

# 1 Vector 向量

有向量的地方，就有几何



几何——指向真理之乡，创造哲学之魂。

*Geometry will draw the soul toward truth and create the spirit of philosophy.*

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



- ◀ matplotlib.pyplot.quiver() 绘制箭头图
- ◀ numpy.add() 向量/矩阵加法
- ◀ numpy.array([[4,3]]) 构造行向量
- ◀ numpy.array([[4,3]]).T 行向量转置得到列向量
- ◀ numpy.array([[4], [3]]) 构造列向量
- ◀ numpy.array([4, 3][:, None]) 构造列向量; 按照 [:, None] 形式广播序列; None 代表 numpy.newaxis, 增加新维度
- ◀ numpy.array([4, 3][:, numpy.newaxis]) 构造列向量
- ◀ numpy.array([4, 3][None, :]) 构造行向量; 按照 [None, :] 形式广播序列; None 代表 numpy.newaxis, 增加新维度
- ◀ numpy.array([4, 3][numpy.newaxis, :]) 构造行向量
- ◀ numpy.array([4,3]) 构造一维序列, 严格来说不是行向量
- ◀ numpy.array([4,3]).reshape((1, -1)) 构造行向量
- ◀ numpy.array([4,3]).reshape((-1,1)) 构造列向量
- ◀ numpy.array([4,3], ndmin=2) 构造行向量
- ◀ numpy.linalg.norm() 默认计算 L2 范数
- ◀ numpy.ones() 生成全 1 向量/矩阵
- ◀ numpy.r\_[] 将一系列的序列合并; 'r' 设定结果以行向量 (默认) 展示, 比如 numpy.r\_[numpy.array([1,2]), 0, 0, numpy.array([4,5])] 默认产生行向量
- ◀ numpy.r\_['c', [4,3]] 构造列向量
- ◀ numpy.subtract() 向量/矩阵减法
- ◀ numpy.zeros() 生成全 0 向量/矩阵
- ◀ zip(\*) 用于将可迭代的对象作为参数, 将对象中对应的元素打包成一个个元组, 然后返回由这些元组组成的列表。\*代表解包, 返回的每一个都是元祖类型, 而并非是原来的数据类型

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)



## 1.1 向量：有箭头的线段，一行或一列元素

向量的内涵极为丰富，本节主要从几何和数据两个视角来看向量。

### 几何视角

如图 1 所示，平面上，向量是**有方向的线段** (directed line segment)。**线段的长度代表向量的大小** (the length of the line segment represents the magnitude of the vector)。**箭头代表向量的方向** (the direction of the arrowhead indicates the direction of the vector)。

本书中，向量符号采用加粗斜体小写字母，比如  $\mathbf{a}$ ；矩阵符号则采用加粗斜体大写字母，比如  $\mathbf{A}$ 。

图 1 中，向量  $\mathbf{a}$  的**起点** (initial point) 是  $O$ ，向量的**终点** (terminal point) 是  $A$ 。如果向量的起点和终点相同，向量则为**零向量** (zero vector)，可以表示为  $\mathbf{0}$ 。

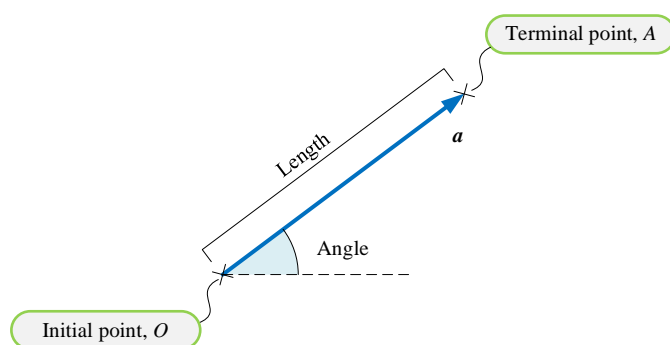


图 1. 向量起点、终点、大小和方向

图 2 给出的是几种向量的类型。

和起点无关的向量叫做**自由向量** (free vector)，如图 2 (a)；和起点有关的向量被称作，**固定向量** (fixed vector)，如图 2 (b) 和 (c)；方向上沿着某一个特定直线的向量，称之为**滑动向量** (sliding vector)，如图 2 (d)。没有特别说明时，本书的向量一般是固定向量。除非特别说明，向量的起点一般都在原点。比如说，向量平移时，向量的起点和终点都要同步移动。

