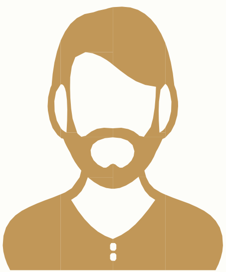
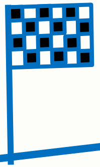
第02讲 向量的基础知识

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



第2章 描述空间的工具—向量

总体说明

空间是贯穿**线性代数**整个领域的主干和核心概念，我们所有的概念和应用都会构架在空间这个逻辑实体上。而向量和矩阵就是我们用来填充这个实体的工具，包括运算、映射、降维、投影、近似求解、特征提取等，都将建立在基于矩阵和向量的空间中实现。

向量的基本知识回顾

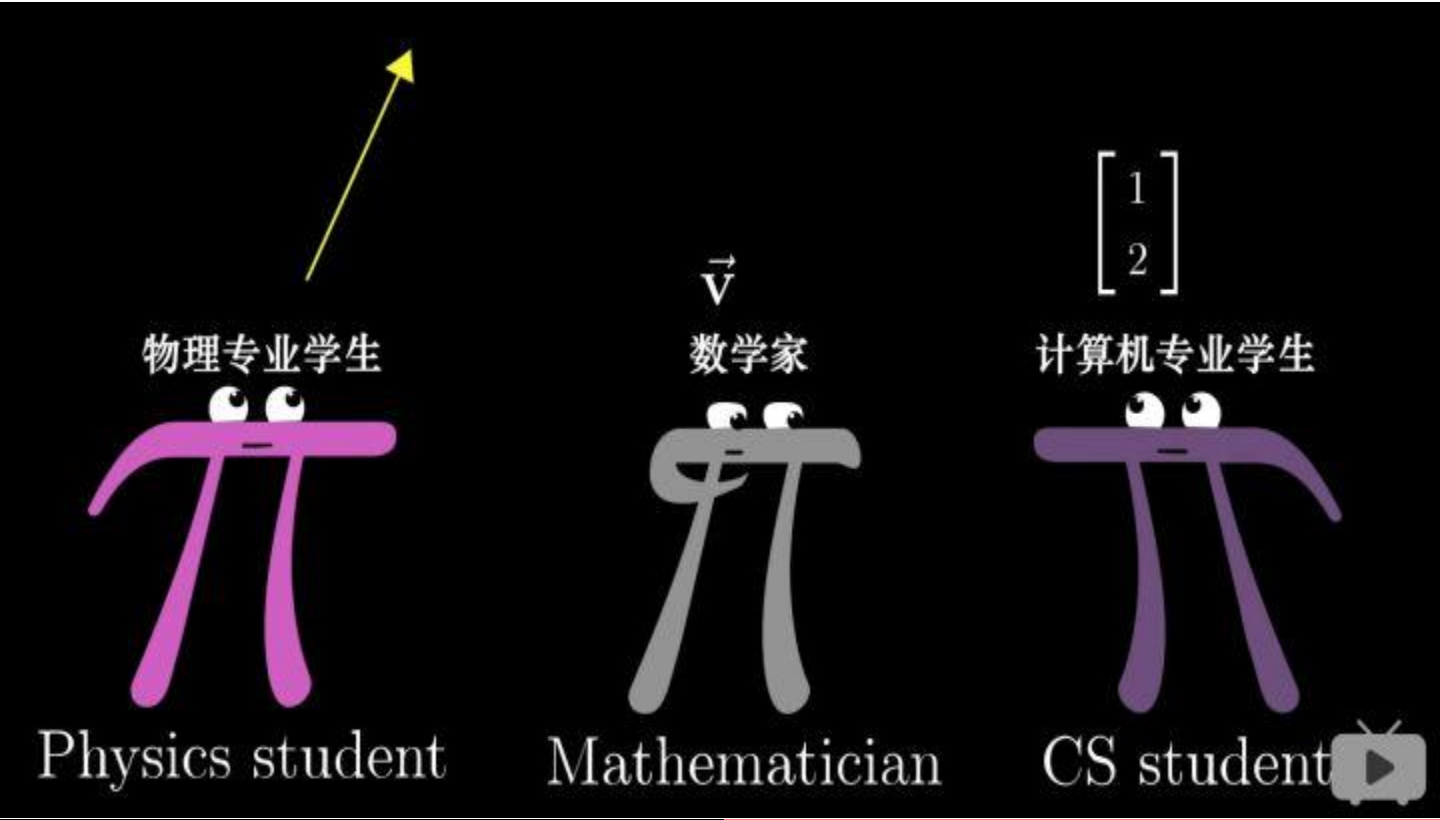
向量的基本知识回顾

向量的定义

- **向量**：也称欧几里得向量、几何向量、矢量，它指具有大小和方向的量。它可以形象化地表示为带箭头的线段。直观地说，一组排列成行或列的有序数字，就是向量。
- **箭头所指**：代表向量的方向；
- **线段长度**：代表向量的大小；
- **向量的记法**：
  - 印刷体，记作*小写粗斜体字母*，如 $\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{u}, \boldsymbol{v}$；
  - 手写体，在字母顶上加一小箭头“→”，如 $\vec{u}$
  - 给定向量的起点A和终点B，可记作$\boldsymbol{AB}$；
