

作者	生姜 DrGinger
脚本	生姜 DrGinger
视频	崔崔 CuiCui
开源学习资源	<a href="https://github.com/Visualize-ML">https://github.com/Visualize-ML</a>
平台	<a href="https://www.youtube.com/@DrGinger_Jiang">https://www.youtube.com/@DrGinger_Jiang</a> <a href="https://space.bilibili.com/3546865719052873">https://space.bilibili.com/3546865719052873</a> <a href="https://space.bilibili.com/513194466">https://space.bilibili.com/513194466</a>

## 2.4 矩阵乘法



### 本节你将掌握的核心技能：

- ▶ 矩阵乘法：矩阵乘法将两个矩阵相乘，生成一个新矩阵。
- ▶ 形状匹配规则：左矩阵列数需等于右矩阵行数，乘积形状由左行、右列决定。
- ▶ NumPy 执行矩阵乘法：可使用 `@`、`np.matmul()` 实现。
- ▶ 乘积元素：由对应行向量与列向量的点积计算得到。
- ▶ 矩阵乘法一般不满足交换律。

矩阵乘法可能是线性代数的最重要的运算，没有之一；矩阵乘法表示两个矩阵相乘生成新矩阵。矩阵乘法中，请大家特别注意形状匹配；此外，矩阵乘法一般不满足交换律。

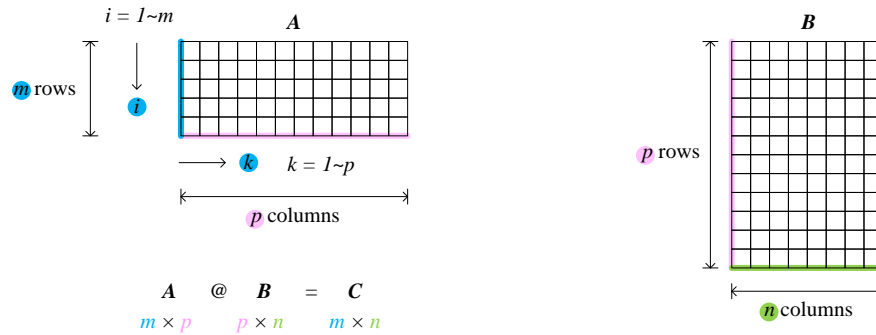
鉴于矩阵乘法的重要性，本册留出几节从各个角度介绍矩阵乘法；本节，让我们初探矩阵乘法。

### 什么是矩阵乘法？

**矩阵乘法** (matrix multiplication) 是线性代数中的一种基本运算，用于将两个矩阵相乘，生成一个新的矩阵。

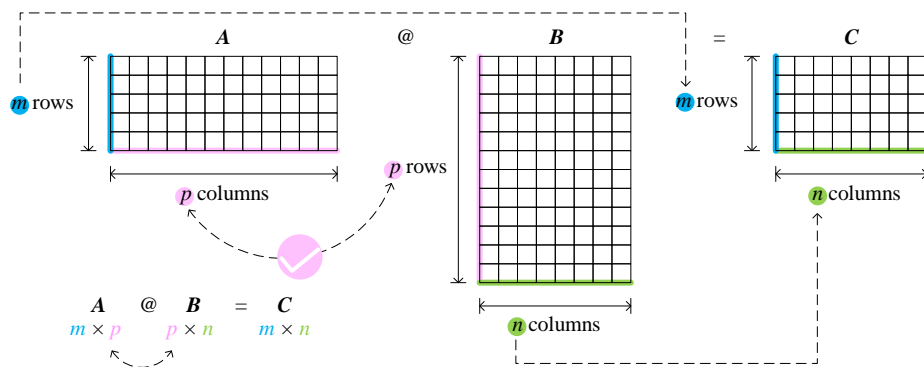
⚠ 注意，有些时候矩阵乘法的结果也可以是  $1 \times 1$  矩阵，我们将其视作标量。大家马上就会看到。

假设有两个矩阵 **A** 和 **B**。如图 1 所示，矩阵 **A** 的形状为  $m \times p$ ，即 **A** 有  $m$  行、 $p$  列；矩阵 **B** 形状为  $p \times n$ ，即 **B** 有  $p$  行、 $n$  列。

图 1. 矩阵  $A$  和  $B$  的形状

那么，如图 2 所示，矩阵  $A$  和  $B$  的乘积  $C = AB = A @ B$  是一个形状为  $m \times n$  的矩阵。

**⚠ 注意**，本书也会将矩阵乘法  $AB$  写成  $A @ B$ ，这种写法是为了与 NumPy 等 Python 科学计算库的语法保持一致。

图 2. 矩阵乘积  $A @ B$ 

从形状匹配角度来看，矩阵乘法  $AB = A @ B$  要求**左侧**矩阵  $A$  的**列数** ( $p$ )，等于，**右侧**矩阵  $B$  的**行数** ( $p$ )；否则，无法进行矩阵乘法运算。