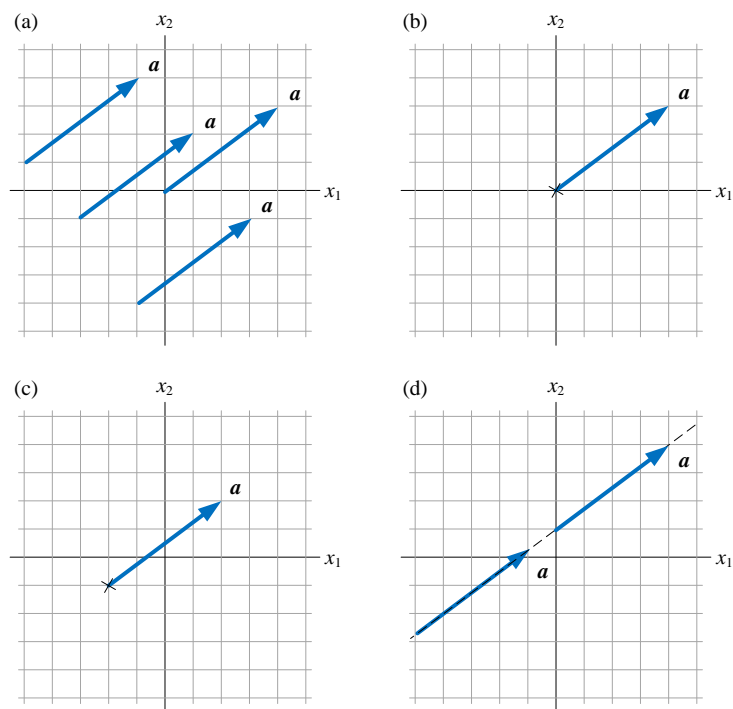


图 2. 几种向量类型

从解析几何角度看，向量和坐标直接相关。

如图 3 (a) 所示，一般情况下直角坐标系中任意一点坐标可以通过**多元组** (tuple) 来表达，比如图 3 (a) 所示平面直角坐标系上  $A$  点坐标  $(4, 3)$ ， $B$  点坐标  $(-3, 4)$ 。

图 3 (b) 所示，以原点作为向量起点， $A$  点对应向量  $\mathbf{a}$  终点， $B$  点对应向量  $\mathbf{b}$  终点。

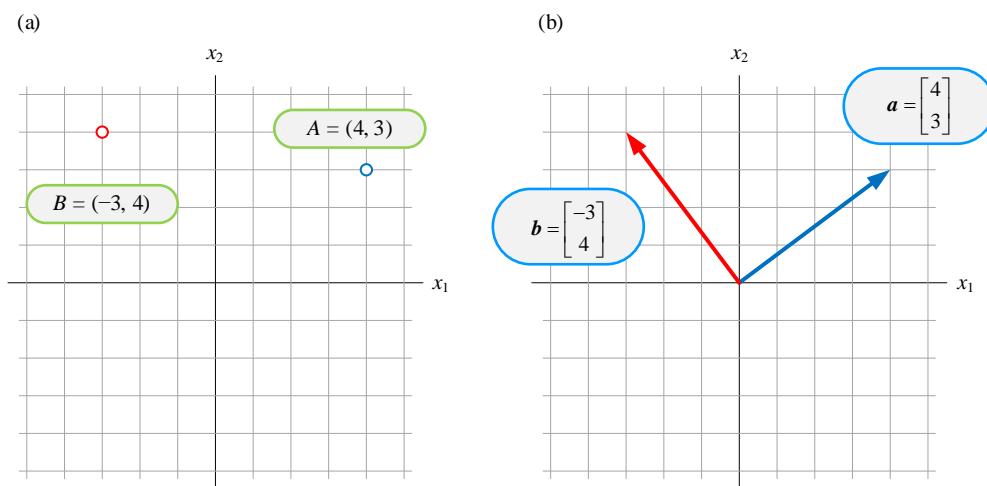


图 3. 平面坐标和向量关系

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)

向量也可以是两点连线构造的有方向线段，如图 4 所示。图中向量  $\mathbf{a}$  以  $P$  为起点、 $A$  为终点，长度是  $A$  和  $P$  两点距离。

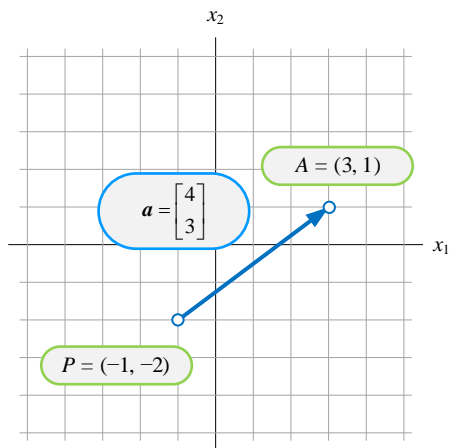


图 4. 连接两点的有方向线段

### 继续丰富向量的几何内涵

向量的几何视角还可以进一步扩展。

几何上，切线指的是一条刚好触碰到曲线上某一点的直线；曲线的法线是垂直于曲线上一点的切线的直线。将切线、法线两个概念进入向量中可以得到**切向量** (tangent vector) 和**法向量** (normal vector) 这两个概念。图 5 所示为直线和曲线某一点处的切向量和法向量。

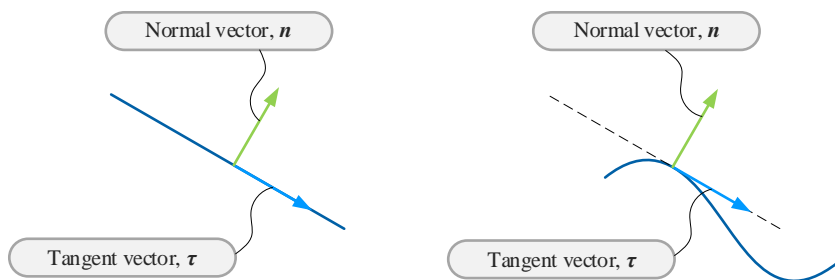


图 5. 切向量和法向量

自然界的风、水流、电磁场，在空间的每一个点上对应的物理量既有强度也有方向。将这些既有大小又有方向的场抽象出来就是**向量场** (vector field)。

在从书中，我们会使用向量场来描述函数在某一点的**梯度向量** (gradient vector)。图 6 所示为函数  $f(x_1, x_2)$  在各个点处的曲面等高线和梯度向量在  $x_1x_2$  平面上的投影。仔细观察，可以发现任意一点处梯度向量垂直于该点处等高线。