  - 在空间直角坐标系中，以数对形式表示，如 (2, 3)

向量的基本知识回顾

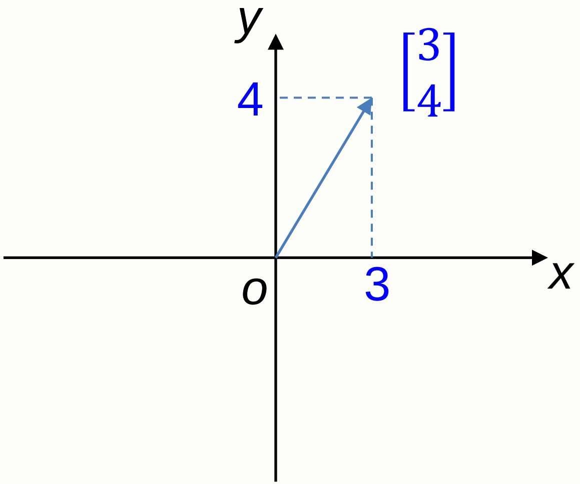
向量的定义



向量的基本知识回顾

二维向量的空间表示

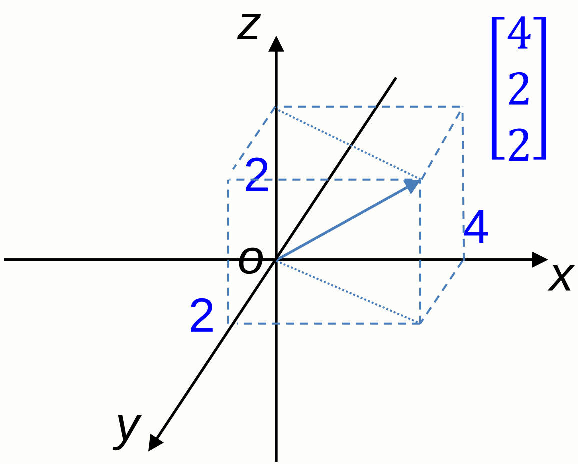
给定二维向量$\boldsymbol{u}=\begin{bmatrix}3\\4\end{bmatrix}$，它有两个分量，其中$x$分量值为3，$y$分量值为4，以原点(0,0)为起点，可以在直角坐标系中构建一条有向线段。



向量的基本知识回顾

三维向量的空间表示

给定三维向量$\boldsymbol{u}=\begin{bmatrix}4\\2\\2\end{bmatrix}$，它有三个分量，其中$x$分量值为4，$y$分量值为2，$z$分量值为2，以原点(0,0,0)为起点，可以在三阶笛卡尔坐标系中构建一条有向线段。



列向量及向量的Python描述

列向量

计算机领域主要使用列向量

根据数字的排列方式，向量可以被分为行向量和列向量。在计算机领域中，我们常使用**列向量**来表示和处理向量。例如，将矩阵***A***映射到向量x上时，可以用***A***x来表示，最常见的应用是求解方程组。列向量通常由两种表示方法。

- 直观表示：$\alpha = \begin{bmatrix} 2 \\ 3 \end{bmatrix}$，$\beta = \begin{bmatrix} 2 \\ 3 \\ 4 \end{bmatrix}$

