

# 5

## Block Matrix

# 分块矩阵

将大矩阵切成小块，简化运算



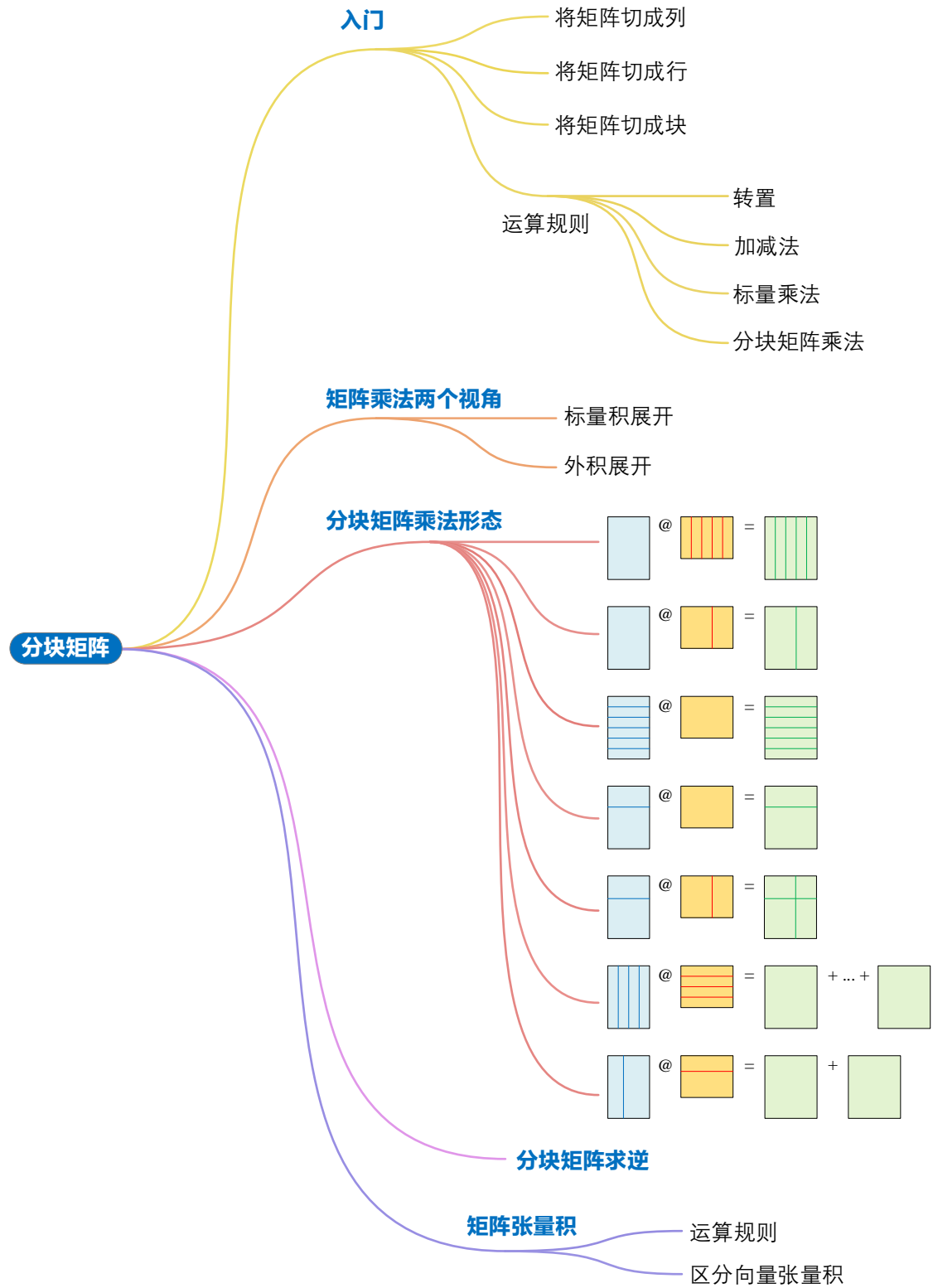
数学的精髓在于自由。

*The essence of mathematics is in its freedom.*

—— 格奥尔格·康托尔 (Georg Cantor) | 德国数学家 | 1845 ~ 1918



- ◀ `numpy.kron()` 计算矩阵张量积
- ◀ `numpy.random.random_integers()` 生成随机整数
- ◀ `numpy.zeros_like()` 用来生成和输入矩阵形状相同的零矩阵
- ◀ `seaborn.heatmap()` 绘制热图



本 PDF 文件为作者草稿，发布目的为方便读者在移动终端学习，终稿内容以清华大学出版社纸质出版物为准。

版权归清华大学出版社所有，请勿商用，引用请注明出处。

代码及 PDF 文件下载：<https://github.com/Visualize-ML>

本书配套微课视频均发布在 B 站——生姜 DrGinger：<https://space.bilibili.com/513194466>

欢迎大家批评指教，本书专属邮箱：[jiang.visualize.ml@gmail.com](mailto:jiang.visualize.ml@gmail.com)

## 5.1 分块矩阵：横平竖直切豆腐

**分块矩阵** (block matrix 或 partitioned matrix) 将一个矩阵用若干条横线和竖线分割成多个**子块矩阵** (submatrices)。这样可以简化运算，同时也会让运算过程变得更加清晰。

白话讲，分块矩阵好比横平竖直切豆腐；但是下刀的手法很有讲究，这是本章后文要着重探讨的内容。

### 切丝切条

实际上，本书一开始就已经不知不觉地使用了分块矩阵这一重要思路。

大家已经清楚知道，如图 1 所示，矩阵  $X$  可以看做是由行向量或列向量按照一定规则构造而成。

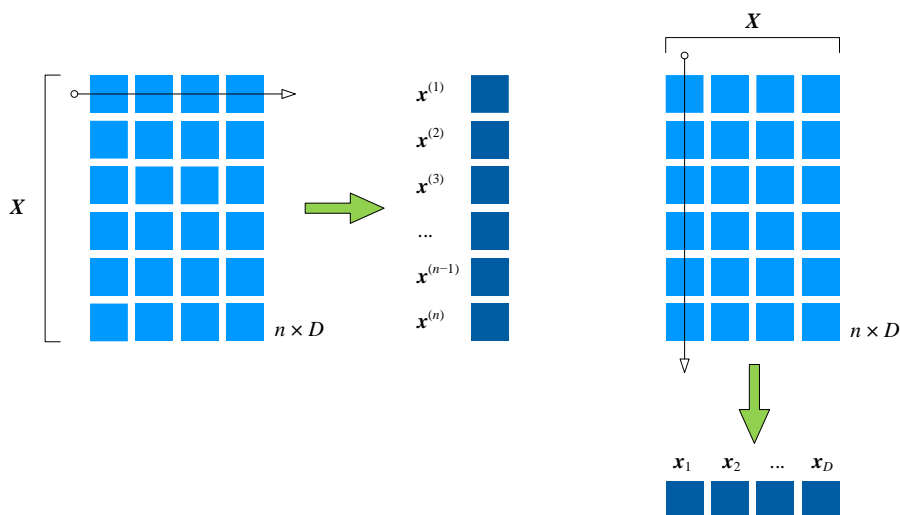


图 1. 矩阵可以写成一系列行向量或列向量

矩阵  $X$  每行之间切一刀，得到一组行向量：

$$X_{n \times D} = \begin{bmatrix} x^{(1)} \\ x^{(2)} \\ \vdots \\ x^{(n)} \end{bmatrix} = \begin{bmatrix} x_{1,1} & x_{1,2} & \cdots & x_{1,D} \\ x_{2,1} & x_{2,2} & \cdots & x_{2,D} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n,1} & x_{n,2} & \cdots & x_{n,D} \end{bmatrix} \quad (1)$$

矩阵  $X$  在每列之间切一刀，将  $X$  切成一组列向量：

$$\mathbf{X}_{n \times D} = [\mathbf{x}_1 \quad \mathbf{x}_2 \quad \cdots \quad \mathbf{x}_D] = \begin{bmatrix} x_{1,1} & x_{1,2} & \cdots & x_{1,D} \\ x_{2,1} & x_{2,2} & \cdots & x_{2,D} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n,1} & x_{n,2} & \cdots & x_{n,D} \end{bmatrix} \quad (2)$$

## 切块

下面介绍分块矩阵其他切法。给出如下矩阵  $\mathbf{A}$ ：

$$\mathbf{A} = \begin{bmatrix} 1 & 2 & 3 & 0 & 0 \\ 4 & 5 & 6 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \quad (3)$$

我们把矩阵  $\mathbf{A}$  横竖都切一刀，得到四个子矩阵：

$$\mathbf{A} = \begin{bmatrix} 1 & 2 & 3 & 0 & 0 \\ 4 & 5 & 6 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \quad (4)$$

给每个子矩阵起个名字，矩阵  $\mathbf{A}$  记做：

$$\mathbf{A} = \begin{bmatrix} \mathbf{A}_{1,1} & \mathbf{A}_{1,2} \\ \mathbf{A}_{2,1} & \mathbf{A}_{2,2} \end{bmatrix} \quad (5)$$