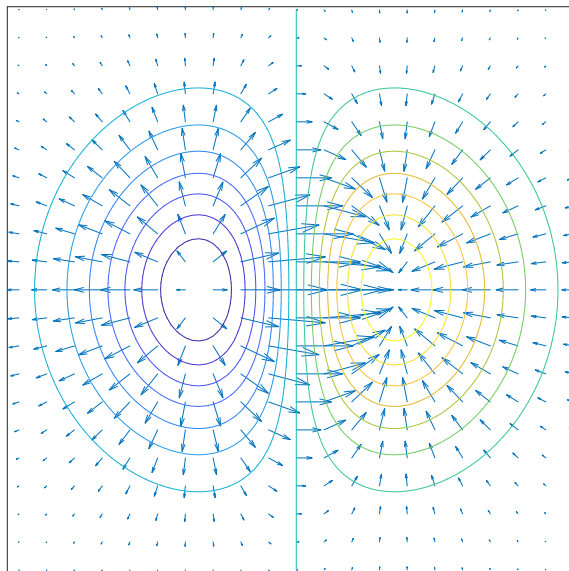


图 6. 函数的向量场

图 7 所示为  $f(x_1, x_2)$  函数对应的三维曲面。把图 7 比作一座山峰的话，山峰上任意一点的梯度就是该点坡度最陡峭方向的反方向，梯度向量的大小告诉我们坡度有多陡。

白话来说，曲面某一点的梯度向量就是该点放置一个小球，滚下瞬间小球运动方向叫做梯度下降方向，反方向就是梯度向量方向。本书将在多元微积分中介绍梯度向量这一概念。

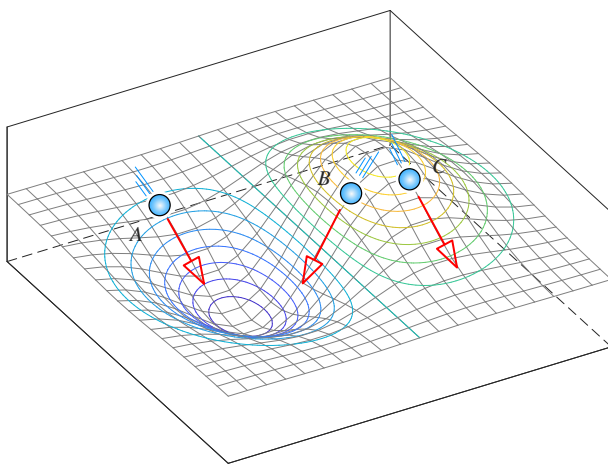


图 7. 三维曲面上定义梯度向量

## 数据视角

从数据角度来看，矩阵就是表格，它是由若干行或若干列元素排列得到的阵列。矩阵内的元素可以是实数、虚数、符号，甚至是数学式。

向量可以看做是**一维矩阵** (one-dimensional matrix)；向量要么一行多列、要么一列多行。而矩阵可以视作是一系列排列整齐的行向量或列向量。

一行多列的向量是**行向量** (row vector)，一列多行的向量叫**列向量** (column vector)；白话讲，行向量将  $n$  个数字排成一行，结构为  $1 \times n$  (代表 1 行， $n$  列)，如图 8 (a)。下式行向量  $\mathbf{a}$  为 1 行 4 列：

$$\mathbf{a} = [1 \quad 2 \quad 3 \quad 4] \quad (1)$$

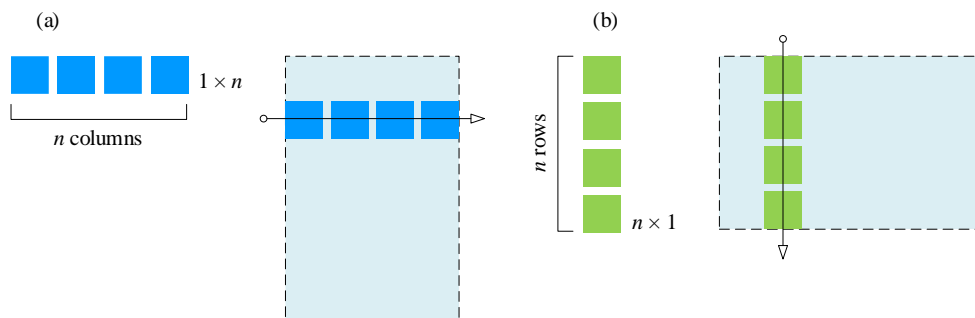


图 8. 行向量和列向量

列向量将  $n$  个数字排成一列，结构为  $n \times 1$  (代表  $n$  行，1 列)。

举个例子，下式行向量  $\mathbf{b}$  为 4 行 1 列：

$$\mathbf{b} = \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \end{bmatrix} \quad (2)$$

不加说明时，本书中向量一般指的是列向量。



数据科学和机器学习都是“数据驱动”。数据是各种算法的绝对核心。“Garbage in, garbage out”，优质的数据本身就极具价值，甚至不需要怎样高深的数据分析。

而有数据的地方，就有向量。

如图 9 所示，数据矩阵  $\mathbf{X}$  的每一行是一个行向量，代表一个观察值； $\mathbf{X}$  的每一列为一个列向量，代表某个特征上的所有数据。