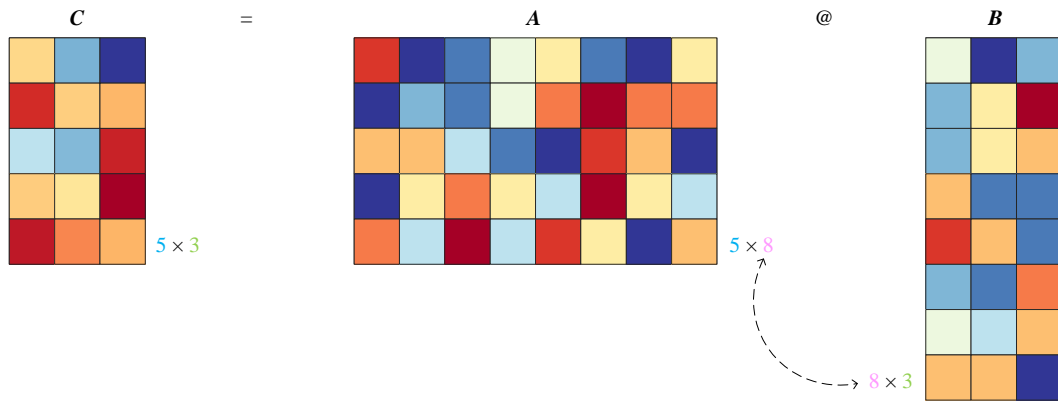
矩阵  $A$ 、 $B$  的乘积  $C$  继承了  $A$  的**行数** ( $m$ )、 $B$  的**列数** ( $n$ )；而  $A$  的**列数** ( $p$ )、 $B$  的**行数** ( $p$ ) 相当于“消去”。

建议大家这样记忆——从矩阵形状来看， $(m \times p) @ (p \times n)$  夹在中间的 ( $p$ ) 被“消去”。

## NumPy 计算矩阵乘法

让我们看看如何用 NumPy 完成上述矩阵乘法计算。在 NumPy 中，矩阵乘法可以通过简单的 `@` 运算符、`np.matmul()` 或 `np.dot()` 函数实现。代码 1 采用 `@` 运算符完成矩阵乘法运算，并用热图 (如图 3) 所示完成可视化。

从矩阵形状来看， $(5 \times 8) @ (8 \times 3)$  夹在中间的 ( $8$ ) 被“消去”，矩阵乘法结果的形状为  $(5 \times 3)$ 。

图 3. 热图可视化矩阵乘积  $A @ B$ 

下面聊聊代码 1 中关键语句。

大家已经很熟悉 <sup>a</sup> 这三句，设定随机数种子后，我们用 `numpy.random.randint()` 创建了两个矩阵。

<sup>b</sup> 用 `@` 完成矩阵乘法，这一句相当于 `np.matmul(A, B)`。

<sup>c</sup> 这行代码创建了一个 Matplotlib 图形对象 `fig`，并在其中生成一个包含 1 行、5 列 5 个子图的轴 `axs` 变量数组。其中，`figsize=(8, 3)` 指定了图形的大小，宽度为 8 英寸，高度为 3 英寸。

<sup>d</sup> 激活第一个子图轴 `axs[0]` 为当前轴，以便作图。

<sup>e</sup> 用 `seaborn.heatmap()` 绘制热图。

其中，`C` 是要可视化的矩阵（二维数组）。

`cmap = 'RdYlBu_r'` 指定了颜色映射方案。

`xticklabels=[]` 和 `yticklabels=[]` 用于隐藏 x 轴和 y 轴的刻度标签。

`cbar_kws={"orientation": "horizontal"}` 指定颜色条 (color bar) 为水平放置。

<sup>f</sup> 这行代码确保子图的长宽比例相等。

<sup>g</sup> 用 `matplotlib.pyplot` (简作 `plt`) 中的 `title()` 给子图增加标题。

<sup>h</sup> 隐藏该子图的坐标轴，使其只显示等号，而不显示边框和刻度。

代码 1. 热图可视化矩阵乘法 |  LA\_Ch02\_04\_01.ipynb

```

## 初始化
import numpy as np
import seaborn as sns
from matplotlib import pyplot as plt

## 创建矩阵A和B
a np.random.seed(88)
A = np.random.randint(0, 10, (5, 8))
B = np.random.randint(0, 10, (8, 3))

## 矩阵乘法 A@B
b C = A @ B

## 可视化矩阵乘法
c fig, axs = plt.subplots(1, 5, figsize=(8, 3))

d plt.sca(axs[0])
e ax = sns.heatmap(C, cmap='RdYlBu_r',
                  xticklabels = [], yticklabels = [],
                  cbar_kws={"orientation": "horizontal"})
f ax.set_aspect("equal")
g plt.title('C')

plt.sca(axs[1])
plt.title('=') # 绘制等号
h plt.axis('off')

plt.sca(axs[2])
ax = sns.heatmap(A, cmap='RdYlBu_r',
                  xticklabels = [], yticklabels = [],
                  cbar_kws={"orientation": "horizontal"})
ax.set_aspect("equal")
plt.title('A')

plt.sca(axs[3])
plt.title('@') # 绘制乘号
plt.axis('off')

plt.sca(axs[4])
ax = sns.heatmap(B, cmap='RdYlBu_r',
                  xticklabels = [], yticklabels = [],
                  cbar_kws={"orientation": "horizontal"})
ax.set_aspect("equal")
plt.title('B')

