bmatrix} \boldsymbol{v} = \begin{bmatrix} \boldsymbol{x}^{(1)}\boldsymbol{v} \\ \boldsymbol{x}^{(2)}\boldsymbol{v} \\ \vdots \\ \boldsymbol{x}^{(n)}\boldsymbol{v} \end{bmatrix} = \begin{bmatrix} z^{(1)} \\ z^{(2)} \\ \vdots \\ z^{(n)} \end{bmatrix} \tag{60}$$

把矩阵 $\boldsymbol{X}$ 中任意一行 $\boldsymbol{x}^{(i)}$ 看做是多维坐标系的一个点，运算 $\boldsymbol{x}^{(i)}\boldsymbol{v}$ 则是点 $\boldsymbol{x}^{(i)}$ 在 $\boldsymbol{v}$ 方向映射，$z^{(i)}$ 则是结果在 $\boldsymbol{v}$ 这个新坐标系中的坐标。

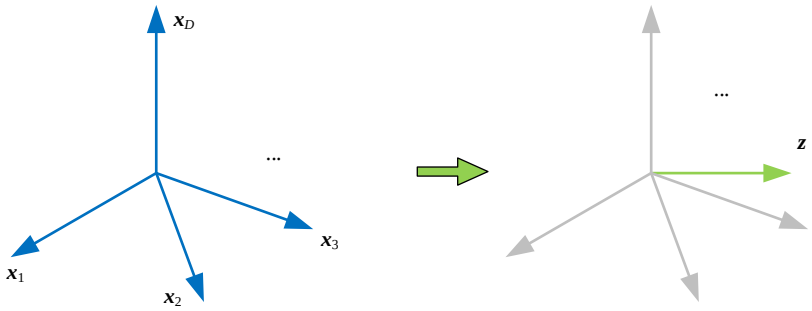


图 25. 映射过程

以鸢尾花数据为例

下面我们将给 $\boldsymbol{v}$ 赋予具体数值来讲解，这样大家更容易理解。

以鸢尾花数据为例，矩阵 $\boldsymbol{X}$ 的 4 列分别对应 4 个特征——花萼长度、花萼宽度、花瓣长度、花瓣宽度。然而，$\boldsymbol{X}\boldsymbol{v} = \boldsymbol{z}$ 结果只有 1 列，相当于只有 1 个特征。

举个例子，如果 $\boldsymbol{v}$ 中有一个元素为 0，其余元素均为 1，如图 26 所示。这个计算的结果是从 $\boldsymbol{X}$ 中提取第 3 列 $\boldsymbol{x}_3$：

$$\boldsymbol{X}\boldsymbol{v} = \begin{bmatrix} \boldsymbol{x}_1 & \boldsymbol{x}_2 & \boldsymbol{x}_3 & \boldsymbol{x}_4 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix} = \boldsymbol{x}_3 \tag{61}$$

也就是说，运算结果只保留第三列花瓣长度相关数据。

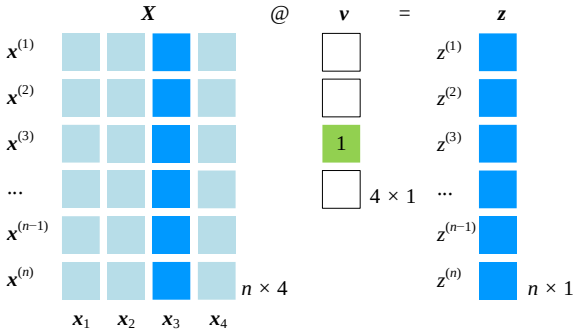


图 26. $\boldsymbol{v}$ 只有一个元素为 1，其余均为 0

再举个例子，若我们想要计算每个样本花萼长度 ($\boldsymbol{x}_1$)、花萼宽度 ($\boldsymbol{x}_2$) 的平均值，可以通过如下运算得到：

$$\boldsymbol{X}\boldsymbol{v}=\begin{bmatrix}\boldsymbol{x}_1 & \boldsymbol{x}_2 & \boldsymbol{x}_3 & \boldsymbol{x}_4\end{bmatrix}\begin{bmatrix}1/2\\1/2\\0\\0\end{bmatrix}=\frac{\boldsymbol{x}_1+\boldsymbol{x}_2}{2} \tag{62}$$

同理，下式可以计算每个样本花萼长度 ($\boldsymbol{x}_1$)、花萼宽度 ($\boldsymbol{x}_2$)、花瓣长度 ($\boldsymbol{x}_3$)、花瓣宽度 ($\boldsymbol{x}_4$) 四个特征平均值：

$$\boldsymbol{X}\boldsymbol{v}=\begin{bmatrix}\boldsymbol{x}_1 & \boldsymbol{x}_2 & \boldsymbol{x}_3 & \boldsymbol{x}_4\end{bmatrix}\begin{bmatrix}1/4\\1/4\\1/4\\1/4\end{bmatrix}=\frac{\boldsymbol{x}_1+\boldsymbol{x}_2+\boldsymbol{x}_3+\boldsymbol{x}_4}{4} \tag{63}$$