本丛书使用频率最高的数据是鸢尾花卉数据集。数据集的全称为**安德森鸢尾花卉数据集** (Anderson's Iris data set)，是植物学家埃**德加·安德森** (Edgar Anderson) 在加拿大魁北克加斯帕半岛上的采集的 150 个鸢尾花样本数据。

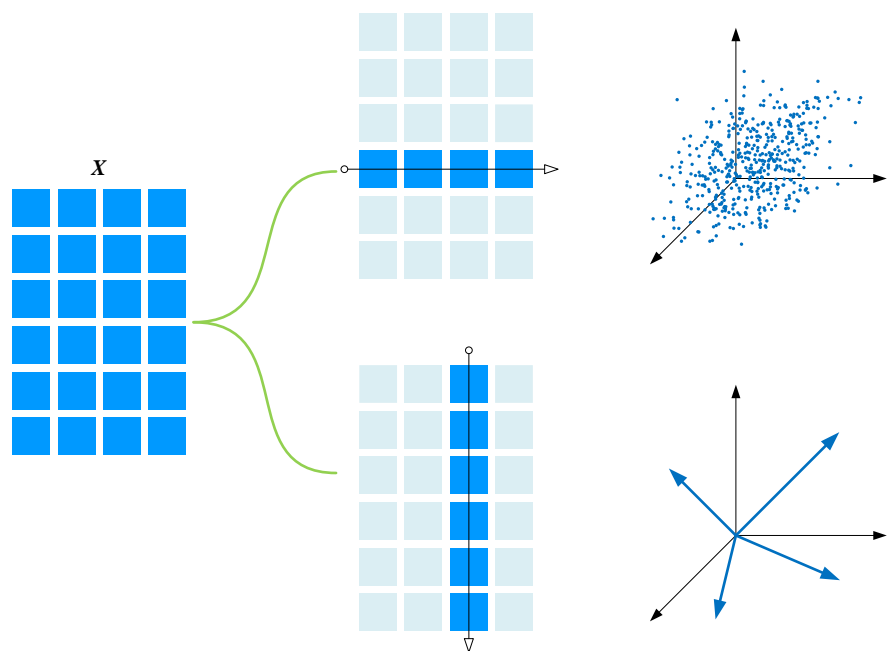
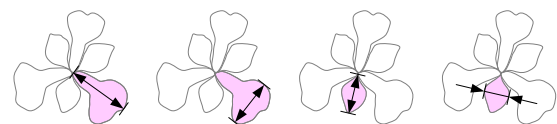


图 9. 观察数据的两个角度



Index	Sepal length $X_1$	Sepal width $X_2$	Petal length $X_3$	Petal width $X_4$	Species $C$
1	5.1	3.5	1.4	0.2	Setosa $C_1$
2	4.9	3	1.4	0.2	
3	4.7	3.2	1.3	0.2	
...	...	...	...	...	
49	5.3	3.7	1.5	0.2	
50	5	3.3	1.4	0.2	Versicolor $C_2$
51	7	3.2	4.7	1.4	
52	6.4	3.2	4.5	1.5	
53	6.9	3.1	4.9	1.5	
...	...	...	...	...	
99	5.1	2.5	3	1.1	Virginica $C_3$
100	5.7	2.8	4.1	1.3	
101	6.3	3.3	6	2.5	
102	5.8	2.7	5.1	1.9	
103	7.1	3	5.9	2.1	
...	...	...	...	...	
149	6.2	3.4	5.4	2.3	
150	5.9	3	5.1	1.8	

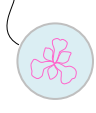




图 10. 鸢尾花数据表格，单位为厘米 (cm)

这些数据都属于鸢尾属下的三个亚属，分别是**山鸢尾** (setosa)、**变色鸢尾** (versicolor) 和**维吉尼亚鸢尾** (virginica)。每一类鸢尾花收集了 50 条样本记录，共计 150 条。

四个特征被用作样本的定量分析，它们分别是**花萼长度** (sepal length)、**花萼宽度** (sepal width)、**花瓣长度** (petal length) 和**花瓣宽度** (petal width)。