也就是，

$$\begin{aligned} \mathbf{A}_{1,1} &= \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}, \quad \mathbf{A}_{1,2} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \\ \mathbf{A}_{2,1} &= \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}, \quad \mathbf{A}_{2,2} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \end{aligned} \quad (6)$$



Numpy 中矩阵分块可以用行列序数就做到。而 `numpy.block()` 函数可以用子块矩阵结合得到原矩阵。请大家参考 `Bk4_Ch5_01.py`。

```
# Bk4_Ch5_01.py

import numpy as np

A = np.array([[1, 2, 3, 0, 0],
              [4, 5, 6, 0, 0],
              [0, 0, 0, -1, 0],
              [0, 0, 0, 0, 1]])

# NumPy array slicing
```

```

A_1_1 = A[0:2,0:3]
A_1_2 = A[0:2,3:]
# A_1_2 = A[0:2,-2:]
A_2_1 = A[2:,0:3]
# A_2_1 = A[-2:,0:3]
A_2_2 = A[2:,3:]
# A_2_2 = A[-2,-2:]

# Assemble a matrix from nested lists of blocks
A_ = np.block([[A_1_1, A_1_2],
               [A_2_1, A_2_2]])

```

## 转置

一般情况， $A_{ij}$  的行数记做  $n_i$ ，列数  $D_j$ ；如果矩阵  $A$  的形状为  $n \times D$ ，按 (5) 分割得到的子块矩阵的行、列数满足：

$$n_1 + n_2 = n, \quad D_1 + D_2 = D \quad (7)$$

对  $A$  求转置，得到：

$$A^T = \begin{bmatrix} A_{1,1}^T & A_{2,1}^T \\ A_{1,2}^T & A_{2,2}^T \end{bmatrix} \quad (8)$$

代入具体值，得到：

$$A^T = \begin{bmatrix} 1 & 4 & 0 & 0 \\ 2 & 5 & 0 & 0 \\ 3 & 6 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (9)$$

请大家仔细对比 (4) 和 (9)，分析转置前后子块矩阵的变化。

## 标量乘法

矩阵  $A$  的标量乘法：

$$kA = \begin{bmatrix} kA_{1,1} & kA_{1,2} \\ kA_{2,1} & kA_{2,2} \end{bmatrix} \quad (10)$$

## 加减法

给定矩阵  $B$ ，它的形状和  $A$  相同，采用相同的分块法分割  $B$ ，得到：

$$\mathbf{B} = \begin{bmatrix} \mathbf{B}_{1,1} & \mathbf{B}_{1,2} \\ \mathbf{B}_{2,1} & \mathbf{B}_{2,2} \end{bmatrix} \quad (11)$$

矩阵  $\mathbf{A}$  和  $\mathbf{B}$  的相同位置的子块矩阵形状相同，两者相加为对应位置子块分别相加：

$$\mathbf{A} + \mathbf{B} = \begin{bmatrix} \mathbf{A}_{1,1} & \mathbf{A}_{1,2} \\ \mathbf{A}_{2,1} & \mathbf{A}_{2,2} \end{bmatrix} + \begin{bmatrix} \mathbf{B}_{1,1} & \mathbf{B}_{1,2} \\ \mathbf{B}_{2,1} & \mathbf{B}_{2,2} \end{bmatrix} = \begin{bmatrix} \mathbf{A}_{1,1} + \mathbf{B}_{1,1} & \mathbf{A}_{1,2} + \mathbf{B}_{1,2} \\ \mathbf{A}_{2,1} + \mathbf{B}_{2,1} & \mathbf{A}_{2,2} + \mathbf{B}_{2,2} \end{bmatrix} \quad (12)$$

上述规则也适用于减法。

## 矩阵乘法

分块矩阵乘法规则也是基于矩阵乘法规则。 $\mathbf{A}$  和  $\mathbf{B}$  相乘时，首先保证  $\mathbf{A}$  的列数等于  $\mathbf{B}$  的行数； $\mathbf{A}$  和  $\mathbf{B}$  分块时，保证  $\mathbf{A}$  的每一个子块矩阵的列数分别等于  $\mathbf{B}$  的每个子块的行数。这样  $\mathbf{A}$  和  $\mathbf{B}$  相乘可以展开写成：

$$\mathbf{AB} = \begin{bmatrix} \mathbf{A}_{1,1} & \mathbf{A}_{1,2} \\ \mathbf{A}_{2,1} & \mathbf{A}_{2,2} \end{bmatrix} \begin{bmatrix} \mathbf{B}_{1,1} & \mathbf{B}_{1,2} \\ \mathbf{B}_{2,1} & \mathbf{B}_{2,2} \end{bmatrix} = \begin{bmatrix} \mathbf{A}_{1,1}\mathbf{B}_{1,1} + \mathbf{A}_{1,2}\mathbf{B}_{2,1} & \mathbf{A}_{1,1}\mathbf{B}_{1,2} + \mathbf{A}_{1,2}\mathbf{B}_{2,2} \\ \mathbf{A}_{2,1}\mathbf{B}_{1,1} + \mathbf{A}_{2,2}\mathbf{B}_{2,1} & \mathbf{A}_{2,1}\mathbf{B}_{1,2} + \mathbf{A}_{2,2}\mathbf{B}_{2,2} \end{bmatrix} \quad (13)$$

上式中分块矩阵的乘法有两层运算。第一层矩阵乘法将子块视作元素，第二层是子块矩阵之间矩阵乘法。本章后会深入讲解不同形态的分块矩阵乘法。

## 5.2 矩阵乘法第一视角：标量积展开

本书前文以两个  $2 \times 2$  矩阵相乘为例，讲解过观察矩阵乘法的两个视角。本节和下一节回顾这两个视角的同时，进一步从分块矩阵视角深入讲解矩阵乘法规则。

本节讨论矩阵乘法的常规视角——**标量积展开** (scalar product expansion)。

首先回顾矩阵乘法规则。

当矩阵  $\mathbf{A}$  的列数等于矩阵  $\mathbf{B}$  的行数时， $\mathbf{A}$  与  $\mathbf{B}$  可以相乘。比如下例中，矩阵  $\mathbf{A}$  的形状为  $n$  行  $D$  列，矩阵  $\mathbf{B}$  的形状为  $D$  行  $m$  列。 $\mathbf{A}$  与  $\mathbf{B}$  相乘时，相当于  $D$  被消掉。

矩阵相乘得到的矩阵  $\mathbf{C}$  的行数等于矩阵  $\mathbf{A}$  的行数， $\mathbf{C}$  的列数等于  $\mathbf{B}$  的列数，即  $n$  行  $m$  列：

$$\mathbf{C}_{n \times m} = \mathbf{A}_{n \times D} \mathbf{B}_{D \times m} = \mathbf{A}_{n \times D} @ \mathbf{B}_{D \times m} = \begin{bmatrix} c_{1,1} & c_{1,2} & \cdots & c_{1,m} \\ c_{2,1} & c_{2,2} & \cdots & c_{2,m} \\ \vdots & \vdots & \ddots & \vdots \\ c_{n,1} & c_{n,2} & \cdots & c_{n,m} \end{bmatrix} \quad (14)$$

其中，

$$\mathbf{A}_{n \times D} = \begin{bmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,D} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,D} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n,1} & a_{n,2} & \cdots & a_{n,D} \end{bmatrix}, \quad \mathbf{B}_{D \times m} = \begin{bmatrix} b_{1,1} & b_{1,2} & \cdots & b_{1,m} \\ b_{2,1} & b_{2,2} & \cdots & b_{2,m} \\ \vdots & \vdots & \ddots & \vdots \\ b_{D,1} & b_{D,2} & \cdots & b_{D,m} \end{bmatrix} \quad (15)$$

将矩阵  $\mathbf{A}$  写成一组行向量：

$$\mathbf{A}_{n \times D} = \begin{bmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,D} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,D} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n,1} & a_{n,2} & \cdots & a_{n,D} \end{bmatrix}_{n \times D} = \begin{bmatrix} \mathbf{a}^{(1)} \\ \mathbf{a}^{(2)} \\ \vdots \\ \mathbf{a}^{(n)} \end{bmatrix}_{n \times 1} \quad (16)$$

将矩阵  $\mathbf{B}$  写成一组列向量：

$$\mathbf{B}_{D \times m} = \begin{bmatrix} b_{1,1} & b_{1,2} & \cdots & b_{1,m} \\ b_{2,1} & b_{2,2} & \cdots & b_{2,m} \\ \vdots & \vdots & \ddots & \vdots \\ b_{D,1} & b_{D,2} & \cdots & b_{D,m} \end{bmatrix}_{D \times m} = [\mathbf{b}_1 \quad \mathbf{b}_2 \quad \cdots \quad \mathbf{b}_m]_{1 \times m} \quad (17)$$

利用 (16) 和 (17)，矩阵乘积  $\mathbf{AB}$  可以写作：

$$\mathbf{C} = \mathbf{AB} = \begin{bmatrix} \mathbf{a}^{(1)} \\ \mathbf{a}^{(2)} \\ \vdots \\ \mathbf{a}^{(n)} \end{bmatrix}_{n \times 1} [\mathbf{b}_1 \quad \mathbf{b}_2 \quad \cdots \quad \mathbf{b}_m]_{1 \times m} = \begin{bmatrix} \mathbf{a}^{(1)}\mathbf{b}_1 & \mathbf{a}^{(1)}\mathbf{b}_2 & \cdots & \mathbf{a}^{(1)}\mathbf{b}_m \\ \mathbf{a}^{(2)}\mathbf{b}_1 & \mathbf{a}^{(2)}\mathbf{b}_2 & \cdots & \mathbf{a}^{(2)}\mathbf{b}_m \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{a}^{(n)}\mathbf{b}_1 & \mathbf{a}^{(n)}\mathbf{b}_2 & \cdots & \mathbf{a}^{(n)}\mathbf{b}_m \end{bmatrix} \quad (18)$$