几何角度来看上式运算，相当于 4 维空间的散点，被压缩到了一条数轴上，具体如图 27。

注意，4 维空间的散点图是不能在纸面上可视化的，图 27 中 4 维空间的散点仅仅是示意图而已。

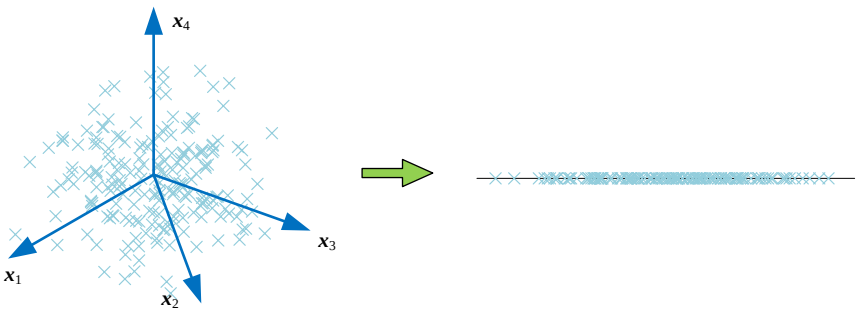


图 27. 4 维空间散点压缩到一条数轴上

5.10 矩阵乘矩阵：映射到多维

两个方向映射

有了上一节内容做基础，这一节我们介绍矩阵乘法在多维映射中扮演色角色。

将矩阵 $\boldsymbol{X}$ 将朝着 $[\boldsymbol{v}_1, \boldsymbol{v}_2]$ 两个方向映射，如图 28 所给示例。图中这个映射提取 $\boldsymbol{X}$ 的第 1、3 两列，并将两者顺序调换。

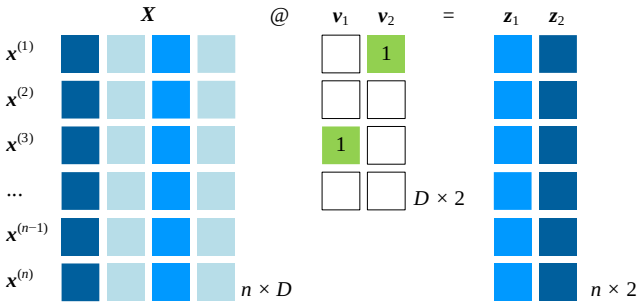


图 28. 矩阵 $\boldsymbol{X}$ 朝两个方向映射

想象一个由鸢尾花四个维度构造的空间 $\mathbb{R}^4$，图 28 相当于将鸢尾花数据映射在一个平面上 $\mathbb{R}^2$，得到的是平面散点图，过程如图 29 所示。根据上一节内容，平面上的数据可以进一步压缩到一条直线上，具体如图 30 所示。

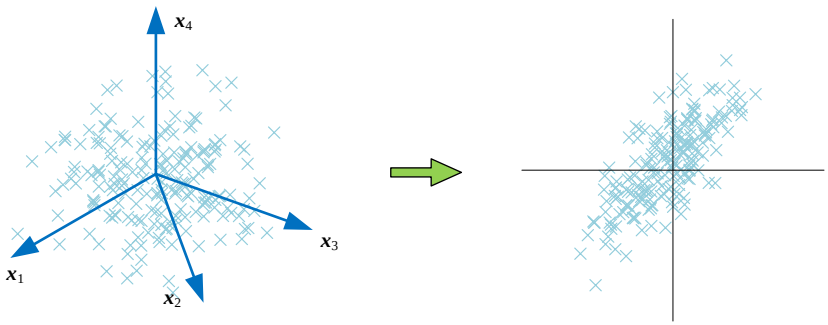


图 29. 4 维空间散点压缩到平面上

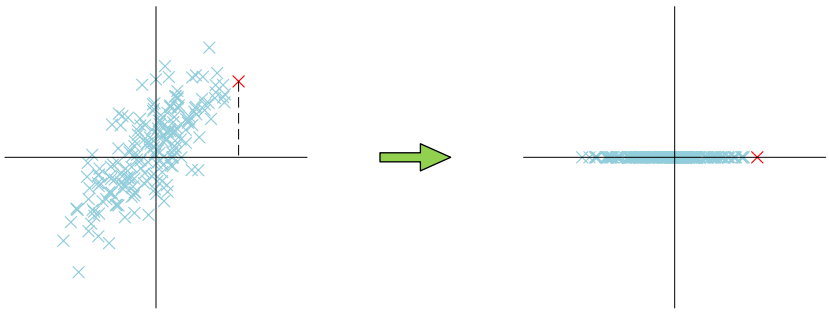


图 30. 平面散点进一步压缩到一维数轴上

多个方向映射

矩阵 $\boldsymbol{X}$ 有 D 个维度，可以通过变换，将 $\boldsymbol{X}$ 映射到另外一个 D 维度的空间中。

下例中，$\boldsymbol{V} = [\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_D]$，$\boldsymbol{Z}$ 对应的每一行元素则是新坐标系中各个维度的坐标值：