```



## 计算每个元素

下面，让我们看看如何具体计算矩阵  $C$  的每个元素。

矩阵  $C$  的第  $i$  行、第  $j$  列元素  $c_{i,j}$  的计算方法为

$$c_{i,j} = a_{i,1}b_{1,j} + a_{i,2}b_{2,j} + \cdots + a_{i,k}b_{k,j} + \cdots + a_{i,p}b_{p,j} = \sum_{k=1}^p a_{i,k}b_{k,j} \quad (1)$$

如图 4 所示，简单来说， $c_{i,j}$  是矩阵  $A$  的第  $i$  行  $a^{(i)}$  与矩阵  $B$  的第  $j$  列  $b_j$  对应元素相乘后的累加和。

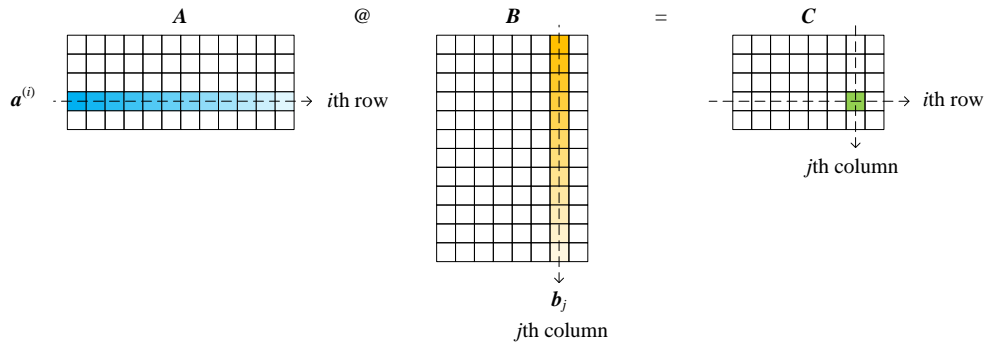


图 4. 计算矩阵乘积  $C = AB$  的每个元素

大家应该注意到， $a^{(i)} @ b_j$  也是一个矩阵乘法运算，即

$$c_{i,j} = a^{(i)} @ b_j \quad (2)$$

如图 5 所示，形状为  $1 \times p$  行向量  $a^{(i)}$  和形状为  $p \times 1$  列向量  $b_j$  乘积结果为  $1 \times 1$  矩阵；而  $1 \times 1$  矩阵只有一个元素，相当于一个标量。

⚠ 上式也告诉我们矩阵乘法的结果可以是一个标量 ( $1 \times 1$  矩阵)！这一点特别值得我们注意。

我们也可以把 (2) 写成向量内积形式

$$c_{i,j} = (a^{(i)})^T \cdot b_j \quad (3)$$

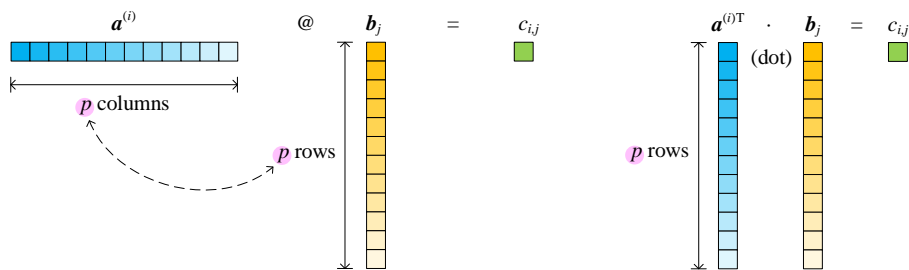


图 5. 矩阵乘积  $a^{(i)} @ b_j = c_{i,j}$

而矩阵乘积  $C = AB$  相当于完成了  $m \times n$  个矩阵乘法，也相当于完成了  $m \times n$  个向量内积。

可以这么理解，矩阵乘法  $AB$  不过是  $A$  的行向量与  $B$  的列向量的一组点积 (dot products)。

## 举个例子

下面举个简单例子计算矩阵乘法。

给定如下矩阵  $A$  和  $B$

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

如图 6 所示，矩阵  $A$  的形状是  $2 \times 3$ ，矩阵  $B$  的形状是  $3 \times 2$ ，因此结果矩阵  $C = AB$  的形状是  $2 \times 2$ 。