- 单行表示（更常用）：$\alpha = [2, 3]^T$，$\beta = [2, 3, 4]^T$

基于Python语言的向量表示

在Python中，最重要，也是最常用的一个库就是数学计算库 **Numpy**，它也是本门课中最主要的python工具包。下面我们将使用numpy库来实现**数组(向量、矩阵)**的创建。

- 创建numpy数组（向量）

```python
import numpy as np
A = np.array([1, 2, 3, 4])
print(a)
```

```
[[1 2 3 4]]
```

- 获取变量数据类型

```python
type(A)
```

```
numpy.ndarray
```

列向量的生成

在机器学习及大多数计算机任务中，都会以列向量的方式对数据进行处理，而numpy默认生成的是行向量。所以，需要事先进行转换。

最容易也是最直接的方法：**矩阵转置（Transpose）**。

值得注意的是，在计算机的存储意识中，向量是一维的量，它只在一个维度上具有值。因此，无法进行转置。

```python
import numpy as np
A = np.array([1, 2, 3, 4])
B = A.transpose()
C = A.T

print('a={}'.format(A))
print('A={}'.format(B))
print('B={}'.format(C))
```

```
a=[1 2 3 4]
A=[1 2 3 4]
B=[1 2 3 4]
```

列向量的生成

如何处理呢?

一维向量 → 二维矩阵

- 当我们使用**向量**来表示一个数据时，可以表示为：$\boldsymbol{a}=[a_1,a_2,\ldots,a_n]$，此时 $\boldsymbol{a}$ 是一个维度为 1，长度为 n 的数据（向量）；
- 当我们使用**矩阵**来表示这个向量时，则可以表示为：$\boldsymbol{A_2}=[a_{11},a_{12},\ldots,a_{1n}]$，此时 $\boldsymbol{A_2}$ 是一个维度为2的数据（矩阵），第一个维度长度为 1，第二个维度长度为 n，我们也可以将这样的矩阵理解为一个**行向量**，一行 n 列，形态为：$(1, n)$。
- 在转换为二维矩阵后，就可以通过矩阵的转置实现行向量向列向量的转换，此时的数据 $\boldsymbol{A_2}$ 将转变为一个列向量 $\boldsymbol{B_2}$，n 行一列，形态为：$(n, 1)$，表示为：$\boldsymbol{A_3}=[a_{11};\ a_{21};\ \ldots;\ a_{n1}]$。

列向量的生成

使用二维矩阵进行转换

```python
import numpy as np
A2 = np.array([[1, 2, 3, 4]])
B2 = A2.transpose()
C2 = A2.T

print('A2={}'.format(A2))
print('B2={}'.format(B2))
print('C2={}'.format(C2))
```

```
A2=[[1 2 3 4]]
B2=[[1]
 [2]
 [3]
 [4]]
C2=[[1]
 [2]
 [3]
 [4]]
```

```python
print('A2的形态: {}'.format(A2.shape))
print('B2的形态: {}; C2的形态: {}'.format(B2.shape, C2.shape))
```

```
A2的形态: (1, 4)
B2的形态: (4, 1); C2的形态: (4, 1)
```

列向量的生成

【结果分析】

- 原始的一维向量a和经过.transpose()或 .T 转换后的向量b和c，都呈现为相同的形态（4，1)，并且值也完全相同。这说明在Python中，转置在向量上是无效的。
- 当我们使用**二维矩阵**进行转换时，新生成的矩阵A_2是一个$1\times n$ 的二维矩阵，当经过 .transpose() 和 .T 转换后，两个矩阵都变成了$n\times 1$的矩阵。这说明，原来以二维矩阵显示的行向量，形态为(1，4)；已经转换为以二维矩阵显示的列向量了，形态为(4, 1)。

列向量的生成

特别注意

在Python中一维向量和二维矩阵的表示非常容易转换，只需要增加一层中括号"[ ]"就可以实现从一维到二维的转换。

相似地，三维矩阵使用三层中括号表示，n 维矩阵使用n层中括号表示。

下面给出一维向量和二维向量的表示。

```
A = np.array([1, 2, 3, 4])
A2 = np.array([[1, 2, 3, 4]])
```