$$\boldsymbol{XV} = \begin{bmatrix} \boldsymbol{x}^{(1)} \\ \boldsymbol{x}^{(2)} \\ \vdots \\ \boldsymbol{x}^{(n)} \end{bmatrix} \begin{bmatrix} \boldsymbol{v}_1 & \boldsymbol{v}_2 & \cdots & \boldsymbol{v}_D \end{bmatrix} = \begin{bmatrix} \begin{bmatrix} \boldsymbol{x}^{(1)}\boldsymbol{v}_1 \\ \boldsymbol{x}^{(2)}\boldsymbol{v}_1 \\ \vdots \\ \boldsymbol{x}^{(n)}\boldsymbol{v}_1 \end{bmatrix} & \begin{bmatrix} \boldsymbol{x}^{(1)}\boldsymbol{v}_1 \\ \boldsymbol{x}^{(2)}\boldsymbol{v}_1 \\ \vdots \\ \boldsymbol{x}^{(n)}\boldsymbol{v}_1 \end{bmatrix} & \cdots & \begin{bmatrix} \boldsymbol{x}^{(1)}\boldsymbol{v}_D \\ \boldsymbol{x}^{(2)}\boldsymbol{v}_D \\ \vdots \\ \boldsymbol{x}^{(n)}\boldsymbol{v}_D \end{bmatrix} \end{bmatrix} = \boldsymbol{Z} = \begin{bmatrix} \boldsymbol{z}^{(1)} \\ \boldsymbol{z}^{(2)} \\ \vdots \\ \boldsymbol{z}^{(n)} \end{bmatrix} \tag{64}$$

上式中矩阵 $\boldsymbol{V}$ 为方阵。

也就是说 $\boldsymbol{V}$ 是 $\boldsymbol{X}$ 和 $\boldsymbol{Z}$ 的桥梁：

$$\boldsymbol{X} = \begin{bmatrix} \boldsymbol{x}^{(1)} \\ \boldsymbol{x}^{(2)} \\ \vdots \\ \boldsymbol{x}^{(n)} \end{bmatrix} \underset{\boldsymbol{V}^{-1}}{\overset{\boldsymbol{V}}{\rightleftharpoons}} \boldsymbol{Z} = \begin{bmatrix} \boldsymbol{z}^{(1)} \\ \boldsymbol{z}^{(2)} \\ \vdots \\ \boldsymbol{z}^{(n)} \end{bmatrix} \tag{65}$$

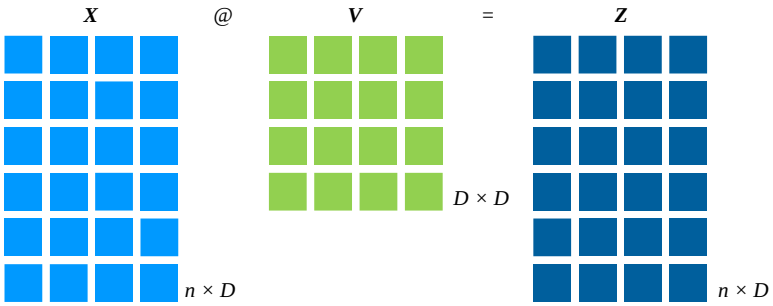


图 31. 一个 D 维度空间 $\boldsymbol{X}$ 数据映射到另一个 D 维度空间

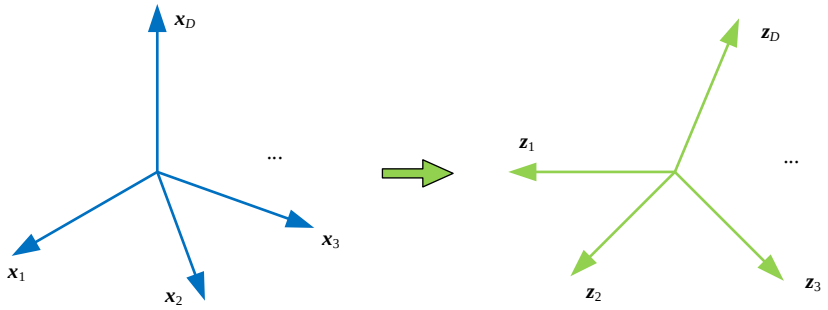


图 32. 多维空间映射

列向量形式

当读者见到如下形式时，也不用慌张。这个也是上文介绍的映射，只不过 $\boldsymbol{x}$ 为列向量：

$$\boldsymbol{V}\boldsymbol{x}=\boldsymbol{z} \tag{66}$$

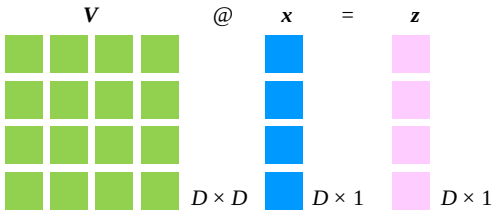


图 33. $\boldsymbol{V}\boldsymbol{x}=\boldsymbol{z}$

(66) 左右转置，便得到类似 (64) 的结构：

$$\boldsymbol{x}^{\mathrm{T}}\boldsymbol{V}^{\mathrm{T}}=\boldsymbol{z}^{\mathrm{T}} \tag{67}$$

