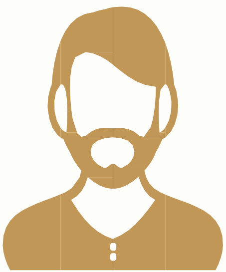
第2章 描述空间的工具—向量

第03讲 向量的四则运算

传媒与信息工程学院
欧 新 宇

第2章 描述空间的工具—向量



- ✓ 向量的基本知识回顾
- ✓ 列向量及向量的Python描述
- ✓ 向量的范数
- ✓ 常用向量
- ✓ 向量的加法和数乘
- ✓ 向量间的乘法
- ✓ 向量的线性组合

向量的加法

向量的加法

要进行**向量相加**，前提是两个向量具有相同的形态（即 $\boldsymbol{a}.shape = \boldsymbol{b}.shape$ ）。向量的加法可以理解为两个向量对应元素的**相加**（按位相加），生成的结果向量维度保持不变（即$(\boldsymbol{a}+\boldsymbol{b}).shape = \boldsymbol{a}.shape = \boldsymbol{b}.shape$ ）。

给定两个 n 维向量 u 和 v ，它们之间的加法运算规则可以表示为：

$$\begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ \cdots \\ u_n \end{bmatrix} + \begin{bmatrix} v_1 \\ v_2 \\ v_3 \\ \cdots \\ v_n \end{bmatrix} = \begin{bmatrix} u_1 + v_1 \\ u_2 + v_2 \\ u_3 + v_3 \\ \cdots \\ u_n + v_n \end{bmatrix}$$

向量的加法

一个例子

【**例1**】求解向量$u=\begin{bmatrix}1\\2\\3\\4\end{bmatrix}$和$v=\begin{bmatrix}5\\6\\7\\8\end{bmatrix}$的**和**的运算结果。

按照运算规则可以表示为：

$$u+v=\begin{bmatrix}1\\2\\3\\4\end{bmatrix}+\begin{bmatrix}5\\6\\7\\8\end{bmatrix}=\begin{bmatrix}1+5\\2+6\\3+7\\4+8\end{bmatrix}=\begin{bmatrix}6\\8\\10\\12\end{bmatrix}$$

向量的加法

使用Python语言进行描述

```
import numpy as np
u = np.array([[1,2,3,4]]).T
v = np.array([[5,6,7,8]]).T
w = u + v

print('u={}\n\n v={}\n\n u+v={}'.format(u,v,w))
```

注意已经转换为二维数组形式

```
u=[[1]
 [2]
 [3]
 [4]]
```

```
v=[[5]
 [6]
 [7]
 [8]]
```

```
u+v=[[ 6]
 [ 8]
 [10]
 [12]]
```

对于形态为1×n的单行矩阵和行向量的相加，也遵循**按位相加**的原则。

向量的数乘

向量的数乘

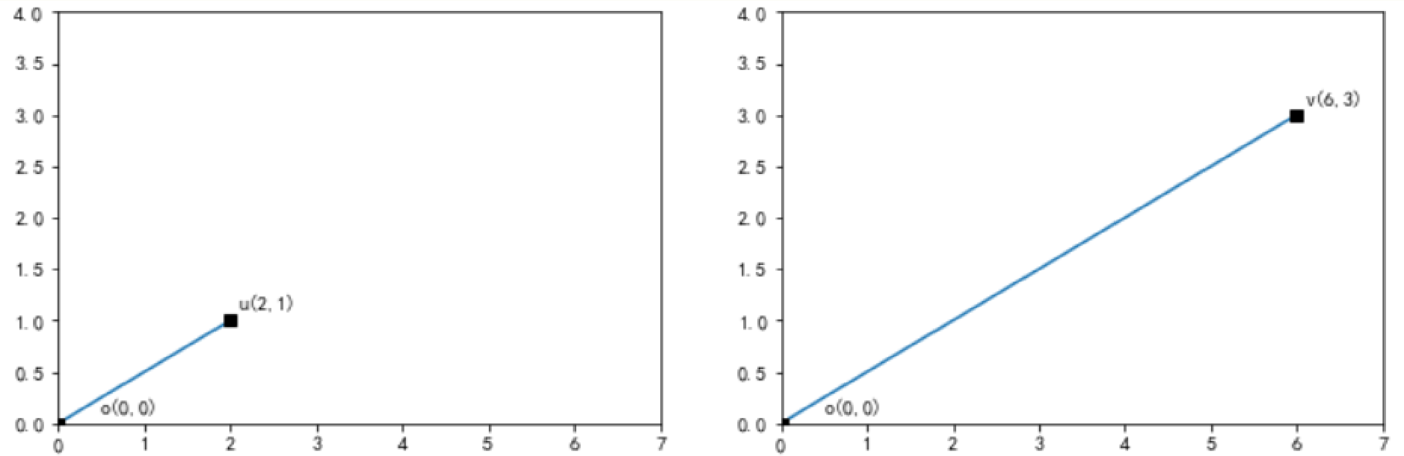
数乘的概念和特性

向量的数乘，又称为向量的数量乘法，它表示的是一个标量和一个向量之间的**乘积**关系。与向量的加法类似，向量的数乘是由标量和向量中的**每个元素**依次相乘，生成的新向量与原来的向量具有相同的形态。向量的数乘从**几何**意义上来说，可以理解为向量沿着所在直线的方向拉升相应的倍数，