课堂互动一 Link

向量的范数

向量的范数

范数的定义

范数(norm)：数学中的一种基本概念。在线性代数、泛函分析及相关的数学领域，范数是一个具有“长度”概念的函数。在泛函分析中，它定义在赋范线性空间中，并满足一定的条件，即①非负性；②齐次性；③三角不等式。

范数常常被用来度量向量空间（或矩阵）中的某个向量的长度或大小。例如，在二维的欧氏几何空间$\boldsymbol{R}$中，元素被刻画成一个从原点出发的带有箭头的有向线段，每一个矢量的有向线段的长度即为该矢量的**欧氏范数**。

向量的范数

范数的定义

对于一个n维向量$L_p = (v_1, v_2, \dots, v_n)$，其大小可以用向量$v$的**范数**来进行衡量，其一般形式可以表示为$L_p$：

$$L_p(v) \equiv \|v\|_p = \left(\sum_{i=1}^{n} |v_i|^p\right)^{\frac{1}{p}}$$

其中，$p \in \Re$且$p \geq 1$。

值得注意的是，符号 L_p 也被称为 ***p*-范数**（p-norm）。

向量的范数

范数的典型性质

范数直观上可以表示为衡量向量长度的函数，相当于求原点o到点v的**距离**，因此所有的范数都满足以下三条性质：

- **非负性：** 函数的值永远都是非负，当且仅当向量为全零向量时，范数函数值为0.

$$\begin{cases} f(\boldsymbol{x}) \geq 0 & \boldsymbol{x} \neq 0 \\ f(\boldsymbol{x}) = 0 & \boldsymbol{x} = 0 \end{cases}$$

  其中，$\boldsymbol{x}$是任意向量，$f(\boldsymbol{x})$是$\boldsymbol{x}$的范数。

- **三角不等式：** 两个向量范数的和大于等于两个向量和的范数。

$$f(\boldsymbol{x}) + f(\boldsymbol{y}) \geq f(\boldsymbol{x} + \boldsymbol{y})$$

- **正值齐次性：** $\forall a \in \boldsymbol{R},\ f(a\boldsymbol{x}) = |a| f(\boldsymbol{x})$

向量的范数

L2范数

对于p-**范数**，当p等于2时，L_2**范数**（L2范数，L2 norm）也被称为欧几里得范数，它表示从源点出发到向量v的欧几里得距离，简称**欧式距离**。向量v的欧式距离通常可以表示为$\|v\|_2$，也可以省略为：$\|v\|$，即：

$$\|v\|_2 = \|v\| = \left(\sum_{i=1}^{n} |v_i|^2\right)^{\frac{1}{2}}$$

在二维空间中，向量$v(x, y)$ 的长度可以表示为：

$$\|v\| = \left(\sum_{i=1}^{2} |v_i|^2\right)^{\frac{1}{2}} = \left(|x|^2 + |y|^2\right)^{\frac{1}{2}}$$

向量的范数

L2范数-向量间的距离

对于空间中的任意两个向量，也可以利用L2范数来求它们之间的距离。

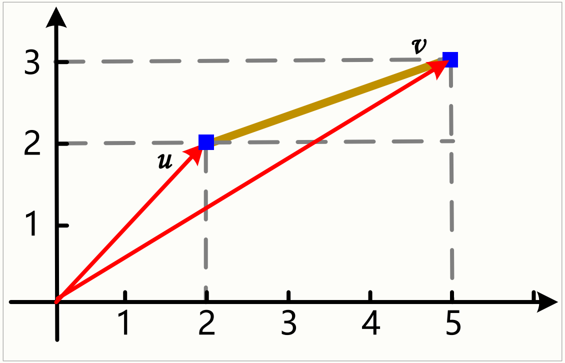
例如，求向量u=[2,2]和向量v=[5,3]之间的距离$|d|$。

可以表示为：

$$|d| = (|x_1 - x_2|^2 + |y_1 - y_2|^2)^{\frac{1}{2}}$$

$$= (|2-5|^2 + |2-3|^2)^{\frac{1}{2}}$$

$$= \sqrt{10}$$



向量的范数

L2范数-向量间的距离

L2范数在机器学习和深度学习中被广泛应用，在具体的应用中，我们可以用两个向量的欧式距离来表达它们之间的相似性。此外，还可以使用L2范数来对样本的特征或权重进行**约束**。