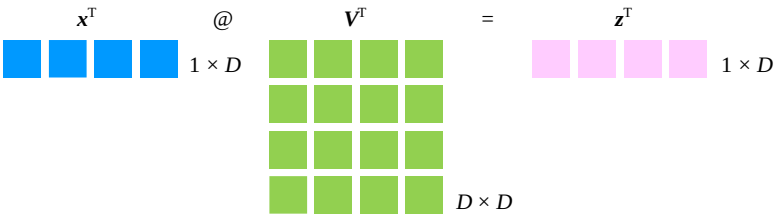


图 34. 等式 $\boldsymbol{V}\boldsymbol{x}=\boldsymbol{z}$ 左右转置

映射是数据转换的重要手段。可以说，本书后续介绍的内容几乎都离不开映射，比如几何变换、特征值分解、奇异值分解等等。对映射这个概念陌生的读者不要担心，本书后续将详细介绍映射相关内容。

5.11 长方阵：奇异值分解，格拉姆矩阵，张量积

本节介绍和长方形矩阵有关的重要矩阵乘法。

奇异值分解

请读者格外注意图 35 所示的矩阵乘法结构。这两种形式经常出现在**奇异值分解** (singular vector decomposition, SVD) 和**主成分分析** (principal component analysis, PCA) 当中。

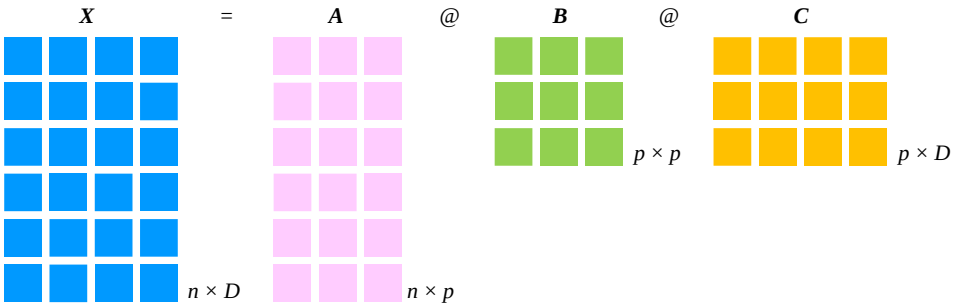


图 35. 三个矩阵相乘

请读者注意图 35 中，D 和 p 的大小关系；不同关系对应着不同类型的特征值分解。

格拉姆矩阵

将矩阵 $\boldsymbol{X}$ 写成一排列向量，如下：

$$\boldsymbol{X}_{n\times D}=\begin{bmatrix} x_{1,1} & x_{1,2} & \cdots & x_{1,D} \\ x_{2,1} & x_{2,2} & \cdots & x_{2,D} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n,1} & x_{n,2} & \cdots & x_{n,D} \end{bmatrix}=\begin{bmatrix} \boldsymbol{x}_1 & \boldsymbol{x}_2 & \cdots & \boldsymbol{x}_D \end{bmatrix} \tag{68}$$

转置 $\boldsymbol{X}^{\mathrm{T}}$ ($D \times n$) 乘矩阵 $\boldsymbol{X}$ ($n \times D$)，得到一个 $D \times D$ 方阵 $\boldsymbol{X}^{\mathrm{T}}\boldsymbol{X}$，可以写成：

$$\boldsymbol{G} = \boldsymbol{X}^{\mathrm{T}}\boldsymbol{X} = \begin{bmatrix} \boldsymbol{x}_1^{\mathrm{T}} \\ \boldsymbol{x}_2^{\mathrm{T}} \\ \vdots \\ \boldsymbol{x}_D^{\mathrm{T}} \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_1 & \boldsymbol{x}_2 & \cdots & \boldsymbol{x}_D \end{bmatrix} = \begin{bmatrix} \boldsymbol{x}_1^{\mathrm{T}}\boldsymbol{x}_1 & \boldsymbol{x}_1^{\mathrm{T}}\boldsymbol{x}_2 & \cdots & \boldsymbol{x}_1^{\mathrm{T}}\boldsymbol{x}_D \\ \boldsymbol{x}_2^{\mathrm{T}}\boldsymbol{x}_1 & \boldsymbol{x}_2^{\mathrm{T}}\boldsymbol{x}_2 & \cdots & \boldsymbol{x}_2^{\mathrm{T}}\boldsymbol{x}_D \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{x}_D^{\mathrm{T}}\boldsymbol{x}_1 & \boldsymbol{x}_D^{\mathrm{T}}\boldsymbol{x}_2 & \cdots & \boldsymbol{x}_D^{\mathrm{T}}\boldsymbol{x}_D \end{bmatrix} \tag{69}$$