上式便是矩阵乘法的常规视角，规则如图 2 所示。

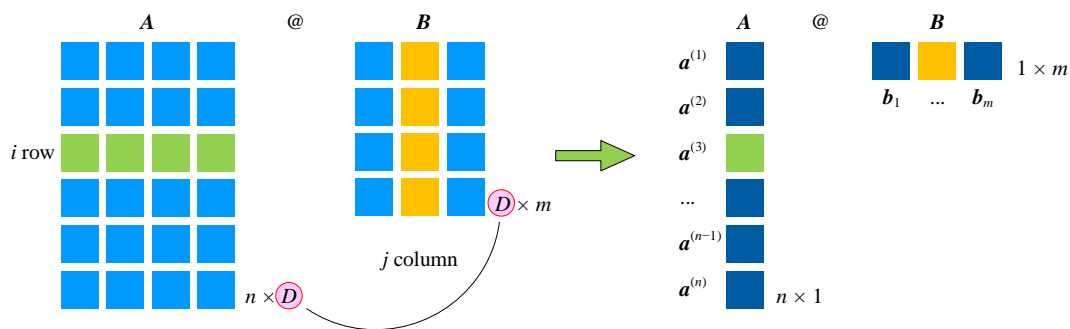


图 2. 矩阵乘法的常规视角

如图 3 所示，矩阵乘积  $\mathbf{C}$  的  $(i,j)$  元素  $c_{i,j}$ ，矩阵  $\mathbf{A}$  的第  $i$  行行向量  $\mathbf{a}^{(i)}$  和矩阵  $\mathbf{B}$  的第  $j$  列列向量  $\mathbf{b}_j$  的乘积：

$$c_{i,j} = \mathbf{a}^{(i)}\mathbf{b}_j \quad (19)$$

白话说，矩阵乘法的常规视角是，左侧矩阵的每个行向量，按规则分别乘右侧矩阵每个列向量。

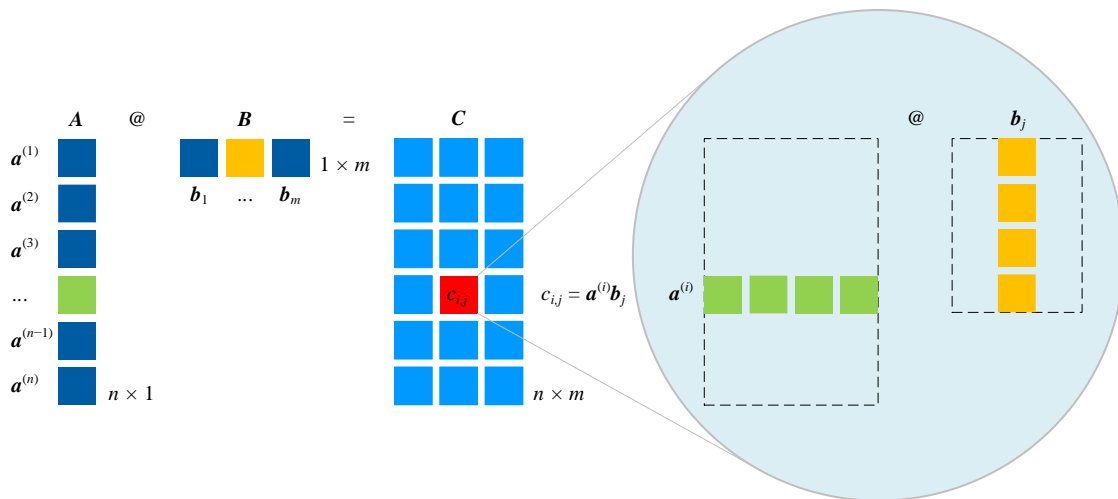


图 3. 矩阵乘法的常规视角中，矩阵乘积  $C$  的  $(i,j)$  元素

## 5.3 矩阵乘法第二视角：外积展开

本节回顾矩阵乘法规则的第二视角——**外积展开** (outer product expansion)。

与矩阵乘法常规视角不同，我们将矩阵  $A$  写成一排列向量：

$$A = \begin{bmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,D} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,D} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n,1} & a_{n,2} & \cdots & a_{n,D} \end{bmatrix} = [a_1 \quad a_2 \quad \cdots \quad a_D] \quad (20)$$

矩阵  $B$  则写成一行行向量：

$$B = \begin{bmatrix} b_{1,1} & b_{1,2} & \cdots & b_{1,m} \\ b_{2,1} & b_{2,2} & \cdots & b_{2,m} \\ \vdots & \vdots & \ddots & \vdots \\ b_{D,1} & b_{D,2} & \cdots & b_{D,m} \end{bmatrix} = \begin{bmatrix} b^{(1)} \\ b^{(2)} \\ \vdots \\ b^{(D)} \end{bmatrix} \quad (21)$$

这样，在计算矩阵乘积  $AB$  时，我们便使用如图 4 所示这个全新的视角。



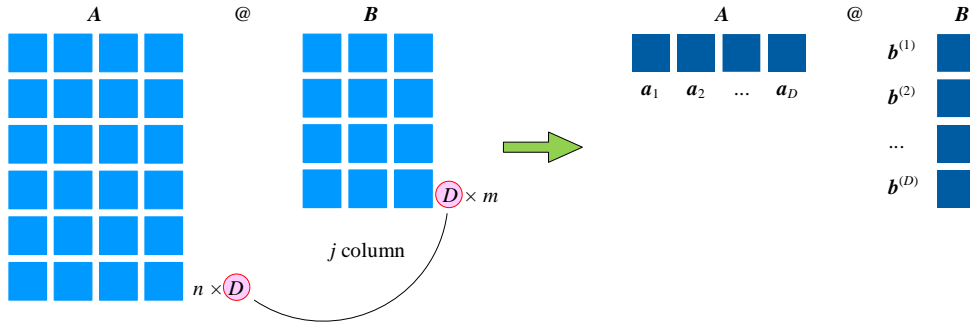


图 4. 矩阵乘法的第二视角

利用 (20) 和 (21)，矩阵乘积  $AB$  展开写成：

$$\begin{aligned}
 C = AB &= [a_1 \ a_2 \ \cdots \ a_D]_{n \times D} \begin{bmatrix} b^{(1)} \\ b^{(2)} \\ \vdots \\ b^{(D)} \end{bmatrix}_{D \times 1} \\
 &= a_1 b^{(1)} + a_2 b^{(2)} + \cdots + a_D b^{(D)} \\
 &= \sum_{i=1}^D a_i b^{(i)}
 \end{aligned} \tag{22}$$

这样，我们将矩阵乘法运算转化成求和运算。

令，

$$C_i = a_i b^{(i)} \tag{23}$$

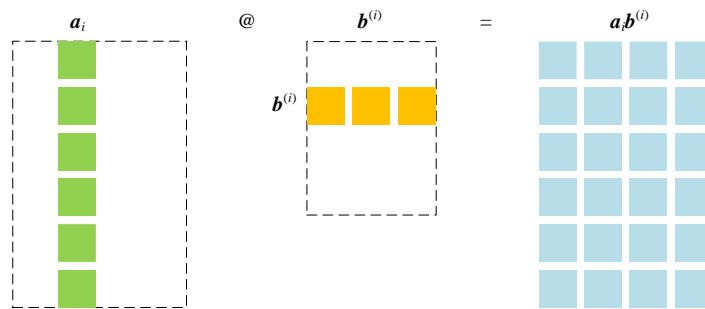
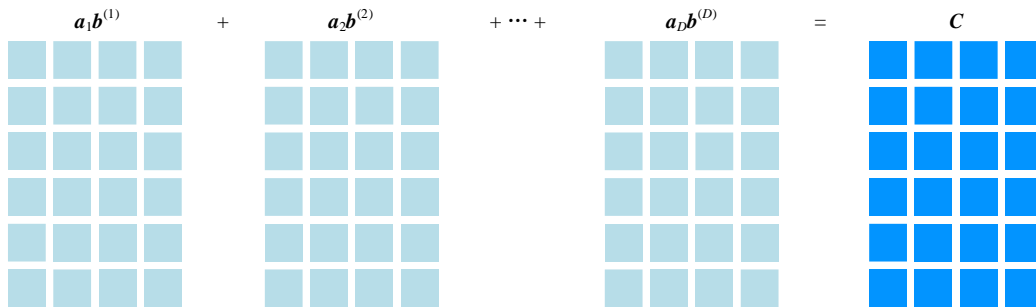