从矩阵形状来看， $(2 \times 3) @ (3 \times 2)$  夹在中间的  $(3)$  被“消去”，矩阵乘法结果的形状为  $(2 \times 2)$ 。

$$\begin{array}{c} A \\ \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix} \end{array} @ \begin{array}{c} B \\ \begin{bmatrix} 1 & 4 \\ 2 & 5 \\ 3 & 6 \end{bmatrix} \end{array} = \begin{array}{c} C \\ \begin{bmatrix} 14 & 32 \\ 32 & 77 \end{bmatrix} \end{array}$$

图 6. 矩阵乘积  $AB$ ； $A$  的形状是  $2 \times 3$ ， $B$  的形状是  $3 \times 2$

下面，计算矩阵乘法  $C = AB$  的每个元素

$$\begin{cases} c_{1,1} = a_{1,1} \cdot b_{1,1} + a_{1,2} \cdot b_{2,1} + a_{1,3} \cdot b_{3,1} = 1 \times 1 + 2 \times 2 + 3 \times 3 = 1 + 4 + 9 = 14 \\ c_{1,2} = a_{1,1} \cdot b_{1,2} + a_{1,2} \cdot b_{2,2} + a_{1,3} \cdot b_{3,2} = 1 \times 4 + 2 \times 5 + 3 \times 6 = 4 + 10 + 18 = 32 \\ c_{2,1} = a_{2,1} \cdot b_{1,1} + a_{2,2} \cdot b_{2,1} + a_{2,3} \cdot b_{3,1} = 4 \times 1 + 5 \times 2 + 6 \times 3 = 4 + 10 + 18 = 32 \\ c_{2,2} = a_{2,1} \cdot b_{1,2} + a_{2,2} \cdot b_{2,2} + a_{2,3} \cdot b_{3,2} = 4 \times 4 + 5 \times 5 + 6 \times 6 = 16 + 25 + 36 = 77 \end{cases} \quad (5)$$

图 7 所示为如何用矩阵乘法计算矩阵  $C$  的每个元素。

$$\begin{array}{l} \begin{array}{c} a^{(1)} \\ \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix} \end{array} @ \begin{array}{c} b_1 \\ \begin{bmatrix} 1 & 4 \\ 2 & 5 \\ 3 & 6 \end{bmatrix} \end{array} = \begin{array}{c} c_{1,1} \\ \begin{bmatrix} 14 & 32 \\ 32 & 77 \end{bmatrix} \end{array} \quad \begin{array}{c} a^{(1)} \\ \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix} \end{array} @ \begin{array}{c} b_2 \\ \begin{bmatrix} 1 & 4 \\ 2 & 5 \\ 3 & 6 \end{bmatrix} \end{array} = \begin{array}{c} c_{1,2} \\ \begin{bmatrix} 14 & 32 \\ 32 & 77 \end{bmatrix} \end{array} \\ \\ \begin{array}{c} a^{(2)} \\ \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix} \end{array} @ \begin{array}{c} b_1 \\ \begin{bmatrix} 1 & 4 \\ 2 & 5 \\ 3 & 6 \end{bmatrix} \end{array} = \begin{array}{c} c_{2,1} \\ \begin{bmatrix} 14 & 32 \\ 32 & 77 \end{bmatrix} \end{array} \quad \begin{array}{c} a^{(2)} \\ \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix} \end{array} @ \begin{array}{c} b_2 \\ \begin{bmatrix} 1 & 4 \\ 2 & 5 \\ 3 & 6 \end{bmatrix} \end{array} = \begin{array}{c} c_{2,2} \\ \begin{bmatrix} 14 & 32 \\ 32 & 77 \end{bmatrix} \end{array} \end{array}$$