图 10 所示为鸢尾花数据集中部分数据。数据整体可以看做是一个矩阵，矩阵的每一列为鸢尾花一个特征的数据样本，矩阵的每一行为某一朵鸢尾花不同特征的数值。

图 11 所示为 9 只股票在 2020 年股价的部分数据。

图 11 所示的数据也是一个矩阵。从向量的角度，数据的每一行是一个行向量，代表一天的收盘价格。数据的每一列是列向量，代表一支股票在一年内的走势。

再次强调，不要被矩阵吓到，矩阵就是一个数据表格。而这个表格可以划分成若干行、若干列，这就对应本章介绍的行向量、列向量。

Date	TSLA	TSM	COST	NVDA	FB	AMZN	AAPL	NFLX	GOOGL
2-Jan-2020	86.05	58.26	281.10	239.51	209.78	1898.01	74.33	329.81	1368.68
3-Jan-2020	88.60	56.34	281.33	235.68	208.67	1874.97	73.61	325.90	1361.52
6-Jan-2020	90.31	55.69	281.41	236.67	212.60	1902.88	74.20	335.83	1397.81
7-Jan-2020	93.81	56.60	280.97	239.53	213.06	1906.86	73.85	330.75	1395.11
8-Jan-2020	98.43	57.01	284.19	239.98	215.22	1891.97	75.04	339.26	1405.04
9-Jan-2020	96.27	57.48	288.75	242.62	218.30	1901.05	76.63	335.66	1419.79
10-Jan-2020	95.63	57.12	286.65	243.92	218.06	1883.16	76.80	329.05	1428.96
13-Jan-2020	104.97	58.28	289.18	251.56	221.91	1891.30	78.44	338.92	1440.03
14-Jan-2020	107.58	58.54	289.07	246.87	219.06	1869.44	77.39	338.69	1430.59
15-Jan-2020	103.70	56.66	290.10	245.17	221.15	1862.02	77.05	339.07	1439.20
16-Jan-2020	102.70	57.01	292.23	248.52	221.77	1877.94	78.02	338.62	1450.16
17-Jan-2020	102.10	56.85	293.82	248.87	222.14	1864.72	78.88	339.67	1479.52
...	...	...	...	...	...	...	...	...	...
30-Dec-2020	694.78	108.49	373.71	525.83	271.87	3285.85	133.52	524.59	1736.25
31-Dec-2020	705.67	108.63	376.04	522.20	273.16	3256.93	132.49	540.73	1752.64

图 11. 股票收盘股价数据

## 1.2 行向量：一行多列，一个样本数据点

如上一节所述，行向量是一个  $1 \times n$  的矩阵，即 1 行、 $n$  列。如图 12 所示，行向量**转置** (transpose) 得到列向量，反之亦然。转置运算符号为正体上标  $T$ 。

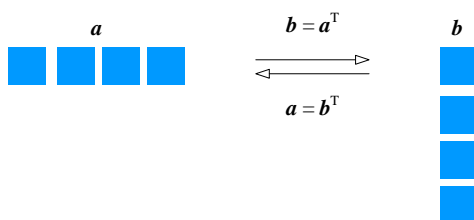


图 12. 行向量的转置结果是一个列向量

表 1 所示为利用 Numpy 构造行向量几种常见方法。可以用 `len(a)` 计算向量的长度，即向量中元素个数。

表 1. 构造行向量

代码	注意事项
<code>a = numpy.array([4,3])</code>	严格地说，这种方法产生的并不是行向量；运行 <code>a.ndim</code> 发现 <code>a</code> 只有一个维度；因此，转置 <code>numpy.array([4,3]).T</code> 得到的仍然是一维序列，只不过默认展示方式为行向量
<code>a = numpy.array([[4,3]])</code>	运行 <code>a.ndim</code> 发现 <code>a</code> 有二个维度；这个行向量转置 <code>a.T</code> 可以获得列向量； <code>a.T</code> 求 <code>a</code> 转置，等价于 <code>a.transpose()</code>
<code>a = numpy.array([4,3], ndmin=2)</code>	<code>ndmin=2</code> 设定数据有两个维度；转置 <code>a.T</code> 可以获得列向量
<code>a = numpy.r_['r', [4,3]]</code>	<code>numpy.r_[]</code> 将一系列的序列合并；'r' 设定结果以行向量（默认）展示，比如 <code>numpy.r_[numpy.array([1,2]), 0, 0, numpy.array([4,5])]</code> 默认产生行向量
<code>a = numpy.array([4,3]).reshape((1, -1))</code>	<code>reshape()</code> 按某种形式重新排列数据；-1 自动获取序列长度 <code>n</code>
<code>a = numpy.array([4, 3])[None, :]</code>	按照 <code>[None, :]</code> 形式广播序列； <code>None</code> 代表 <code>numpy.newaxis</code> ，增加新维度
<code>a = numpy.array([4, 3])[numpy.newaxis, :]</code>	等同于上一例

### 行向量和矩阵

丛书常用  $X$  表达数据矩阵， $X$  的每一行代表一个数据点。为了方便区分，构造  $X$  的一系列行向量序号采用“上标加括号”表达，比如  $\mathbf{x}^{(1)}$  代表  $X$  的第一行。

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)

如图 13 所示，矩阵  $X$  可以写作：

$$X = \begin{bmatrix} \mathbf{x}^{(1)} \\ \mathbf{x}^{(2)} \\ \vdots \\ \mathbf{x}^{(6)} \end{bmatrix} \quad (3)$$

再次强调，数据分析偏爱用行向量表达坐标点。

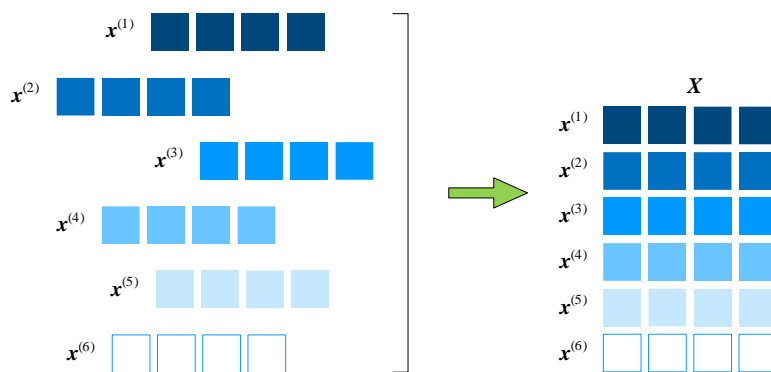


图 13. 矩阵由一系列行向量构造

### 以鸢尾花为例

图 14 所示为鸢尾花数据。前文提到，每一行为一朵鸢尾花样本花萼长度、花萼宽度、花瓣长度和花瓣宽度测量结果。 $\mathbf{x}^{(15)}$  代表第 15 个样本数据点，也就是编号为 15 的鸢尾花样本。