图 5. 列向量  $a_i$  和行向量  $b^{(i)}$  乘积的结果

如图 5 所示，列向量  $a_i$  和行向量  $b^{(i)}$  乘积的结果的形状为  $n \times m$ ，即乘积  $C$  矩阵的形状。通过观察 (22)，可以发现乘积  $C$  矩阵相当于  $D$  个矩阵  $C_i$  叠加得到的结果：

$$C = C_1 + C_2 + \cdots + C_D = \sum_{i=1}^D C_i \quad (24)$$

图 6. 乘积  $C$  矩阵相当于  $D$  个矩阵叠加得到的结果

## 张量积

用张量积运算规则，把矩阵  $C$  写成一系列张量积之和：

$$\begin{aligned} C &= a_1 \otimes (b^{(1)})^T + a_2 \otimes (b^{(2)})^T + \cdots + a_D \otimes (b^{(D)})^T \\ &= \sum_{i=1}^D a_i \otimes (b^{(i)})^T \end{aligned} \quad (25)$$

请大家格外注意上式中的转置。

这是一个非常重要的矩阵乘法视角，它不仅仅是上一节常规视角的补充。在很多数据科学和机器学习算法中，这一视角扮演至关重要的角色。

## 热图示例

下面我们用具体数字和热图可视化矩阵乘法外积展开。图 7 所示为  $A$  和  $B$  矩阵乘法热图。

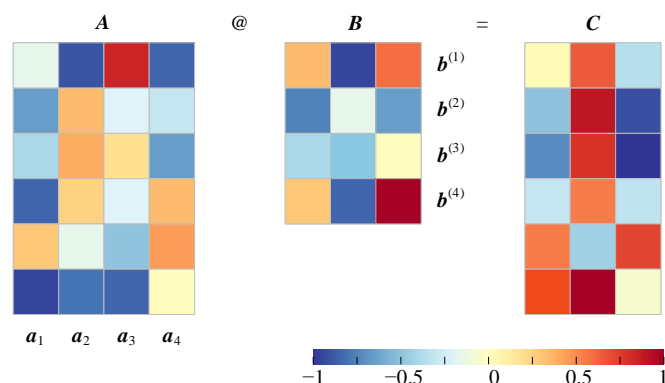


图 7. 矩阵乘法热图

将矩阵  $A$  拆解为一组列向量，矩阵  $B$  拆解为一组行向量。按照 (22) 计算矩阵乘法，得到如图 8 所示 4 幅热图。

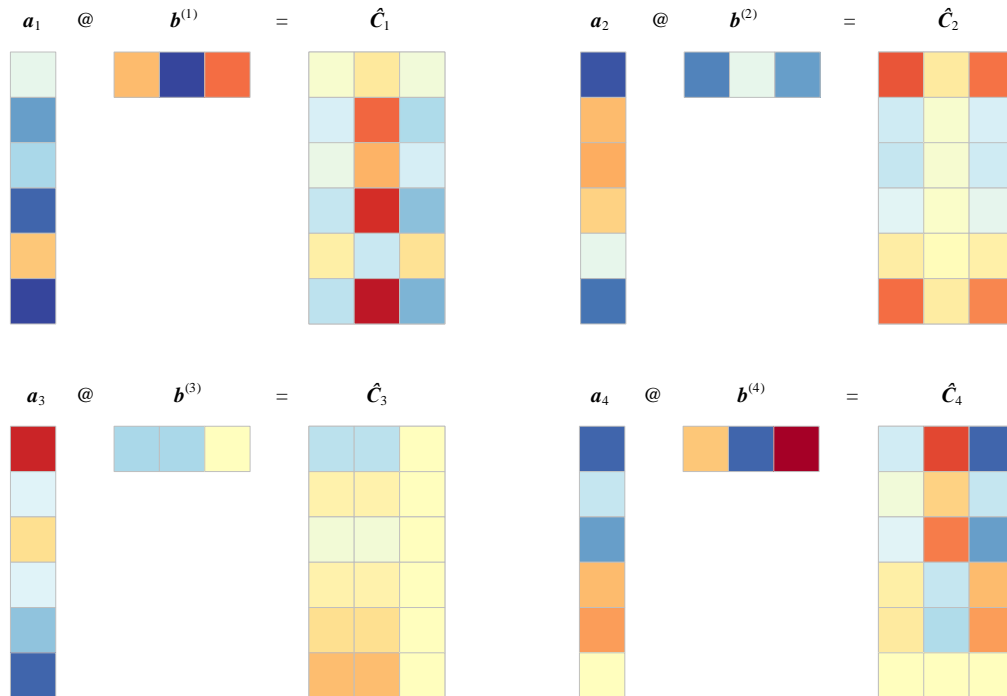


图 8. 列向量乘行向量结果热图

同样，也可以用张量积来计算得到这 4 幅热图，如图 9 所示。如图 10 所示，将这 4 幅热图叠加，我们可以得到乘积结果矩阵  $C$ 。这个思路对于奇异值分解和主成分分析非常重要。

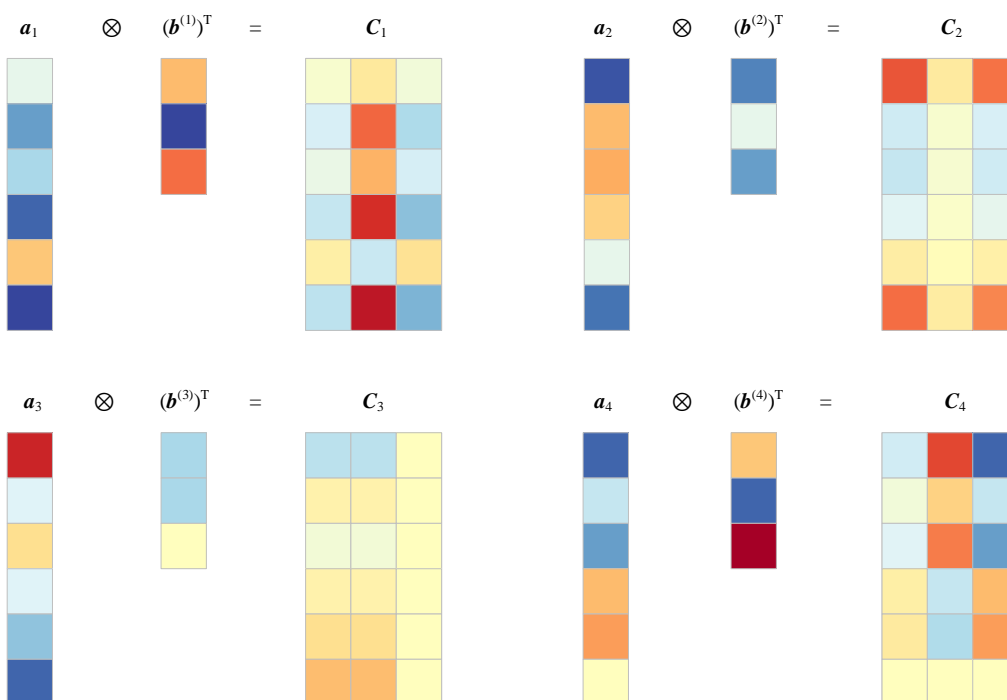


图 9. 张量积热图

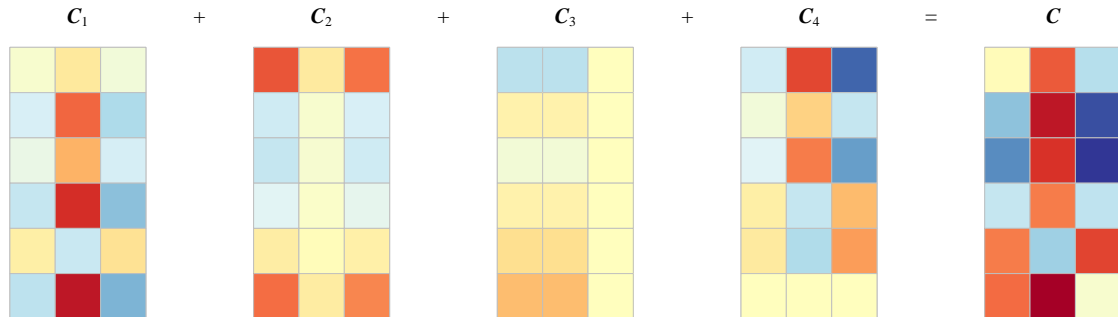
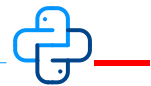


图 10. 四幅热图叠加

Bk4\_Ch5\_02.py 绘制图 10 的每幅热图。



```
# Bk4_Ch5_02.py

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

def plot_heatmap(x,title):

    fig, ax = plt.subplots()
    ax = sns.heatmap(x,
                    cmap='RdYlBu_r',
                    cbar_kws={"orientation": "horizontal"}, vmin=-1, vmax=1)
    ax.set_aspect("equal")
    plt.title(title)

# Generate matrices A and B
A = np.random.random_integers(0,40,size=(6,4))
A = A/20 - 1

B = np.random.random_integers(0,40,size=(4,3))
B = B/20 - 1

# visualize matrix A and B
plot_heatmap(A,'A')

plot_heatmap(B,'B')

# visualize A@B
C = A@B
plot_heatmap(C,'C = AB')

C_rep = np.zeros_like(C)

# reproduce C
for i in np.arange(4):
    C_i = A[:,i]@B[[i],:];
    title = 'C' + str(i + 1)
    plot_heatmap(C_i,title)
```