- 拉升的**倍数**由标量决定，
- 拉升的**方向**与原向量方向保持不变。

向量的数乘

数乘的几何示意图



【结果分析】

从上图可以看到向量u和向量$v = 3 * u$的几何示意图，两个向量具有相同的方向，但具有不同的长度。向量$v = 3 * u$的长度刚好是向量u的3倍。

向量的数乘

数乘的运算规则

给定一个n维向量$\boldsymbol{u}$和一个标量v，他们的数乘变换运算规则可以表示为：

$$c * u = c * \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ \cdots \\ u_n \end{bmatrix} = \begin{bmatrix} c * u_1 \\ c * u_2 \\ c * u_3 \\ \cdots \\ c * u_n \end{bmatrix}$$

向量的数乘

一个例子

【例2】 给定标量5和向量$\boldsymbol{u}=\begin{bmatrix}1\\2\\3\\4\end{bmatrix}$，可以得到它们的数乘结果为：

$$5*\boldsymbol{u}=5*\begin{bmatrix}1\\2\\3\\4\end{bmatrix}=\begin{bmatrix}5*1\\5*2\\5*3\\5*4\end{bmatrix}=\begin{bmatrix}5\\10\\15\\20\end{bmatrix}$$

向量的数乘

使用Python语言进行描述

```
import numpy as np
u = np.array([[1,2,3,4]]).T
res = 5*u
print(res)
```

```
[[ 5]
 [10]
 [15]
 [20]]
```

- **结果分析：**

　　向量的数乘是没有方向的，无论左乘还是右乘都具有相同的效果，这意味着$\boldsymbol{u} * a = a * \boldsymbol{u}$。这个结论，可以轻松推广到矩阵的数乘。

课堂互动一 Link

向量间的乘法：内积和外积

向量的乘法：内积和外积

向量的内积

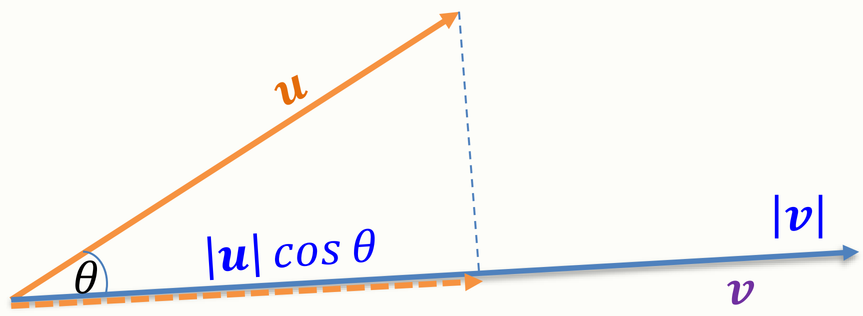
- 内积的**前提**：两个向量维数相同，长度相同
- 向量内积的**结果**：标量
- 内积的**别称**：点乘
- **运算规则**：对应位置上的元素相乘，然后合并相加

$$\boldsymbol{u}\cdot\boldsymbol{v}=\begin{bmatrix}u_1\\u_2\\u_3\\\cdots\\u_n\end{bmatrix}\cdot\begin{bmatrix}v_1\\v_2\\v_3\\\cdots\\v_n\end{bmatrix}=v_1v_1+v_2v_2+v_3v_3+\cdots v_nv_n$$

向量的乘法：内积和外积

向量的内积

- 内积的**几何形式**：$\boldsymbol{u} \cdot \boldsymbol{v} = |\boldsymbol{u}||\boldsymbol{v}| \cos\theta$
- 内积的**几何意义**：向量$\boldsymbol{u}$在向量$\boldsymbol{v}$方向上的投影长度乘以向量$\boldsymbol{v}$的模长。
- 内积的**几何表示**：



向量的乘法：内积和外积

向量的内积：一个例子

【例3】试计算，向量 $\boldsymbol{u}=\begin{bmatrix}2\\4\\6\end{bmatrix}$与向量 $\boldsymbol{v}=\begin{bmatrix}1\\3\\5\end{bmatrix}$的内积。