例如，公式$argmin\|X\theta - y\|^2 + \|\Gamma\theta\|^2$，给出了岭回归算法（一种使用L2正则化的线性模型）的典型形式。

如上例所示，为了方便计算，经常会使用平方L2范数来替代L2范数，省去开平方操作。

向量的范数

L2范数-向量的点积

两个向量的点积 $x^T y$可以用范数来表示，具体如下：

$$x^T y = \|x\|_2 \|y\|_2 \cos\theta$$

向量的范数

L1范数

当p等于1时，***p*-范数**就会退化成另外一种常用的特殊范数$\boldsymbol{L_1}$**范数**（L1范数, L1 norm）。其数学表达为：

$$L_1(v) \equiv \|v\|_1 = \sum_{i=1}^{n} |v_i|$$

不难看出，它是对向量中的每个元素的绝对值的累加。因此，L1范数可以用来区分零元素和非零元素。在机器学习和深度学习中，区分零和非零是非常重要，因此它也可以实现对样本特征或权重的约束。

例如，公式$argmin\|X\theta - y\|^2 + \alpha\|\theta\|_1$，给出了套索(Lasso)回归算法（一种使用L1正则化的线性模型）的典型形式。

向量的范数

无穷范数

当p趋近于∞时，***p*-范数**就会变为机器学习中经常出现的另外一种范数：$\boldsymbol{L_\infty}$**范数**，又称为无穷范数或最大范数(max norm)。L_∞范数表示向量中最大分量的绝对值:

$$L_\infty(v) \equiv \|v\|_\infty = \max_i |v_i|$$

在数据处理中，有时需要选择出响应最大的分量来进行处理。此时，就可以使用无穷范数。在深度学习中的卷积神经网络CNN模型中的最大池化（max-pooling）就是这种思想最典型的应用。

向量的范数

F-范数

弗罗贝尼乌斯范数（Frobenius范数）：简称为F范数，是一种定义在矩阵上的范数，用于衡量矩阵的大小。F范数表示矩阵中各元素的平方和开方，其数学表达为：

$$\|\boldsymbol{A}\|_{\mathrm{F}} = \sqrt{\sum_{i,j} A_{i,j}^2}$$

F范数的计算规则与L2范数非常相似，区别是它是定义在矩阵上的范数，用于衡量矩阵的大小。

向量的范数

范数的Python描述

- **L2范数：**

  np.linalg.norm(x) *# 默认状态为L2范数*

  np.linalg.norm(x, ord=2)

- **L1范数：**

  np.linalg.norm(x, ord=1)

- **无穷范数：**

  np.linalg.norm(x, ord=np.inf)

- **F范数：**

  np.linalg.norm(X) *# 当X为矩阵时，即为F范数*

课堂互动二 Link

常用的向量

常用的向量

全0向量和全1向量

全0向量：所有分量都为0的向量，用一个粗体的0表示。

全1向量：所有分量都为1的向量，用一个粗体的1表示。

全0向量和全1向量，通常都是为了保证表达式描述的正确性或为变量进行初始化使用。

$$\mathbf{0} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \qquad \mathbf{1} = \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix}$$

常用的向量

One-hot向量

One-Hot向量：又称为独热码（独热编码），有且仅有一个分量为1，其它分量都为0的**向量**。其形式如下：

$$a = [0,0,1,0,0,0,0,0]$$

One-Hot向量在编码中使用广泛，例如，在分类应用中，对于一个有8个类别的场景，通常将分类结果编码为一个One-Hot向量，元素为1的那个分量对应的类别即为真实（预测）的分类类别。

One-Hot属于一种稀疏编码，不同分量之间默认没有关联。并且不同分量间通常是独立同分布（IID）的。

常用的向量

单位向量

单位向量（Unit Vector）：L2范数为1的向量。

$$\|v\| = 1$$

单位向量将向量的长度约束为1，这样很好地屏蔽了模长带来的影响，使向量只用于表达方向一种量。例如常用的**余弦相似性**（cosine similarity）就通过比较两个向量的夹角大小来确定向量间的相似性。

$$\cos\theta = u^T v = v^T u$$

其中，θ为两个向量的夹角，u和v为两个单位向量。

课堂互动三 Link

第02讲 向量的基础知识

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