本 PDF 文件为作者草稿，发布目的为方便读者在移动终端学习，终稿内容以清华大学出版社纸质出版物为准。

版权归清华大学出版社所有，请勿商用，引用请注明出处。

代码及 PDF 文件下载：<https://github.com/Visualize-ML>

本书配套微课视频均发布在 B 站——生姜 DrGinger：<https://space.bilibili.com/513194466>

欢迎大家批评指教，本书专属邮箱：[jiang.visualize.ml@gmail.com](mailto:jiang.visualize.ml@gmail.com)

```
C_rep = C_rep + C_i
# Visualize reproduced C
plot_heatmap(C_rep, 'C reproduced')
```

## 5.4 矩阵乘法更多视角：分块多样化

本节介绍常见几种分块矩阵乘法形态，它们都可以视作观察矩阵乘法的不同视角。

### **B**切成列向量

$A$  和  $B$  矩阵相乘时，将  $B$  分割成列向量，这样  $AB$  结果为：

$$C = AB = A[b_1 \ b_2 \ \cdots \ b_m] = [Ab_1 \ Ab_2 \ \cdots \ Ab_m] \quad (26)$$

图 11 所示为上述运算示意图。请大家格外注意这个视角，我们将会在今后的投影运算中经常见到这种展开方法。

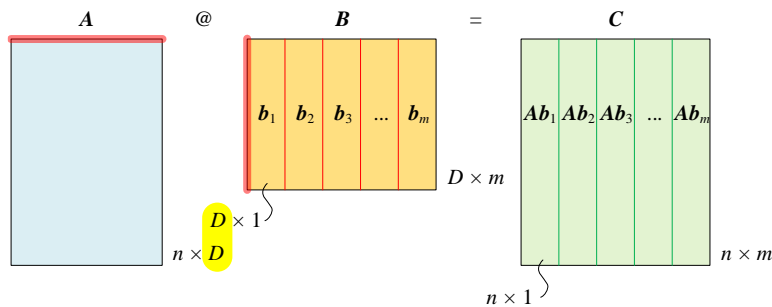


图 11.  $A$  和  $B$  矩阵相乘时，将  $B$  写成一排列向量

反方向来看，如果存在以下一系列矩阵乘法运算：

$$Ab_1 = c_1, \quad Ab_2 = c_2, \quad \cdots \quad Ab_m = c_m \quad (27)$$

且列向量  $b_1, b_2 \dots b_m$  的形状相同，则可以合成得到：

$$A \underbrace{[b_1 \ b_2 \ \cdots \ b_m]}_B = \underbrace{[c_1 \ c_2 \ \cdots \ c_m]}_C \quad (28)$$

### **B**左右切一刀

$B$  先左右切一刀后，矩阵  $A$  再左乘  $B$ ，乘积  $AB$  展开写成：

$$AB = A[B_1 \ B_2] = [AB_1 \ AB_2] \quad (29)$$

图 12 所示为上述运算示意图。

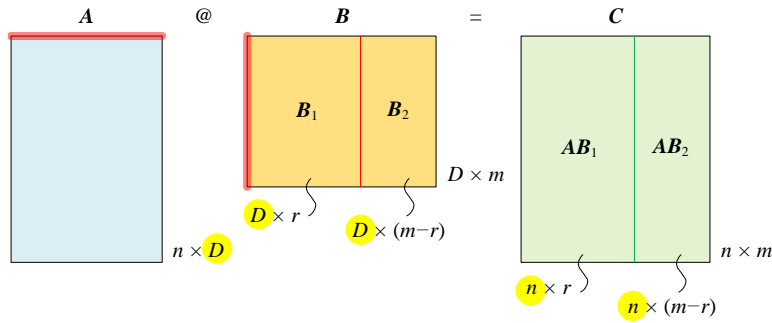


图 12. 将  $B$  左右切一刀再右乘  $A$

### $A$ 切成一排行向量

$A$  和  $B$  矩阵相乘，将  $A$  分割成一排行向量，乘积  $AB$  结果为：

$$C = AB = \begin{bmatrix} a^{(1)} \\ a^{(2)} \\ \vdots \\ a^{(n)} \end{bmatrix}_{n \times 1} @ B = \begin{bmatrix} a^{(1)}B \\ a^{(2)}B \\ \vdots \\ a^{(n)}B \end{bmatrix}_{n \times 1} \quad (30)$$

图 13 所示为上述运算示意图。此外，请大家也试着从矩阵乘法“合成”角度，逆向来看上述运算。

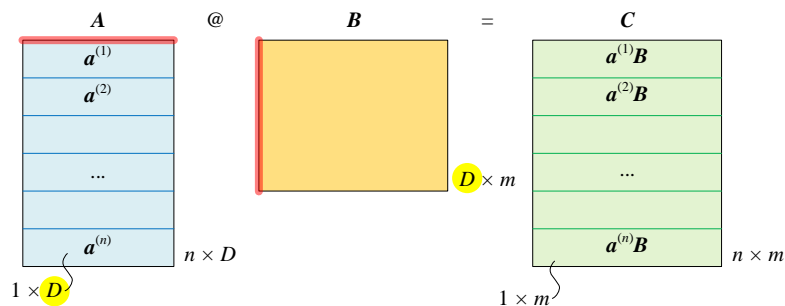


图 13.  $A$  和  $B$  矩阵相乘，将  $A$  分割成一排行向量

### $A$ 上下切一刀

将  $A$  先上下切一刀， $A$  再左乘  $B$ ，乘积  $AB$  结果为：

$$AB = \begin{bmatrix} A_1 \\ A_2 \end{bmatrix} B = \begin{bmatrix} A_1 B \\ A_2 B \end{bmatrix} \quad (31)$$

图 14 所示为上述运算示意图。

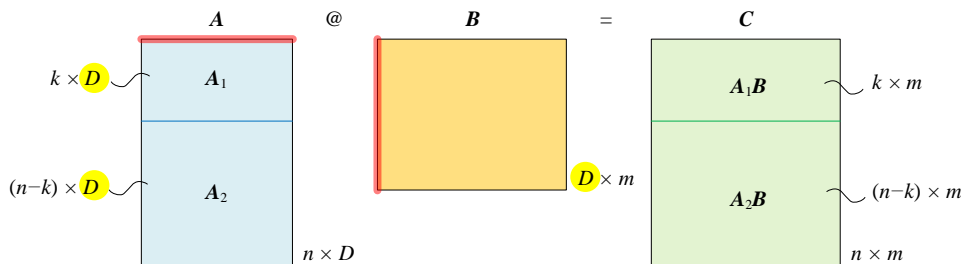


图 14. A 上下切一刀，再左乘 B

### A 上下切，B 左右切

上下分块的 A 乘左右分块的 B，乘积 AB 结果展开为：

$$AB = \begin{bmatrix} A_1 \\ A_2 \end{bmatrix} \begin{bmatrix} B_1 & B_2 \end{bmatrix} = \begin{bmatrix} A_1 B_1 & A_1 B_2 \\ A_2 B_1 & A_2 B_2 \end{bmatrix} \quad (32)$$

如图 15 所示，A<sub>1</sub> 和 A<sub>2</sub> 的列数还是 D，B<sub>1</sub> 和 B<sub>2</sub> 的行数也是 D。这个视角类似矩阵乘法的第一视角。我们可以把 A<sub>1</sub> 和 A<sub>2</sub> 视作矩阵 A 的两个元素，B<sub>1</sub> 和 B<sub>2</sub> 看成矩阵 B 的两个元素。

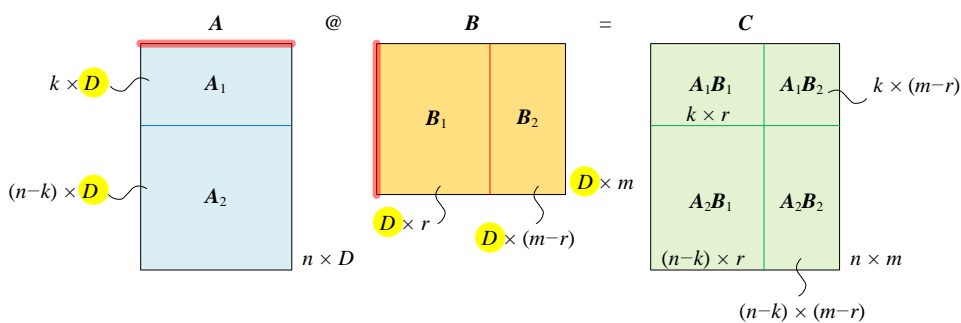


图 15. 上下分块的 A 乘左右分块的 B

### A 左右切，B 上下切

左右分块的 A 乘上下分块的 B，乘积 AB 结果展开为：

$$AB = \begin{bmatrix} A_1 & A_2 \end{bmatrix} \begin{bmatrix} B_1 \\ B_2 \end{bmatrix} = A_1 B_1 + A_2 B_2 \quad (33)$$