(69) 中的 $\boldsymbol{G}$ 也称**格拉姆矩阵** (Gram matrix)。格拉姆矩阵在数据分析、机器学习算法中有重要作用。

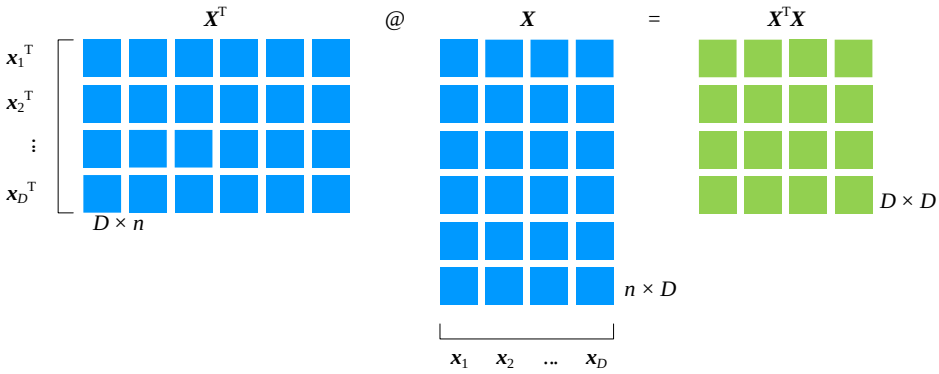


图 36. $\boldsymbol{X}^{\mathrm{T}}\boldsymbol{X}$ 运算过程

$\boldsymbol{X}^{\mathrm{T}}\boldsymbol{X}$ 的 (i,j) 元素是 $\boldsymbol{X}$ 中第 i 列向量转置乘以 $\boldsymbol{X}$ 的第 j 列向量：

$$\left(\boldsymbol{X}^{\mathrm{T}}\boldsymbol{X}\right)_{i,j} = \boldsymbol{x}_i^{\mathrm{T}}\boldsymbol{x}_j \tag{70}$$

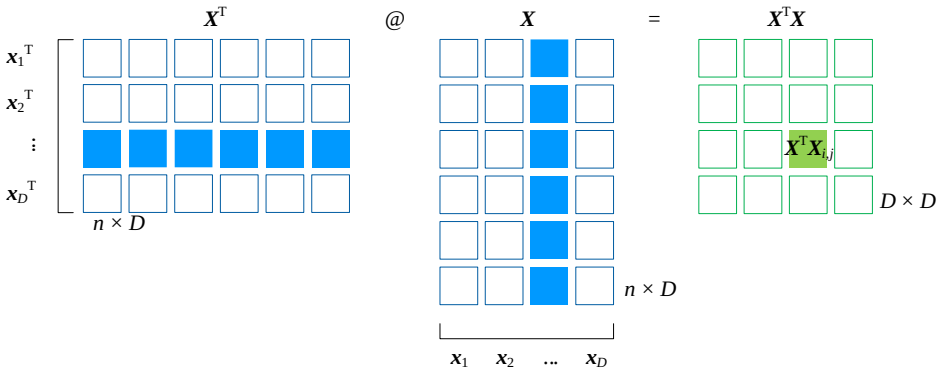


图 37. $\boldsymbol{X}^{\mathrm{T}}\boldsymbol{X}$ 的 (i,j) 元素

标量积

$\boldsymbol{G}$ 还可以写成标量积：

$$\boldsymbol{G}=\begin{bmatrix}\boldsymbol{x}_1\cdot\boldsymbol{x}_1 & \boldsymbol{x}_1\cdot\boldsymbol{x}_2 & \cdots & \boldsymbol{x}_1\cdot\boldsymbol{x}_D\\ \boldsymbol{x}_2\cdot\boldsymbol{x}_1 & \boldsymbol{x}_2\cdot\boldsymbol{x}_2 & \cdots & \boldsymbol{x}_2\cdot\boldsymbol{x}_D\\ \vdots & \vdots & \ddots & \vdots\\ \boldsymbol{x}_D\cdot\boldsymbol{x}_1 & \boldsymbol{x}_D\cdot\boldsymbol{x}_2 & \cdots & \boldsymbol{x}_D\cdot\boldsymbol{x}_D\end{bmatrix}=\begin{bmatrix}\langle\boldsymbol{x}_1,\boldsymbol{x}_1\rangle & \langle\boldsymbol{x}_1,\boldsymbol{x}_2\rangle & \cdots & \langle\boldsymbol{x}_1,\boldsymbol{x}_D\rangle\\ \langle\boldsymbol{x}_2,\boldsymbol{x}_1\rangle & \langle\boldsymbol{x}_2,\boldsymbol{x}_2\rangle & \cdots & \langle\boldsymbol{x}_2,\boldsymbol{x}_D\rangle\\ \vdots & \vdots & \ddots & \vdots\\ \langle\boldsymbol{x}_D,\boldsymbol{x}_1\rangle & \langle\boldsymbol{x}_D,\boldsymbol{x}_2\rangle & \cdots & \langle\boldsymbol{x}_D,\boldsymbol{x}_D\rangle\end{bmatrix} \tag{71}$$

显然，格拉姆矩阵 $\boldsymbol{G}$ 为对称矩阵：

$$\boldsymbol{G}^{\mathrm{T}}=\left(\boldsymbol{X}^{\mathrm{T}}\boldsymbol{X}\right)^{\mathrm{T}}=\boldsymbol{X}^{\mathrm{T}}\boldsymbol{X}=\boldsymbol{G} \tag{72}$$