Index	Sepal length	Sepal width	Petal length	Petal width
1	5.1	3.5	1.4	0.2
2	4.9	3	1.4	0.2
3	4.7	3.2	1.3	0.2
4	4.6	3.1	1.5	0.2
5	5	3.6	1.4	0.2
...	...	...	...	...
150	5.9	3	5.1	1.8

图 14. 鸢尾花数据，行向量代表样本数据点

## 数据云、投影

取出鸢尾花前两个特征——花萼长度、花萼宽度——对应的数据。把它们以坐标的形式画在平面直角坐标系（记做  $\mathbb{R}^2$ ）中，我们便得到平面散点图。如图 15 所示，这个图好比“样本数据云”。

图 15 中数据点 (5.0, 2.0) 可以写成行向量 [5.0, 2.0]。它的横坐标为 5.0，纵坐标为 2.0。

数据点 (5.0, 2.0) 是序号为 61 的样本点，对应的行向量可以写成  $\mathbf{x}^{(61)}$ 。

从集合视角来看，(5.0, 2.0) 属于  $\mathbb{R}^2$ ，即  $(5.0, 2.0) \in \mathbb{R}^2$ 。图 15 中整团数据云都属于  $\mathbb{R}^2$ 。

从统计视角来看，我们可以计算样本数据各个特征的均值 ( $\mu_1$  和  $\mu_2$ )，并找到质心 ( $\mu_1, \mu_2$ ) 在图 15 中的位置。我们还可计算不同特征上样本数据的均方差 ( $\sigma_1$  和  $\sigma_2$ )，并在图中标注  $\mu_1 \pm \sigma_1$ 、 $\mu_1 \pm 2\sigma_1$ 、 $\mu_2 \pm \sigma_2$ 、 $\mu_2 \pm 2\sigma_2$  对应的位置。

从几何视角来看，(5.0, 2.0) 在横轴的正交投影 (orthogonal projection) 结果为 5.0。(5.0, 2.0) 在纵轴的正交投影结果为 2.0。正交投影很好理解，即原来数据点和投影点连线垂直于投影点所在直线或平面。打个比方，头顶正上方太阳光将自己身体的影子投影在地面，阳光光线垂直于地面。不特别强调的话，本书的投影均指正交投影。

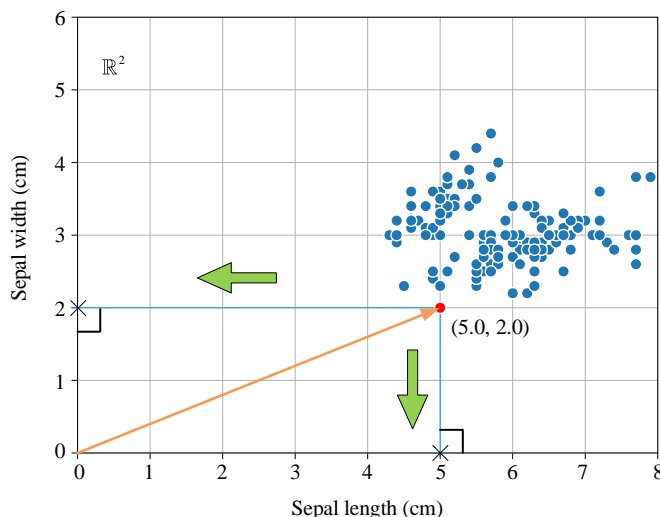


图 15. 鸢尾花前两个特征数据散点图

再进一步，将整团数据云全部正交投影在横轴，得到图 16。图 16 中  $\times$  代表的的数据实际上就是鸢尾花数据集第一列花萼长度数据点。图 16 中横轴相当于一个一维空间，即一条数轴  $\mathbb{R}$ 。