注意，如图 16 所示， $A_1$  列数等于  $B_1$  行数， $A_2$  列数等于  $B_2$  行数。这类似前面讲到的矩阵乘法的第二视角。

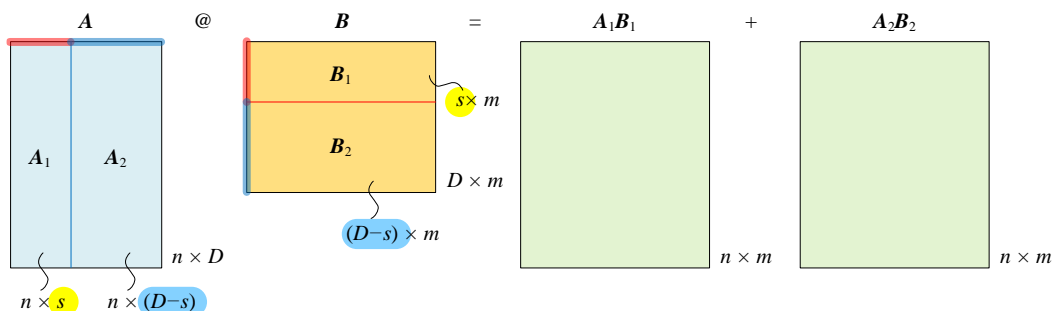


图 16. 左右分块的  $A$  乘以上下分块的  $B$

### $A$ 和 $B$ 都大卸四块

$A$  和  $B$  都上下左右分块，乘积  $AB$  结果为：

$$AB = \begin{bmatrix} A_{1,1} & A_{1,2} \\ A_{2,1} & A_{2,2} \end{bmatrix} \begin{bmatrix} B_{1,1} & B_{1,2} \\ B_{2,1} & B_{2,2} \end{bmatrix} = \begin{bmatrix} A_{1,1}B_{1,1} + A_{1,2}B_{2,1} & A_{1,1}B_{1,2} + A_{1,2}B_{2,2} \\ A_{2,1}B_{1,1} + A_{2,2}B_{2,1} & A_{2,1}B_{1,2} + A_{2,2}B_{2,2} \end{bmatrix} \quad (34)$$

注意分块的规则，如图 17 所示， $A_{1,1}$ 、 $A_{1,2}$ 、 $A_{2,1}$ 、 $A_{2,2}$  的列数分别等于  $B_{1,1}$ 、 $B_{2,1}$ 、 $B_{1,2}$ 、 $B_{2,2}$  的行数。图 17 中给出的矩阵乘法相当于两个  $2 \times 2$  矩阵相乘，结果  $C$  还是  $2 \times 2$ 。矩阵  $C$  的四个元素分别为  $C_{1,1}$ 、 $C_{1,2}$ 、 $C_{2,1}$ 、 $C_{2,2}$ 。这也相当于矩阵乘法的第一视角。

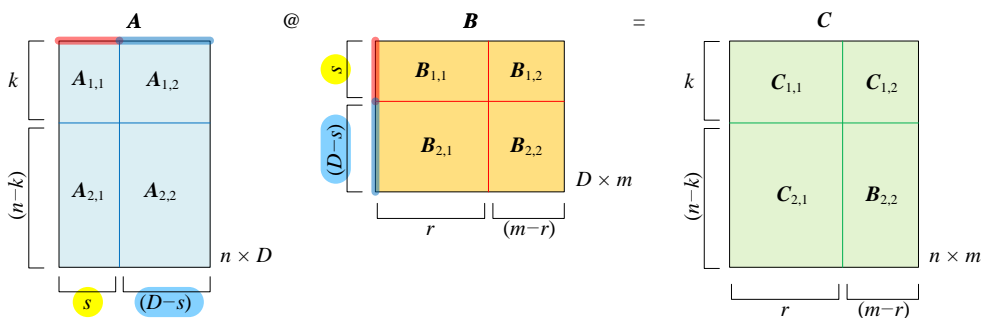
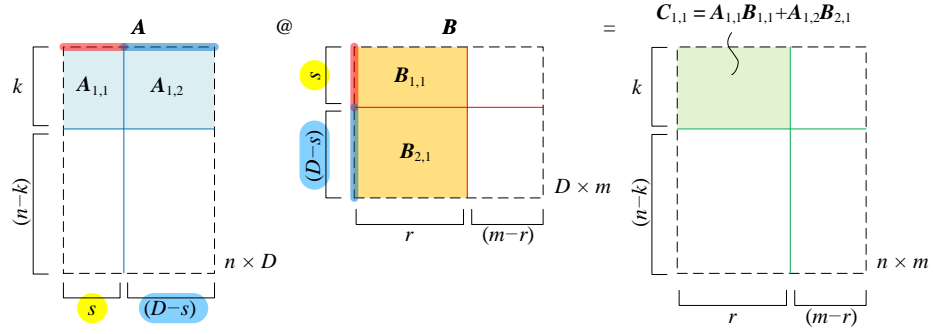
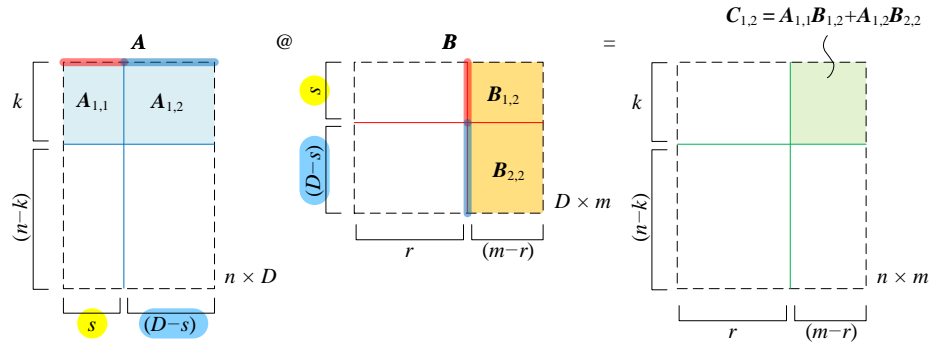
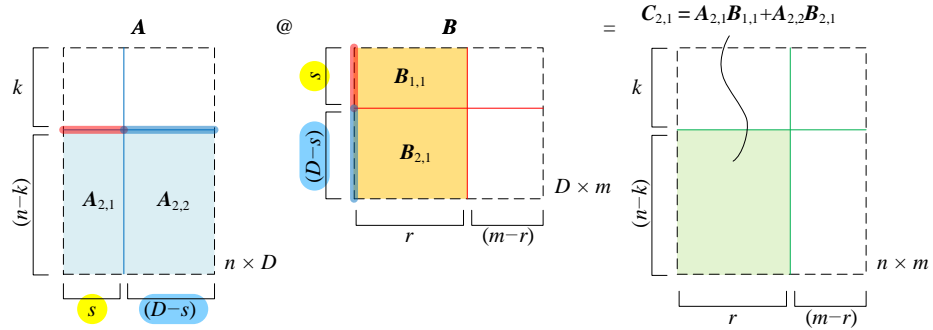
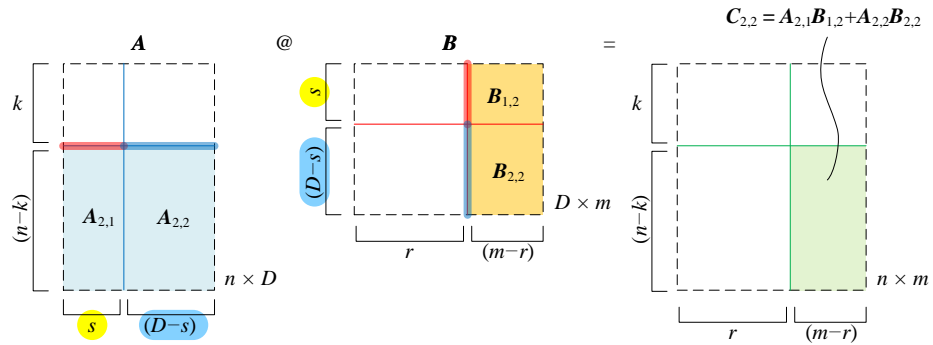


图 17.  $A$  和  $B$  都上下左右分块

图 18 到图 21 分别展示如何计算  $C_{1,1}$ 、 $C_{1,2}$ 、 $C_{2,1}$ 、 $C_{2,2}$ 。以  $C_{1,1}$  为例， $C_{1,1}$  的行数为  $A_{1,1}$  的行数， $C_{1,1}$  的列数为  $B_{1,1}$  的列数。



图 18. 计算  $C_{1,1}$ 图 19. 计算  $C_{1,2}$ 图 20. 计算  $C_{2,1}$ 图 21. 计算  $C_{2,2}$

## 逐步分块

还有一个办法解释图 17 分块矩阵乘法——逐步分块。

首先将  $A$  左右分块， $B$  上下分块， $AB$  乘积的结果如 (33)，乘积  $AB$  结果写成  $A_1B_1$  和  $A_2B_2$  相加，具体如图 22 所示。