本书后续将要介绍的**协方差矩阵** (covariance matrix) 计算时，也将采用类似上式的计算思路。

张量积

将矩阵 $\boldsymbol{X}$ 写成一系列行向量，如下：

$$\boldsymbol{X}_{n\times D}=\begin{bmatrix}x_{1,1} & x_{1,2} & \cdots & x_{1,D}\\ x_{2,1} & x_{2,2} & \cdots & x_{2,D}\\ \vdots & \vdots & \ddots & \vdots\\ x_{n,1} & x_{n,2} & \cdots & x_{n,D}\end{bmatrix}=\begin{bmatrix}\boldsymbol{x}^{(1)}\\ \boldsymbol{x}^{(2)}\\ \vdots\\ \boldsymbol{x}^{(n)}\end{bmatrix} \tag{73}$$

格拉姆矩阵 $\boldsymbol{X}^{\mathrm{T}}\boldsymbol{X}$，可以写成一系列张量积的和：

$$\boldsymbol{G}=\boldsymbol{X}^{\mathrm{T}}\boldsymbol{X}=\begin{bmatrix}\boldsymbol{x}^{(1)\mathrm{T}} & \boldsymbol{x}^{(2)\mathrm{T}} & \cdots & \boldsymbol{x}^{(n)\mathrm{T}}\end{bmatrix}\begin{bmatrix}\boldsymbol{x}^{(1)}\\ \boldsymbol{x}^{(2)}\\ \vdots\\ \boldsymbol{x}^{(n)}\end{bmatrix}=\sum_{i=1}^{n}\boldsymbol{x}^{(i)\mathrm{T}}\boldsymbol{x}^{(i)}=\sum_{i=1}^{n}\boldsymbol{x}^{(i)}\otimes\boldsymbol{x}^{(i)} \tag{74}$$

式中，行向量张量积等价于转置乘列向量后的张量积。

元素平方和

此外，下式可以计算得到矩阵 $\boldsymbol{X}$ 的所有元素的平方和：

$$\begin{aligned}\operatorname{trace}\left(\boldsymbol{X}^{\mathrm{T}}\boldsymbol{X}\right)&=\operatorname{trace}\begin{bmatrix}\boldsymbol{x}_1\cdot\boldsymbol{x}_1 & \boldsymbol{x}_1\cdot\boldsymbol{x}_2 & \cdots & \boldsymbol{x}_1\cdot\boldsymbol{x}_D\\ \boldsymbol{x}_2\cdot\boldsymbol{x}_1 & \boldsymbol{x}_2\cdot\boldsymbol{x}_2 & \cdots & \boldsymbol{x}_2\cdot\boldsymbol{x}_D\\ \vdots & \vdots & \ddots & \vdots\\ \boldsymbol{x}_D\cdot\boldsymbol{x}_1 & \boldsymbol{x}_D\cdot\boldsymbol{x}_2 & \cdots & \boldsymbol{x}_D\cdot\boldsymbol{x}_D\end{bmatrix}\\ &=\boldsymbol{x}_1\cdot\boldsymbol{x}_1+\boldsymbol{x}_2\cdot\boldsymbol{x}_2+\cdots+\boldsymbol{x}_D\cdot\boldsymbol{x}_D\\ &=\sum_{i=1}^{n}x_{i,1}^2+\sum_{i=1}^{n}x_{i,2}^2+\cdots+\sum_{i=1}^{n}x_{i,D}^2\\ &=\sum_{j=1}^{D}\sum_{i=1}^{n}x_{i,j}^2\end{aligned} \tag{75}$$

上一章讲解矩阵迹 (trace) 时提到，如果 $\boldsymbol{AB}$ 和 $\boldsymbol{BA}$ 都存在，$\mathrm{tr}(\boldsymbol{AB}) = \mathrm{tr}(\boldsymbol{BA})$。也就是说，对于 (75)，下式成立：

$$\mathrm{trace}\left(\boldsymbol{X}^{\mathrm{T}}\boldsymbol{X}\right) = \mathrm{trace}\left(\boldsymbol{X}\boldsymbol{X}^{\mathrm{T}}\right) = \sum_{j=1}^{D}\sum_{i=1}^{n} x_{i,j}^{2} \tag{76}$$

本书后文还会在不同位置用到 $\mathrm{tr}(\boldsymbol{AB}) = \mathrm{tr}(\boldsymbol{BA})$，请大家格外注意。