图 7. 计算矩阵乘积  $AB$  的每个元素； $A$  的形状是  $2 \times 3$ ， $B$  的形状是  $3 \times 2$



LA\_02\_04\_02.ipynb 计算上述矩阵乘法的每个元素，请大家自行学习。

## 不满足交换律

此外，大家应该已经注意到矩阵乘法的左右位置非常重要，不能随意交换。这是因为矩阵乘法一般情况下**不满足交换律** (non-commutative)。

首先， $A @ B$  满足矩阵乘法运算法则，不代表  $B @ A$  也同样满足矩阵乘法运算法则 (如图 8 所示)。

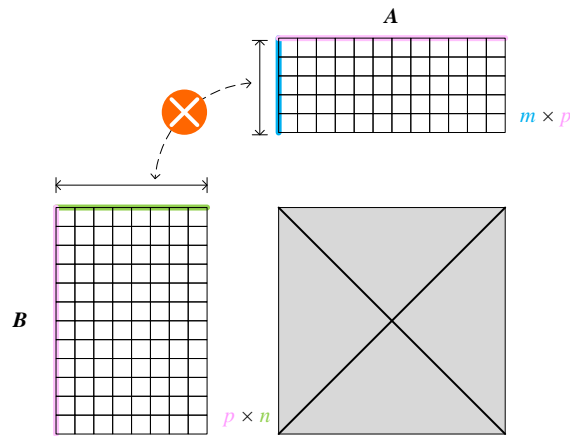


图 8.  $B @ A$  不满足矩阵乘法运算原则

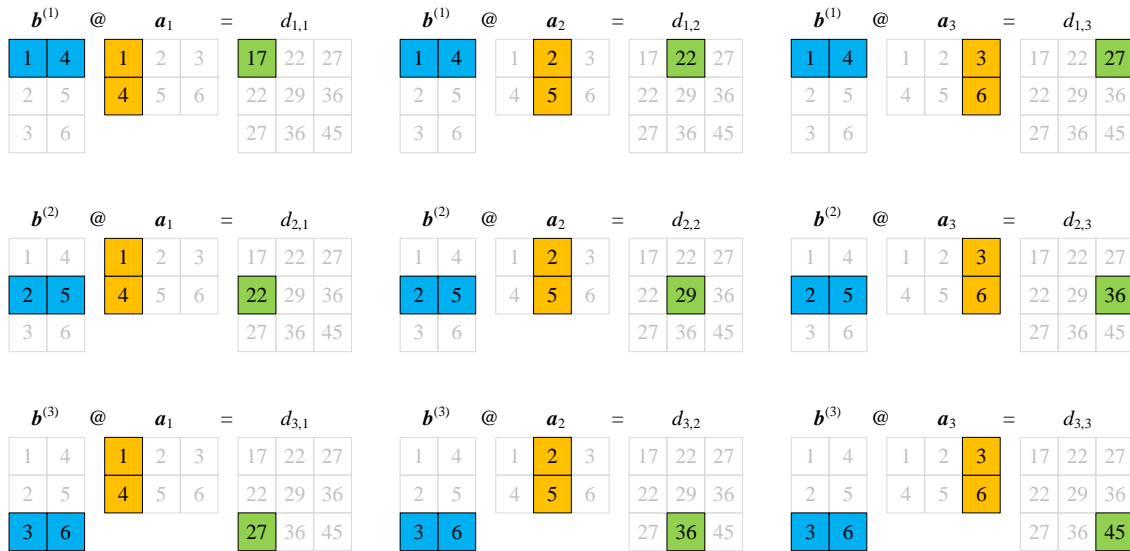
即便， $B @ A$  也满足矩阵乘法运算法则，也不意味着  $A @ B$  和  $B @ A$  的结果相同。

比如，下例中的  $B @ A$  结果显然不同于  $A @ B$

$$D = B @ A = \begin{bmatrix} 1 & 4 \\ 2 & 5 \\ 3 & 6 \end{bmatrix} @ \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix} = \begin{bmatrix} 17 & 22 & 27 \\ 22 & 29 & 36 \\ 27 & 36 & 45 \end{bmatrix} \quad (6)$$

从矩阵形状来看， $(3 \times 2) @ (2 \times 3)$  夹在中间的  $(2)$  被“消去”，矩阵乘法结果的形状为  $(3 \times 3)$ 。

图 9 展示如何计算  $B @ A$  的每个元素。

图 9. 计算矩阵乘法  $B @ A$  的每个元素； $A$  的形状是  $2 \times 3$ ， $B$  的形状是  $3 \times 2$ 

**?** 请大家自行手算上述矩阵乘法。大家可能已经发现， $A @ B$  和  $B @ A$  都是对称矩阵，这是巧合？还是另有原因？

## 自定义 Python 函数计算矩阵乘法

代码 2 自定义 Python 函数计算矩阵乘法。

**a** 定义一个自定义函数，名字叫 `matrix_multiplication`，这个函数可以被反复调用。它有两个输入参数， $A$  和  $B$ ，你可以传入两个矩阵，返回矩阵乘法结果。

**b** 先取出矩阵  $A$  的尺寸大小，`.shape` 是 NumPy 数组自带的属性，会返回矩阵的“行数”和“列数”。然后取出矩阵  $B$  的尺寸。`p_B` 表示  $B$  的行数，`n` 表示  $B$  的列数。这样我们就知道两个输入矩阵的大小了。

**c** 是一个条件判断，如果矩阵  $A$  的列数 `p_A` 和矩阵  $B$  的行数 `p_B` 不相等，就不满足矩阵乘法条件。如果条件成立，也就是两个矩阵形状不匹配，那就报错，提醒用户不能做乘法。`raise` 是抛出一个错误的意义，`ValueError` 是错误类型，这种写法是 Python 报错机制的一部分，用来保护程序不崩溃。