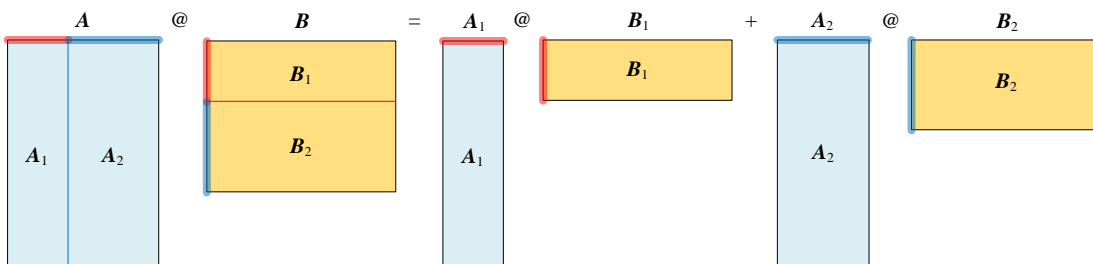


图 22. 首先将  $A$  左右分块， $B$  上下分块

然后再对  $A_1$  和  $A_2$  上下分块， $B_1$  和  $B_2$  左右分块：

$$A_1 = \begin{bmatrix} A_{1,1} \\ A_{2,1} \end{bmatrix}, \quad A_2 = \begin{bmatrix} A_{1,2} \\ A_{2,2} \end{bmatrix}, \quad B_1 = \begin{bmatrix} B_{1,1} & B_{1,2} \end{bmatrix}, \quad B_2 = \begin{bmatrix} B_{2,1} & B_{2,2} \end{bmatrix} \quad (35)$$

如图 23 所示， $A_1B_1$  按如下方式计算得到：

$$A_1B_1 = \begin{bmatrix} A_{1,1} \\ A_{2,1} \end{bmatrix} \begin{bmatrix} B_{1,1} & B_{1,2} \end{bmatrix} = \begin{bmatrix} A_{1,1}B_{1,1} & A_{1,1}B_{1,2} \\ A_{2,1}B_{1,1} & A_{2,1}B_{1,2} \end{bmatrix} \quad (36)$$

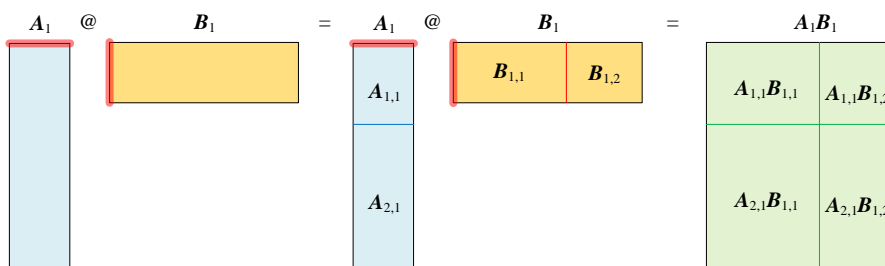
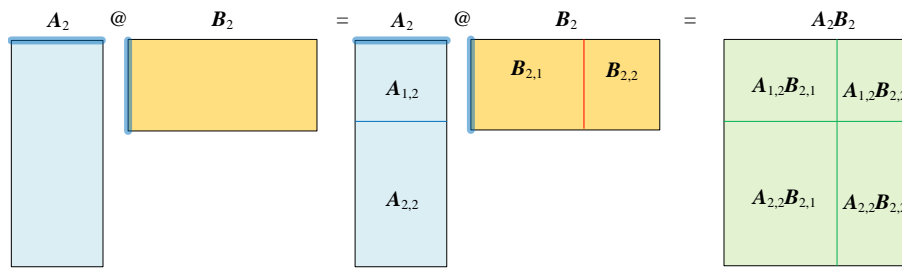


图 23. 计算  $A_1B_1$

同理，如图 24 所示，计算  $A_2B_2$ ：

$$A_2B_2 = \begin{bmatrix} A_{1,2} \\ A_{2,2} \end{bmatrix} \begin{bmatrix} B_{2,1} & B_{2,2} \end{bmatrix} = \begin{bmatrix} A_{1,2}B_{2,1} & A_{1,2}B_{2,2} \\ A_{2,2}B_{2,1} & A_{2,2}B_{2,2} \end{bmatrix} \quad (37)$$

图 24. 计算  $A_2 B_2$ 

(36) 和 (37) 相加就可以获得 (34) 结果：

$$\begin{bmatrix} A_{1,1}B_{1,1} & A_{1,1}B_{1,2} \\ A_{2,1}B_{1,1} & A_{2,1}B_{1,2} \end{bmatrix} + \begin{bmatrix} A_{1,2}B_{2,1} & A_{1,2}B_{2,2} \\ A_{2,2}B_{2,1} & A_{2,2}B_{2,2} \end{bmatrix} = \begin{bmatrix} A_{1,1}B_{1,1} + A_{1,2}B_{2,1} & A_{1,1}B_{1,2} + A_{1,2}B_{2,2} \\ A_{2,1}B_{1,1} + A_{2,2}B_{2,1} & A_{2,1}B_{1,2} + A_{2,2}B_{2,2} \end{bmatrix} \quad (38)$$

实际上，这个思路便是矩阵乘法第二视角。

这也足见矩阵乘法的灵活性，以及矩阵乘法两个视角的重要性。本书会在不同场合反复提到矩阵乘法的两个视角，以便强化认知。

## 5.5 分块矩阵的逆

如图 25 所示，将一个方阵分割成四个子块矩阵  $A$ 、 $B$ 、 $C$  和  $D$ ，其中  $A$  和  $D$  为方阵。当原矩阵可逆时，原矩阵的逆可以通过子块矩阵运算得到：

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix}^{-1} = \begin{bmatrix} (A - BD^{-1}C)^{-1} & -(A - BD^{-1}C)^{-1}BD^{-1} \\ -D^{-1}C(A - BD^{-1}C)^{-1} & D^{-1} + D^{-1}C(A - BD^{-1}C)^{-1}BD^{-1} \end{bmatrix} \quad (39)$$

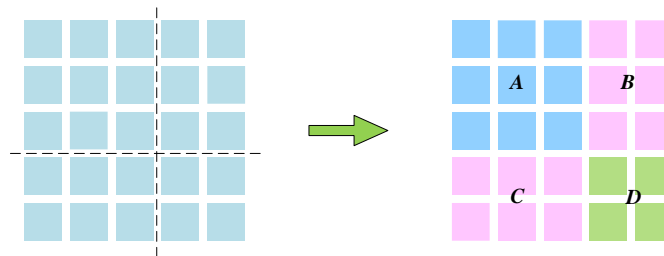


图 25. 分块矩阵求逆

令，

$$H = (A - BD^{-1}C)^{-1} \quad (40)$$

(39) 分块矩阵的逆可以写成：

$$\begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{C} & \mathbf{D} \end{bmatrix}^{-1} = \begin{bmatrix} \mathbf{H} & -\mathbf{HBD}^{-1} \\ -\mathbf{D}^{-1}\mathbf{CH} & \mathbf{D}^{-1} + \mathbf{D}^{-1}\mathbf{CHBD}^{-1} \end{bmatrix} \quad (41)$$

当然，这个分块矩阵的逆还有其他表达方式，我们这里不一一赘述。分块矩阵的逆将会用在协方差矩阵，特别是求解条件概率相关运算。

## 5.6 克罗内克积：矩阵张量积

**克罗内克积** (Kronecker product)，也叫矩阵张量积，是两个任意大小矩阵之间的运算，运算符为  $\otimes$ 。

矩阵  $\mathbf{A}$  的形状为  $n \times D$ ，矩阵  $\mathbf{B}$  的形状为  $p \times q$ ，那么  $\mathbf{A} \otimes \mathbf{B}$  的形状为  $np \times Dq$ ，结果为：

$$\mathbf{A} \otimes \mathbf{B} = \begin{bmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,D} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,D} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n,1} & a_{n,2} & \cdots & a_{n,D} \end{bmatrix} \otimes \mathbf{B} = \begin{bmatrix} a_{1,1}\mathbf{B} & a_{1,2}\mathbf{B} & \cdots & a_{1,D}\mathbf{B} \\ a_{2,1}\mathbf{B} & a_{2,2}\mathbf{B} & \cdots & a_{2,D}\mathbf{B} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n,1}\mathbf{B} & a_{n,2}\mathbf{B} & \cdots & a_{n,D}\mathbf{B} \end{bmatrix} \quad (42)$$

上式中每个  $a_{i,j}$  可以看成是缩放系数。

克罗内克积讲究顺序，一般情况  $\mathbf{A} \otimes \mathbf{B} \neq \mathbf{B} \otimes \mathbf{A}$ 。

请读者注意以下有关克罗内克积性质：

$$\begin{aligned} \mathbf{A} \otimes (\mathbf{B} + \mathbf{C}) &= \mathbf{A} \otimes \mathbf{B} + \mathbf{A} \otimes \mathbf{C} \\ (\mathbf{B} + \mathbf{C}) \otimes \mathbf{A} &= \mathbf{B} \otimes \mathbf{A} + \mathbf{C} \otimes \mathbf{A} \\ (k\mathbf{A}) \otimes \mathbf{B} &= \mathbf{A} \otimes (k\mathbf{B}) = k(\mathbf{A} \otimes \mathbf{B}) \\ (\mathbf{A} \otimes \mathbf{B}) \otimes \mathbf{C} &= \mathbf{A} \otimes (\mathbf{B} \otimes \mathbf{C}) \\ \mathbf{A} \otimes \mathbf{0} &= \mathbf{0} \otimes \mathbf{A} = \mathbf{0} \end{aligned} \quad (43)$$