另一个格拉姆矩阵

本节前文的数据矩阵 $\boldsymbol{X}$ 是细高的，它转置之后得到粗矮的矩阵 $\boldsymbol{X}^{\mathrm{T}}$。而 $\boldsymbol{X}^{\mathrm{T}}$ 也有自己的格拉姆矩阵，即矩阵 $\boldsymbol{X}$ ($n \times D$) 乘其转置 $\boldsymbol{X}^{\mathrm{T}}$ ($D \times n$)，得到一个 $n \times n$ 格拉姆矩阵：

$$\begin{aligned}\boldsymbol{X}\boldsymbol{X}^{\mathrm{T}} &= \begin{bmatrix}\boldsymbol{x}_1 & \boldsymbol{x}_2 & \cdots & \boldsymbol{x}_D\end{bmatrix}\begin{bmatrix}\boldsymbol{x}_1^{\mathrm{T}} \\ \boldsymbol{x}_2^{\mathrm{T}} \\ \vdots \\ \boldsymbol{x}_D^{\mathrm{T}}\end{bmatrix} \\ &= \boldsymbol{x}_1\boldsymbol{x}_1^{\mathrm{T}} + \boldsymbol{x}_2\boldsymbol{x}_2^{\mathrm{T}} + \cdots + \boldsymbol{x}_D\boldsymbol{x}_D^{\mathrm{T}} \\ &= \sum_{i=1}^{D}\boldsymbol{x}_i\boldsymbol{x}_i^{\mathrm{T}}\end{aligned} \tag{77}$$

观察 (77)，大家是否也发现了张量积的影子？

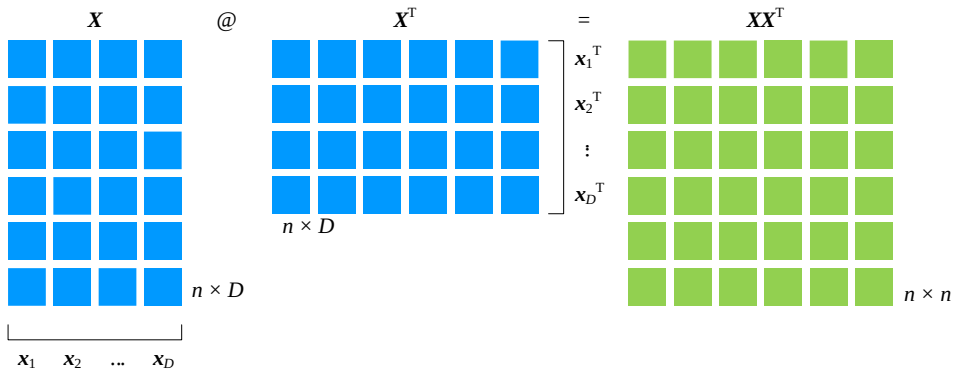


图 38. $\boldsymbol{X}\boldsymbol{X}^{\mathrm{T}}$ 运算过程

用张量积运算工具，(77) 可以写成：

$$\begin{aligned}\boldsymbol{X}\boldsymbol{X}^{\mathrm{T}} &= \boldsymbol{x}_1\boldsymbol{x}_1^{\mathrm{T}} + \boldsymbol{x}_2\boldsymbol{x}_2^{\mathrm{T}} + \cdots + \boldsymbol{x}_D\boldsymbol{x}_D^{\mathrm{T}} \\ &= \boldsymbol{x}_1 \otimes \boldsymbol{x}_1 + \boldsymbol{x}_2 \otimes \boldsymbol{x}_2 + \cdots + \boldsymbol{x}_D \otimes \boldsymbol{x}_D \\ &= \sum_{i=1}^{D}\boldsymbol{x}_i \otimes \boldsymbol{x}_i\end{aligned} \tag{78}$$

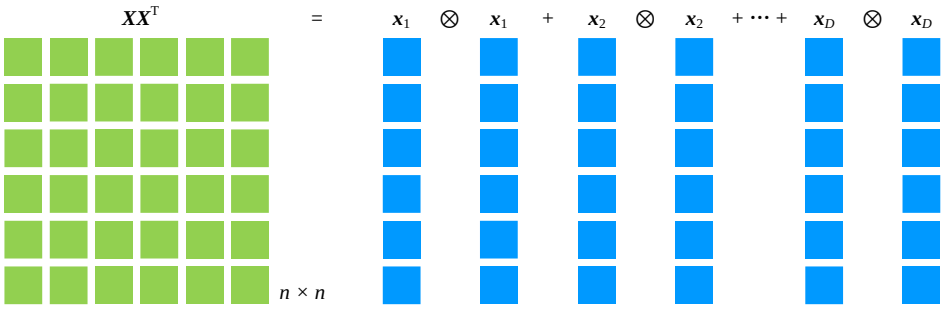


图 39. 用张量积表达 $\boldsymbol{X}\boldsymbol{X}^{\mathrm{T}}$运算过程

5.12 爱因斯坦求和约定

本书之前的所有矩阵运算都是适用于二阶情况，比如 $n \times D$ 的这种 n 行、D 列形式。在数据科学和机器学习很多实践中，我们不可避免地要处理高阶矩阵，比如图 40 所示三阶矩阵。Python 中 Xarray 专门用来存储和运算高阶矩阵。

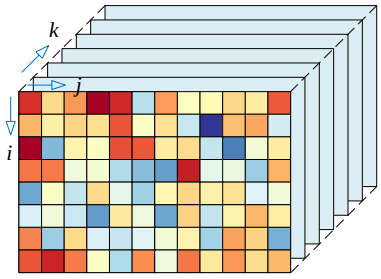


图 40. 三维数组，三阶矩阵

本节则要介绍一种简洁表达高阶矩阵运算的数学工具——爱因斯坦求和约定 (Einstein summation convention 或 Einstein notation)。本节介绍如何用 `numpy.einsum()` 函数完成本书前文介绍的主要线性代数运算。PyTorch 中 `torch.einsum()` 函数原理和 `numpy.einsum()` 基本相同，本书不特别介绍。