**d** 用 `numpy.zeros()` 创建一个全 0 矩阵  $C$ ，它的形状是  $m$  行  $n$  列，和最终的结果矩阵一样大。

然后，我们利用三层 `for` 循环计算矩阵乘法的每个元素，如图 10 所示。

**e** 是第一层 `for` 循环， $i$  是用来遍历矩阵  $A$  的行的。`range(m)` 会产生从 0 到  $m-1$  的整数，用来一个一个访问  $A$  每一行。

**f** 是第二层 `for` 循环， $j$  是用来遍历矩阵  $B$  的列。`range(n)` 会产生从 0 到  $n-1$  的整数，用来一个一个访问  $B$  每一列。

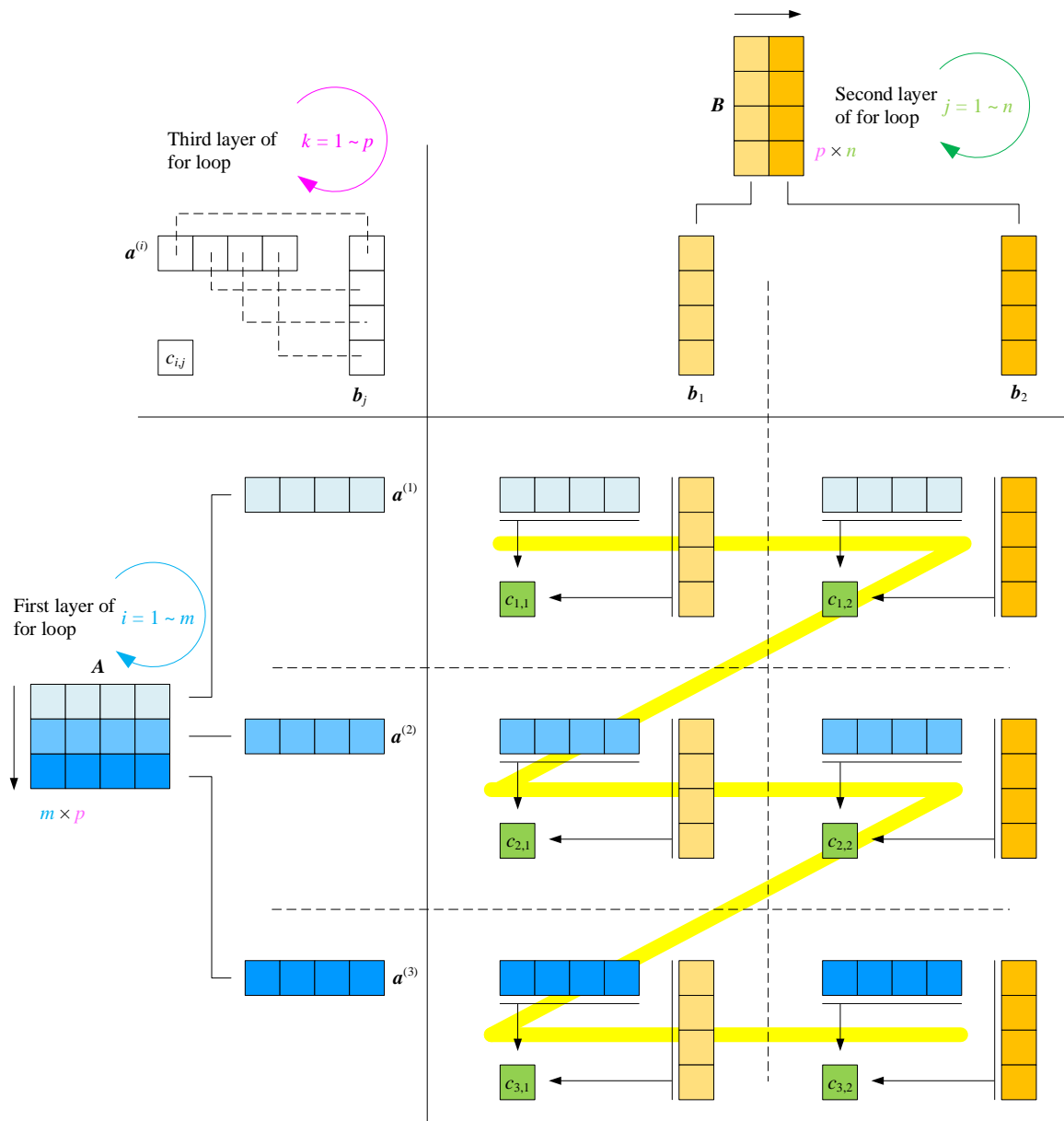
**g** 是第三层 `for` 循环， $k$  用来遍历  $A$  的列数，也就是  $B$  的行数。它是矩阵乘法中用来对应元素相乘再累加的过程。`C[i, j] += A[i, k] * B[k, j]` 取出  $A$  的第  $i$  行第  $k$  个元素和  $B$  的第  $k$  行第  $j$  个元素，把它们相乘后，累加到 `C[i, j]`。