我们也可以把整团数据云全部投影在纵轴，得到图 17。图中的  $\times$  是鸢尾花数据第二列花萼宽度数据。

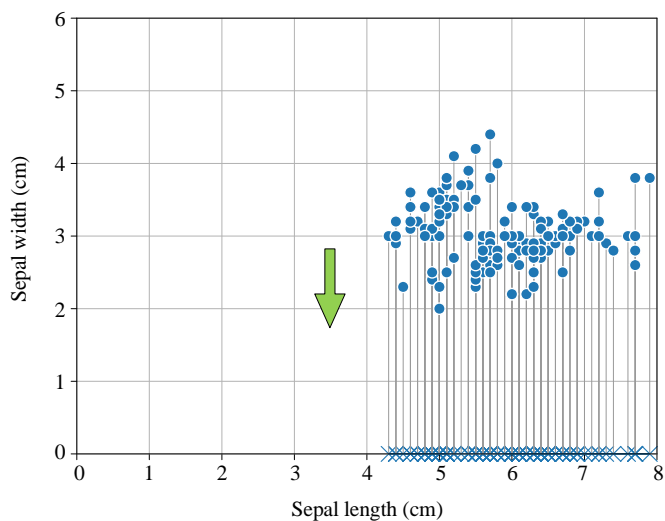


图 16. 二维散点正交投影到横轴

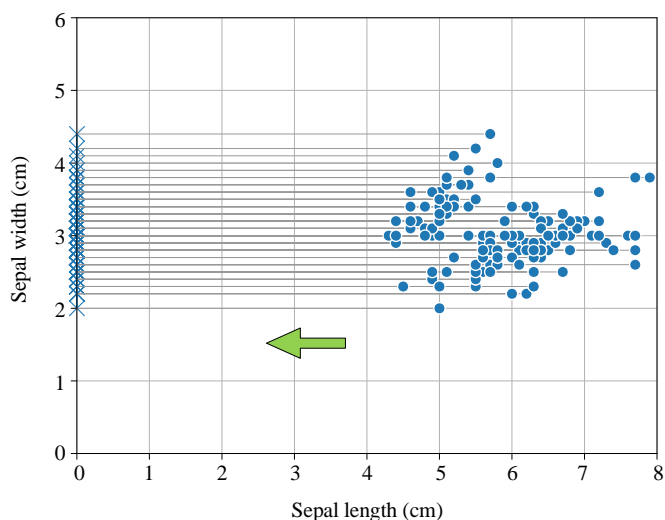


图 17. 二维散点正交投影到纵轴

你可能会问，是否可以将图 16 中所有点投影在一条斜线上？

答案是肯定的。

如图 18 所示，鸢尾花数据投影到一条斜线上，这条斜线通过原点和横轴夹角  $15^\circ$ 。大家可能会问，怎么计算图 18 中投影点的坐标？这种几何变换有何用途？这是本书后续要深入讲解的内容。

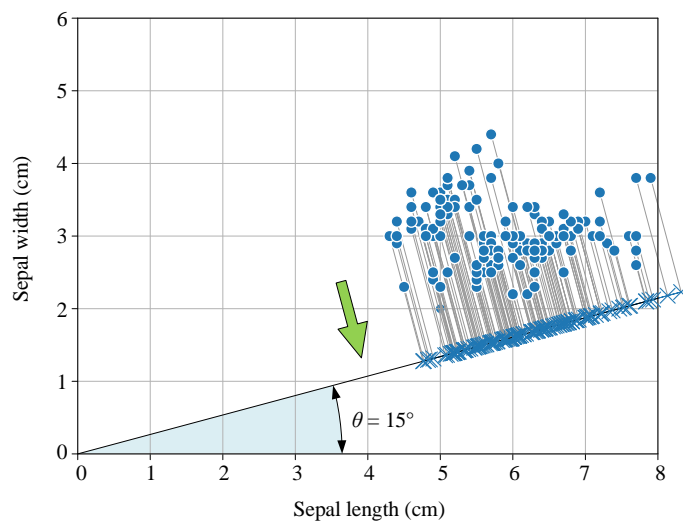


图 18. 二维散点正交投影到一条斜线

取出鸢尾花前三个特征 (花萼长度、花萼宽度、花瓣长度) 对应的数据, 并在  $\mathbb{R}^3$  绘制散点图, 得到图 19。图 15 相当于图 19 在水平面 (浅蓝色背景) 正交投影结果。

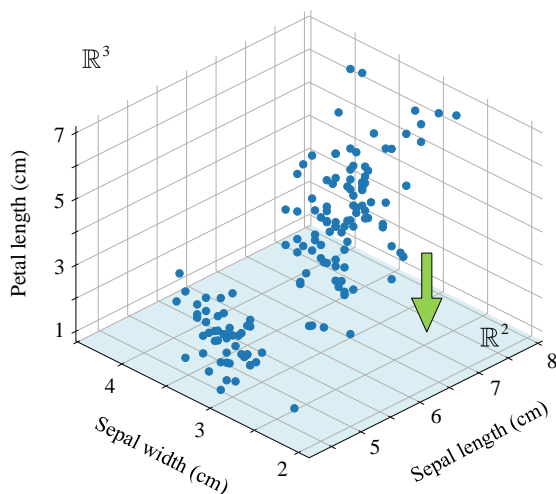


图 19. 鸢尾花前三个特征数据散点图

回顾本系列丛书《数学要素》一册介绍过的成对特征散点图, 具体如图 20 所示。成对特征散点图不但可以可视化鸢尾花四个特征 (花萼长度、花萼宽度、花瓣长度和花瓣宽度), 通过颜色还可以展示鸢尾花三个类别 (山鸢尾、变色鸢尾、维吉尼亚鸢尾)。图 20 中的二维散点图相当于四维空间散点在不同平面上的投影结果。