解：

$$\boldsymbol{u}\cdot\boldsymbol{v}=\begin{bmatrix}2\\4\\6\end{bmatrix}\cdot\begin{bmatrix}1\\3\\5\end{bmatrix}$$=2*1+4*3+6*5=2+12+30=44

向量的乘法：内积和外积

向量的内积：Python描述

```
[178]: import numpy as np
       u = np.array([2,4,6])
       v = np.array([1,3,5])
       print(np.dot(u,v))

       44
```

dot：点，点乘

- **结果分析：**

向量间的内积要求两个元素必须是向量形式，同时具有相同的形态。这意味，以**矩阵**形式表示的“向量”无法进行内积运算。

向量的乘法：内积和外积

向量的内积：Python描述

```
import numpy as np
u = np.array([[2,4,6]])
v = np.array([[1,3,5]])
print(np.dot(u,v))
```

行矩阵

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-181-54e4fb57e3e4> in <module>
      2 u = np.array([[2,4,6]])
      3 v = np.array([[1,3,5]])
----> 4 print(np.dot(u,v))

ValueError: shapes (1,3) and (1,3) not aligned: 3 (dim 1) != 1 (dim 0)
```

```
import numpy as np
u = np.array([[2,4,6]]).T
v = np.array([[1,3,5]]).T
print(np.dot(u,v))
```

列矩阵

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-180-bac849b462b9> in <module>
      2 u = np.array([[2,4,6]]).T
      3 v = np.array([[1,3,5]]).T
----> 4 print(np.dot(u,v))

ValueError: shapes (3,1) and (3,1) not aligned: 1 (dim 1) != 3 (dim 0)
```

结果分析：

可以看到相同形态的二维矩阵无法进行内积运算，哪怕是行数或列数为1的二维数组。这似乎和前面的运算规则相违背。

向量的乘法：内积和外积

向量的内积：Python描述

若使用二维数组表示的“**向量**”进行内积运算，则要求两个数组具有相同的长度，同时两个数组互为转置。

```
import numpy as np
u = np.array([[2,4,6]])
v = np.array([[1,3,5]]).T
print(np.dot(u,v))
```

```
[[44]]
```

具体的运算规则将在后面的矩阵乘法中进行解释。

向量的乘法：内积和外积

向量的外积

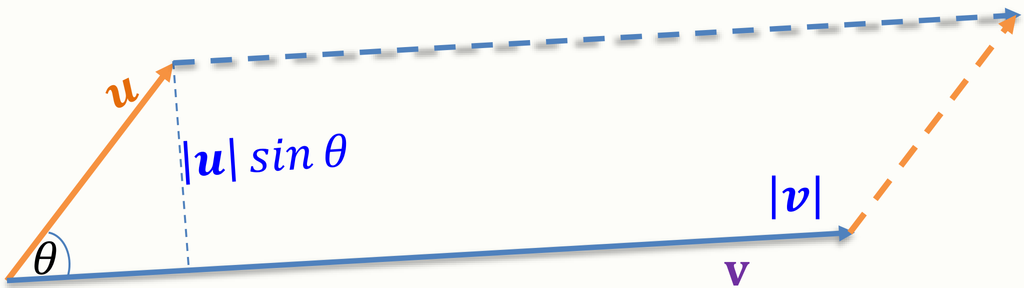
- 向量外积的**结果**：标量（二维）、向量（三维以上）
- 外积的**别称**：叉乘、向量积
- 二维平面的**运算规则**：

$$\boldsymbol{u} \times \boldsymbol{v} = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} \times \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} = u_1 v_2 - u_2 v_1$$

向量的乘法：内积和外积

向量的外积

- 外积的**几何形式：** $\boldsymbol{u} \times \mathbf{v} = |\boldsymbol{u}||\boldsymbol{v}| \sin\theta$
- **几何意义（二维）**：向量u和向量v张成的平行四边形的面积。
- **几何表示（二维）**：



向量的乘法：内积和外积

向量的外积

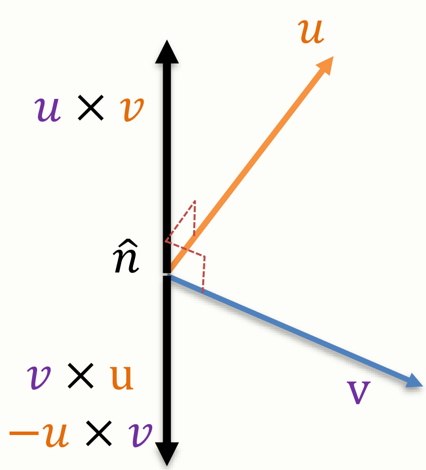
- 三维平面的**运算规则**：

$$\boldsymbol{u} \times \boldsymbol{v} = \begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix} \times \begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix} = \begin{bmatrix} u_2 v_3 - u_3 v_2 \\ u_3 v_1 - u_1 v_3 \\ u_1 v_2 - u_2 v_1 \end{bmatrix}$$