这三层 `for` 循环的关系如图 10 所示。

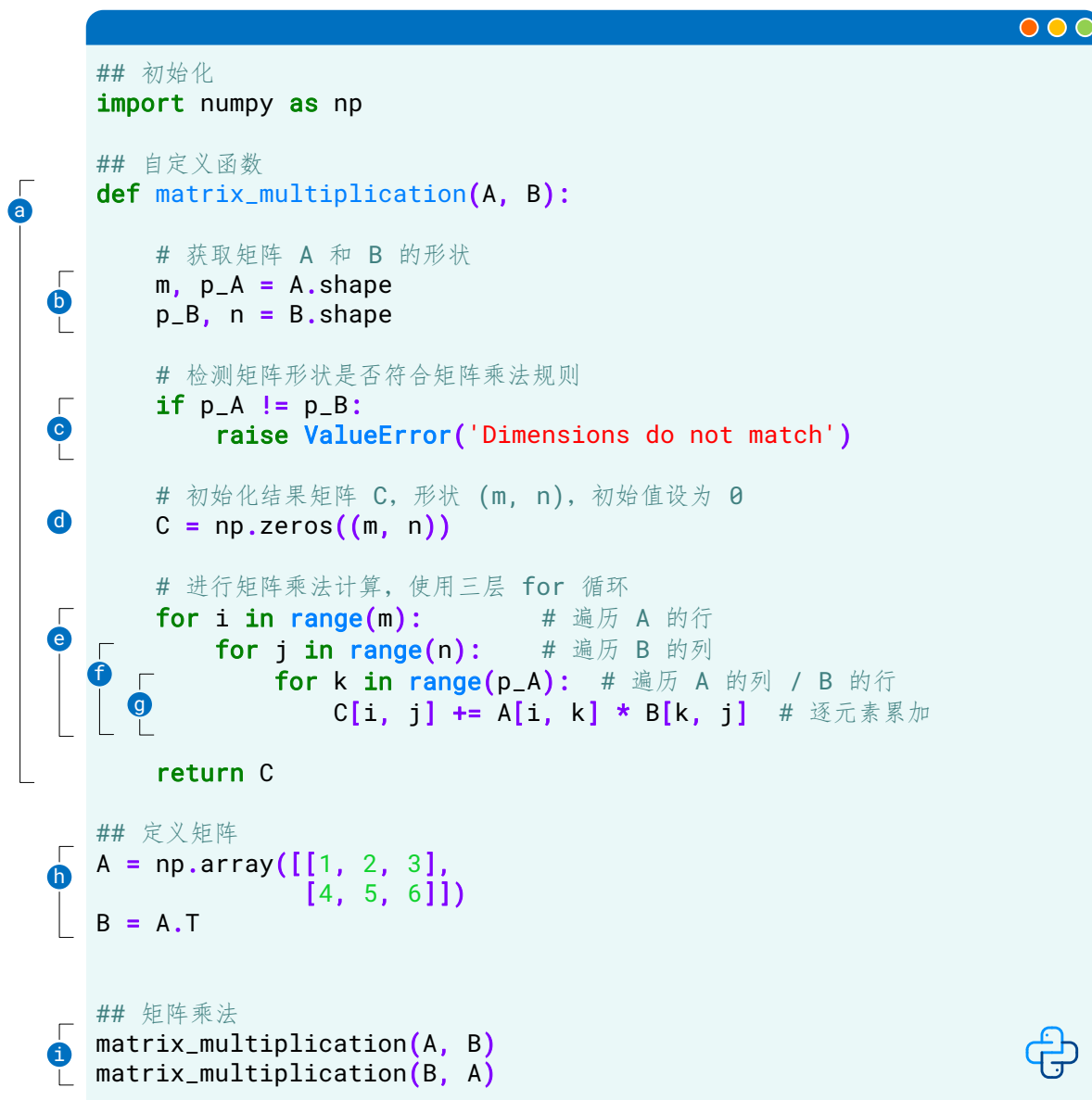
**h** 分别定义了矩阵  $A$ 、 $B$ 。

**i** 调用自定义函数计算  $AB$ 、 $BA$ 。



图 10. 矩阵乘法  $AB$  规则，利用三层 for 循环实现

代码 2. 自定义 Python 函数计算矩阵乘法，利用三层 for 循环实现 |  LA\_02\_04\_03.ipynb



矩阵乘法在线性代数中的重要性，怎么说都不为过。本节仅仅展示了矩阵乘法世界的冰山一角；接下来，我们要关注矩阵乘法的几何直觉，并且用几何视角去解释矩阵乘法的很多性质。然后，我们要从分块矩阵角度，来探索理解矩阵乘法的不同视角。这些视角都会在后续的线性代数话题中发挥至关重要的作用。



请大家用 DeepSeek/ChatGPT 等工具完成本节如下习题。

**Q1.** 判定如下成对矩阵的形状是否能够完成  $A @ B$  或  $B @ A$  矩阵乘法。

►  $A_{2 \times 2} = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}, B_{2 \times 2} = \begin{bmatrix} 5 & 6 \\ 7 & 8 \end{bmatrix}$

$$\blacktriangleright \mathbf{A}_{1 \times 3} = [1 \quad 2 \quad 3], \mathbf{B}_{3 \times 1} = \begin{bmatrix} 4 \\ 5 \\ 6 \end{bmatrix}$$

$$\blacktriangleright \mathbf{A}_{3 \times 2} = \begin{bmatrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \end{bmatrix}, \mathbf{B}_{2 \times 2} = \begin{bmatrix} 7 & 8 \\ 9 & 10 \end{bmatrix}$$

$$\blacktriangleright \mathbf{A}_{2 \times 3} = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}, \mathbf{B}_{3 \times 2} = \begin{bmatrix} 7 & 8 \\ 9 & 10 \\ 11 & 12 \end{bmatrix}$$

$$\blacktriangleright \mathbf{A}_{2 \times 2} = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}, \mathbf{B}_{2 \times 1} = \begin{bmatrix} 5 \\ 6 \end{bmatrix}$$