使用 `numpy.einsum()` 时，大家记住一个要点——输入中重复的索引元素相乘，输出中消去的索引意味着相加。

举个例子，矩阵 $\boldsymbol{A}$ 和 $\boldsymbol{B}$ 相乘用 `numpy.einsum()` 函数可以写成：

```
np.einsum('ij,jk->ik', A, B)
```

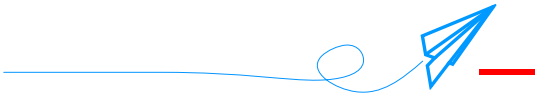
“->” 之前分别为矩阵 $\boldsymbol{A}$ 和 $\boldsymbol{B}$ 的索引，它们用逗号隔开；“->” 之后为输出结果的索引。矩阵 $\boldsymbol{A}$ 行索引为 `i`，列索引为 `j`。矩阵 $\boldsymbol{B}$ 行索引为 `j`，列索引为 `k`。`j` 为重复索引，因此在这个方向上元素相乘。而输出结果索引为 `ik`，没有 `j`，因此在 `j` 索引方向上存在求和运算。

表 1 总结使用 numpy.einsum() 完成常见线性代数运算。有关 numpy.einsum()，不需要大家掌握。希望大家在日后有可能用到爱因斯坦求和约定时，再回过头来深入学习。

表 1. 使用 numpy.einsum() 完成常见线性代数运算

运算	使用 numpy.einsum() 完成运算
向量 $\boldsymbol{a}$ 所有元素求和	np.einsum('ij->',a) np.einsum('i->',a_1D)
等长向量 $\boldsymbol{a}$ 和 $\boldsymbol{b}$ 的逐项积	np.einsum('ij,ij->ij',a,b) np.einsum('i,i->i',a_1D,b_1D)
等长向量 $\boldsymbol{a}$ 和 $\boldsymbol{b}$ 的向量内积	np.einsum('ij,ij->',a,b) np.einsum('i,i->',a_1D,b_1D)
向量 $\boldsymbol{a}$ 和自身的张量积	np.einsum('ij,ji->ij',a,a) np.einsum('i,j->ij',a_1D,a_1D)
向量 $\boldsymbol{a}$ 和 $\boldsymbol{b}$ 的张量积	np.einsum('ij,ji->ij',a,b) np.einsum('i,j->ij',a_1D,b_1D)
矩阵 $\boldsymbol{A}$ 的转置	np.einsum('ji',A)
矩阵 $\boldsymbol{A}$ 所有元素求和	np.einsum('ij->',A)
矩阵 $\boldsymbol{A}$ 对每一列元素求和	np.einsum('ij->j',A)
矩阵 $\boldsymbol{A}$ 对每一行元素求和	np.einsum('ij->i',A)
提取方阵 $\boldsymbol{A}$ 的对角元素	np.einsum('ii->i',A)
计算方阵 $\boldsymbol{A}$ 的迹 trace($\boldsymbol{A}$)	np.einsum('ii->',A)
计算矩阵 $\boldsymbol{A}$ 和 $\boldsymbol{B}$ 乘积	np.einsum('ij,jk->ik', A, B)
乘积 $\boldsymbol{AB}$ 结果所有元素求和	np.einsum('ij,jk->', A, B)
矩阵 $\boldsymbol{A}$ 和 $\boldsymbol{B}$ 相乘后再转置，即$(\boldsymbol{AB})^{\mathrm{T}}$	np.einsum('ij,jk->ki', A, B)
形状相同矩阵 $\boldsymbol{A}$ 和 $\boldsymbol{B}$ 逐项积	np.einsum('ij,ij->ij', A, B)

表 1 中变量定义和运算都在 Bk4_Ch5_01.py 中。



本章全景展示了常见的矩阵乘法形态，每种形态的矩阵乘法都很重要，因此也不能用四幅图来总结本章重要内容。

大家想要活用线性代数这个宝库中的各种数学工具，熟练掌握矩阵乘法规则是绕不过去的一道门槛。一点建议，大家在学习中，要试图从几何和数据这两个视角去理解不同的矩阵运算。这也是本书要特别强化的一点。

此外，本章针对矩阵乘法运算也没有给出任何代码，因为在 Numpy 中矩阵乘法运算符常用的就是 @。本章最后介绍了爱因斯坦求和约定，不要求大家掌握。

矩阵乘法规则像是枷锁，它条条框框、冷酷无情、不容妥协；但是，在枷锁下，我们看到了矩阵乘法的另一面——无拘无束、血脉偾张、海纳百川。

希望大家一边学习本书剩余内容，一边能够不断回头看这一章内容，相信大家一定会和我一样，叹服于矩阵乘法展现出来的自由、包容，和纯粹的美。