### 举个例子

比如两个  $2 \times 2$  矩阵  $\mathbf{A}$  和  $\mathbf{B}$  的张量积为  $4 \times 4$  矩阵：

$$\begin{aligned} \mathbf{A} \otimes \mathbf{B} &= \begin{bmatrix} a_{1,1} & a_{1,2} \\ a_{2,1} & a_{2,2} \end{bmatrix} \otimes \begin{bmatrix} b_{1,1} & b_{1,2} \\ b_{2,1} & b_{2,2} \end{bmatrix} = \begin{bmatrix} a_{1,1}\mathbf{B} & a_{1,2}\mathbf{B} \\ a_{2,1}\mathbf{B} & a_{2,2}\mathbf{B} \end{bmatrix} \\ &= \begin{bmatrix} a_{1,1} \begin{bmatrix} b_{1,1} & b_{1,2} \\ b_{2,1} & b_{2,2} \end{bmatrix} & a_{1,2} \begin{bmatrix} b_{1,1} & b_{1,2} \\ b_{2,1} & b_{2,2} \end{bmatrix} \\ a_{2,1} \begin{bmatrix} b_{1,1} & b_{1,2} \\ b_{2,1} & b_{2,2} \end{bmatrix} & a_{2,2} \begin{bmatrix} b_{1,1} & b_{1,2} \\ b_{2,1} & b_{2,2} \end{bmatrix} \end{bmatrix} = \begin{bmatrix} a_{1,1}b_{1,1} & a_{1,1}b_{1,2} & a_{1,2}b_{1,1} & a_{1,2}b_{1,2} \\ a_{1,1}b_{2,1} & a_{1,1}b_{2,2} & a_{1,2}b_{2,1} & a_{1,2}b_{2,2} \\ a_{2,1}b_{1,1} & a_{2,1}b_{1,2} & a_{2,2}b_{1,1} & a_{2,2}b_{1,2} \\ a_{2,1}b_{2,1} & a_{2,1}b_{2,2} & a_{2,2}b_{2,1} & a_{2,2}b_{2,2} \end{bmatrix} \end{aligned} \quad (44)$$

`numpy.kron()` 可以用来计算矩阵张量积。

### 和向量张量积的关系

克罗内克积相当于向量张量积的推广；反过来，向量张量积也可以看做克罗内克积的特例。

但两者稍有不同，为了方便计算，两个列向量的张量积定义为  $\mathbf{a} \otimes \mathbf{b} = \mathbf{a}\mathbf{b}^T$ ，也就是：

$$\mathbf{a} \otimes \mathbf{b} = \begin{bmatrix} a_1 \\ a_2 \end{bmatrix} \otimes \mathbf{b} = \begin{bmatrix} a_1 \mathbf{b}^T \\ a_2 \mathbf{b}^T \end{bmatrix} \quad (45)$$

请读者注意上式中的转置运算。而 (42) 中不存在转置。

举个例子， $\mathbf{A}$  和  $\mathbf{B}$  分别为：

$$\mathbf{A} = \begin{bmatrix} -1 & 1 \\ 0.7 & -0.4 \end{bmatrix}, \quad \mathbf{B} = \begin{bmatrix} 0.5 & -0.6 \\ -0.8 & 0.3 \end{bmatrix} \quad (46)$$

$\mathbf{A}$  和  $\mathbf{B}$  的张量积  $\mathbf{A} \otimes \mathbf{B}$  为：

$$\begin{aligned} \mathbf{A} \otimes \mathbf{B} &= \begin{bmatrix} -1 & 1 \\ 0.7 & -0.4 \end{bmatrix} \otimes \begin{bmatrix} 0.5 & -0.6 \\ -0.8 & 0.3 \end{bmatrix} \\ &= \begin{bmatrix} -1 \times \begin{bmatrix} 0.5 & -0.6 \\ -0.8 & 0.3 \end{bmatrix} & 1 \times \begin{bmatrix} 0.5 & -0.6 \\ -0.8 & 0.3 \end{bmatrix} \\ 0.7 \times \begin{bmatrix} 0.5 & -0.6 \\ -0.8 & 0.3 \end{bmatrix} & -0.4 \times \begin{bmatrix} 0.5 & -0.6 \\ -0.8 & 0.3 \end{bmatrix} \end{bmatrix} \quad (47) \end{aligned}$$

图 26 所示为上述计算的热图。

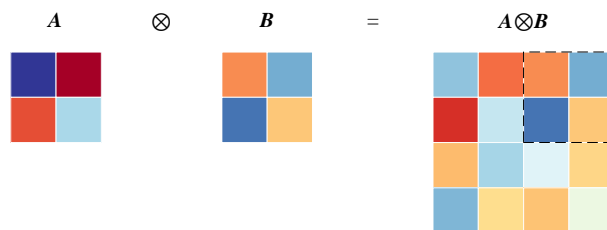
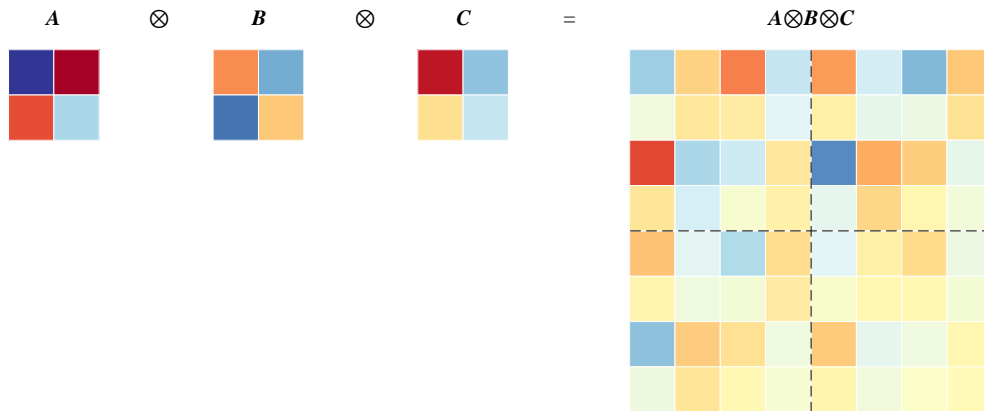


图 26.  $\mathbf{A}$  和  $\mathbf{B}$  的张量积  $\mathbf{A} \otimes \mathbf{B}$

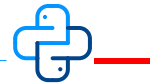
再给出第三个  $2 \times 2$  矩阵  $\mathbf{C}$ ：

$$\mathbf{C} = \begin{bmatrix} -1 & 1 \\ 0.7 & -0.4 \end{bmatrix} \quad (48)$$

在  $\mathbf{A} \otimes \mathbf{B} \otimes \mathbf{C}$  的张量积的运算如图 27 所示。

图 27.  $A$ 、 $B$ 、 $C$  的张量积  $A \otimes B \otimes C$ 

Bk4\_Ch5\_03.py 绘制图 26。请读者自行绘制图 27。



```
# Bk4_Ch5_03.py

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

A = np.array([[ -1,  1],
              [ 0.7, -0.4]])

B = np.array([[ 0.5, -0.6],
              [-0.8,  0.3]])

A_kron_B = np.kron(A, B)

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

plt.sca(axs[0])
ax = sns.heatmap(A, cmap='RdYlBu_r', vmax = 1, vmin = -1,
                 cbar_kws={"orientation": "horizontal"})
ax.set_aspect("equal")
plt.title('A')

plt.sca(axs[1])
plt.title('$\otimes$')
plt.axis('off')

plt.sca(axs[2])
ax = sns.heatmap(B, cmap='RdYlBu_r', vmax = 1, vmin = -1,
                 cbar_kws={"orientation": "horizontal"})
ax.set_aspect("equal")
plt.title('B')

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

plt.sca(axs[4])
ax = sns.heatmap(A_kron_B, cmap='RdYlBu_r', vmax = 1, vmin = -1,
                 cbar_kws={"orientation": "horizontal"})
ax.set_aspect("equal")
```

本 PDF 文件为作者草稿，发布目的为方便读者在移动终端学习，终稿内容以清华大学出版社纸质出版物为准。

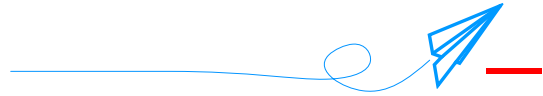
版权归清华大学出版社所有，请勿商用，引用请注明出处。

代码及 PDF 文件下载：<https://github.com/Visualize-ML>

本书配套微课视频均发布在 B 站——生姜 DrGinger：<https://space.bilibili.com/513194466>

欢迎大家批评指教，本书专属邮箱：[jiang.visualize.ml@gmail.com](mailto:jiang.visualize.ml@gmail.com)

```
| plt.title('C')
```



本章介绍的几种矩阵分块运算都很重要，找出四幅图去总结主要内容，也很难。

虽然分块矩阵运算，特别是乘法运算，让人看的眼花缭乱；但是，万变不离其宗，大家首先要把握的是矩阵乘法规则，这是根本。其次，同等重要的就是，我们在本书中反复强调的——矩阵乘法两个视角。

此外，大家注意矩阵乘法的“合成”，也就是分块矩阵乘法的逆向运算。掌握这个逆向思维方式有助于理解和简化很多运算，大家将会在本书后文数据投影映射中看到大量实例。