向量的乘法：内积和外积

向量的外积

- **几何意义（三维）**：向量u和向量v张成的平面的法向量，该向量垂直于 u 和 v 向量构成的平面。
- **几何表示（三维）**：



向量的乘法：内积和外积

向量的外积：二个例子（二维向量）

【例4】 试计算，向量 $\boldsymbol{u} = \begin{bmatrix} 2 \\ 4 \end{bmatrix}$ 与向量 $\boldsymbol{v} = \begin{bmatrix} 3 \\ 5 \end{bmatrix}$ 的外积。

解：$\boldsymbol{u} \times \boldsymbol{v} = \begin{bmatrix} 2 \\ 4 \end{bmatrix} \times \begin{bmatrix} 3 \\ 5 \end{bmatrix} = 2*5 - 4*3 = 10 - 12 = -2$

Python描述

```
import numpy as np
u = np.array([2,4])
v = np.array([3,5])
print(np.cross(u,v))
```

```
-2
```

向量的乘法：内积和外积

向量的外积：二个例子（三维向量）

【例5】试计算，向量 $\boldsymbol{u}=\begin{bmatrix}2\\3\\4\end{bmatrix}$与向量 $\boldsymbol{v}=\begin{bmatrix}1\\3\\6\end{bmatrix}$的外积。

解：

$$\boldsymbol{u}\times\boldsymbol{v}=\begin{bmatrix}2\\3\\4\end{bmatrix}\times\begin{bmatrix}1\\3\\6\end{bmatrix}=\begin{bmatrix}u_2v_3-u_3v_2\\u_3v_1-u_1v_3\\u_1v_2-u_2v_1\end{bmatrix}=\begin{bmatrix}3*6-4*3\\4*1-2*6\\2*3-3*1\end{bmatrix}=\begin{bmatrix}6\\-8\\3\end{bmatrix}$$

Python描述

```
import numpy as np
u = np.array([2,3,4])
v = np.array([1,3,6])
print(np.cross(u,v))
```

```
[ 6 -8  3]
```

课堂互动二 Link

向量的线性组合

向量的线性组合

概念和运算规则

向量的**线性组合**：基于向量加法和数乘构建的基本运算。

基本**运算规则**：假设存在标量a, b, c 和向量$\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w}$，则有：

$$a\boldsymbol{u} + b\boldsymbol{v} + c\boldsymbol{w} = a\begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix} + b\begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix} + c\begin{bmatrix} w_1 \\ w_2 \\ w_3 \end{bmatrix} = \begin{bmatrix} au_1 + bv_1 + cw_1 \\ au_2 + bv_2 + cw_2 \\ au_3 + bv_3 + cw_3 \end{bmatrix}$$

向量的线性组合

一个例子

【例6】给定标量$a=2,b=4,c=6$和向量$\boldsymbol{u}=\begin{bmatrix}1\\2\\3\end{bmatrix},\boldsymbol{v}=\begin{bmatrix}4\\5\\6\end{bmatrix},\boldsymbol{w}=\begin{bmatrix}7\\8\\9\end{bmatrix}$，试求线性组合$a\boldsymbol{u}+b\boldsymbol{v}+c\boldsymbol{w}$。

解：$a\boldsymbol{u}+b\boldsymbol{v}+c\boldsymbol{w}=2\begin{bmatrix}1\\2\\3\end{bmatrix}+4\begin{bmatrix}4\\5\\6\end{bmatrix}+6\begin{bmatrix}7\\8\\9\end{bmatrix}$

$$=\begin{bmatrix}2*1+4*4+6*7\\2*2+4*5+6*8\\2*3+4*6+6*9\end{bmatrix}=\begin{bmatrix}60\\72\\84\end{bmatrix}$$

向量的线性组合

Python描述

```
import numpy as np
u = np.array([[1,2,3]]).T
v = np.array([[4,5,6]]).T
w = np.array([[7,8,9]]).T
print(2*u + 4*v + 6*w)
```

```
[[60]
 [72]
 [84]]
```

```
import numpy as np
u = np.array([1,2,3]).T
v = np.array([4,5,6]).T
w = np.array([7,8,9]).T
print(2*u + 4*v + 6*w)
```

```
[60 72 84]
```

- **结果分析：**

  向量的线性组合需要将向量转换为**列向量**，因此需要使用**二维数组**来表示列向量。直接进行线性变换，可以运算，但无法获得最终的列向量。

课堂互动三 Link

第03讲 向量的四则运算

读万卷书 行万里路 只为最好的修炼

QQ：14777591 (宇宙骑士)

Email：ouxinyu@alumni.hust.edu.cn

Tel：18687840023