**Q2.** 如下成对矩阵形状都同时支持  $\mathbf{A} @ \mathbf{B}$  和  $\mathbf{B} @ \mathbf{A}$ ，请通过合适的办法（手算，编程）判断  $\mathbf{A} @ \mathbf{B}$  和  $\mathbf{B} @ \mathbf{A}$  是否相等。

$$\blacktriangleright \mathbf{A}_{2 \times 2} = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}, \mathbf{B}_{2 \times 2} = \begin{bmatrix} 5 & 6 \\ 7 & 8 \end{bmatrix}$$

$$\blacktriangleright \mathbf{A}_{1 \times 2} = [1 \quad 2], \mathbf{B}_{2 \times 1} = \begin{bmatrix} 3 \\ 4 \end{bmatrix}$$

$$\blacktriangleright \mathbf{A}_{2 \times 2} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \mathbf{B}_{2 \times 2} = \begin{bmatrix} 2 & 3 \\ 4 & 5 \end{bmatrix}$$

$$\blacktriangleright \mathbf{A}_{2 \times 2} = \begin{bmatrix} 1 & 2 \\ 2 & 1 \end{bmatrix}, \mathbf{B}_{2 \times 2} = \begin{bmatrix} 3 & 4 \\ 4 & 3 \end{bmatrix}$$

$$\blacktriangleright \mathbf{A}_{2 \times 2} = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}, \mathbf{B}_{2 \times 2} = \begin{bmatrix} 2 & 2 \\ 2 & 2 \end{bmatrix}$$

$$\blacktriangleright \mathbf{A}_{2 \times 2} = \begin{bmatrix} 1 & 0 \\ 0 & 2 \end{bmatrix}, \mathbf{B}_{2 \times 2} = \begin{bmatrix} 3 & 0 \\ 0 & 4 \end{bmatrix}$$

$$\blacktriangleright \mathbf{A}_{2 \times 2} = \begin{bmatrix} \cos(30^\circ) & -\sin(30^\circ) \\ \sin(30^\circ) & \cos(30^\circ) \end{bmatrix}, \mathbf{B}_{2 \times 2} = \begin{bmatrix} \cos(60^\circ) & -\sin(60^\circ) \\ \sin(60^\circ) & \cos(60^\circ) \end{bmatrix}$$

$$\blacktriangleright \mathbf{A}_{2 \times 2} = \begin{bmatrix} \cos(60^\circ) & -\sin(60^\circ) \\ \sin(60^\circ) & \cos(60^\circ) \end{bmatrix}, \mathbf{B}_{2 \times 2} = \begin{bmatrix} \cos(45^\circ) & -\sin(45^\circ) \\ \sin(45^\circ) & \cos(45^\circ) \end{bmatrix}$$

$$\blacktriangleright \mathbf{A}_{2 \times 2} = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}, \mathbf{B}_{2 \times 2} = \begin{bmatrix} 1 & 2 \\ 0 & 1 \end{bmatrix}$$

$$\blacktriangleright \mathbf{A}_{2 \times 2} = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}, \mathbf{B}_{2 \times 2} = \begin{bmatrix} 1 & 0 \\ 2 & 1 \end{bmatrix}$$

**Q3.** 给定列向量  $\mathbf{a} = \begin{bmatrix} 1 \\ 2 \end{bmatrix}$ ，计算  $\mathbf{a}^T @ \mathbf{a}$ 、 $\mathbf{a} @ \mathbf{a}^T$ 。

**Q4.** 给定列向量  $\mathbf{a} = \begin{bmatrix} 1 \\ 2 \end{bmatrix}$  和  $\mathbf{b} = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}$  计算  $\mathbf{a} @ \mathbf{b}^T$ 、 $\mathbf{b} @ \mathbf{a}^T$ 。

**Q5.** 给定  $\mathbf{A} = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$  和  $\mathbf{b} = \begin{bmatrix} 1 \\ 2 \end{bmatrix}$ ，计算  $\mathbf{A} @ \mathbf{b}$ 、 $\mathbf{b}^T @ \mathbf{A}^T$ 。

**Q6.** 给定  $\mathbf{A} = \begin{bmatrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \end{bmatrix}$ ，计算  $\mathbf{A} @ \mathbf{A}^T$ 、 $\mathbf{A}^T @ \mathbf{A}$ 。

**Q7.** 请用 NumPy 计算图 9 中每个子图对应的矩阵乘法，并且用向量内积演算。

**Q8.** 修改代码 2，交换第 1、2 层 for 循环，计算矩阵乘法。