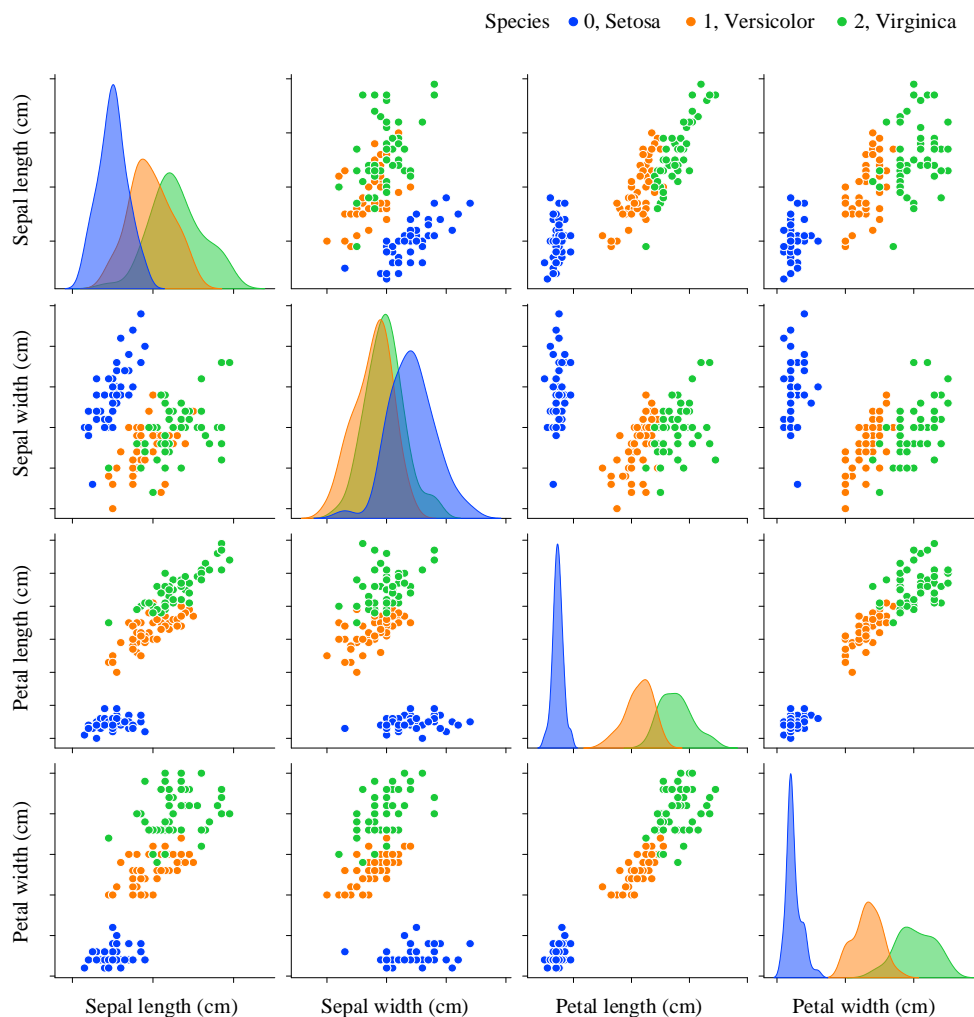


图 20. 鸢尾花数据成对特征散点图，考虑分类标签

## 1.3 列向量：一行多行，一个特征

列向量是一个  $n \times 1$  的矩阵，即  $n$  行、1 列。列向量转置得到行向量。

数据矩阵  $X$  的每一列代表一个特征，因此列向量又常称作**特征向量** (feature vector)；注意，此特征向量不同于特征值分解中的**特征向量** (eigenvector)。

同样，为了方便区分，构造  $X$  的列向量序号采用下标表达，比如  $x_1$ 。如图 21 所示，矩阵  $X$  看做是 4 个等长列向量整齐排列得到：

$$X = [x_1 \quad x_2 \quad x_3 \quad x_4] \quad (4)$$

数据分析偏爱列向量 ( $x_j$ ) 表达特征，它对应概率统计的某个随机变量 ( $X_j$ )，或者代数中的变量 ( $x_j$ )。

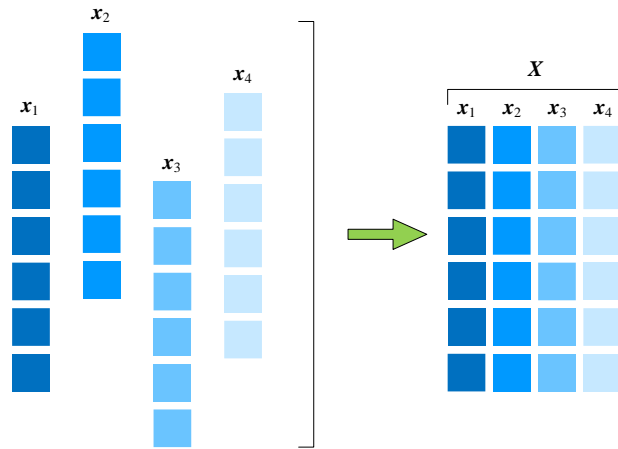


图 21. 矩阵由一排列向量构造

表 2 总结 Numpy 构造列向量几种常见方法。

表 2. 构造列向量

代码	注意事项
<code>a = numpy.array([[4], [3]])</code>	运行 <code>a.ndim</code> 发现 <code>a</code> 有二个维度; <code>numpy.array([[4], [3]]).T</code> 获得行向量
<code>a = numpy.r_['c', [4,3]]</code>	<code>numpy.r_[]</code> 将一系列的序列合并; 'c' 设定结果以列向量展示, 比如 <code>np.r_['c', numpy.array([1,2]), 0, 0, numpy.array([4,5])]</code> 默认产生行向量
<code>a = numpy.array([4,3]).reshape((-1, 1))</code>	<code>reshape()</code> 按某种形式重新排列数据; -1 自动获取序列长度 <code>n</code>
<code>a = numpy.array([4, 3])[ :, None]</code>	按照 <code>[ :, None]</code> 形式广播序列; <code>None</code> 代表 <code>numpy.newaxis</code> , 增加新维度
<code>a = numpy.array([4, 3])[ :, numpy.newaxis]</code>	等同于上一例

图 22 所示鸢尾花数据每一列代表鸢尾花的一个特征, 比如花萼长度 (第 1 列, 列向量  $x_1$ )、花萼宽度 (第 2 列, 列向量  $x_2$ )、花瓣长度 (第 3 列, 列向量  $x_3$ ) 和花瓣宽度 (第 4 列, 列向量  $x_4$ )。



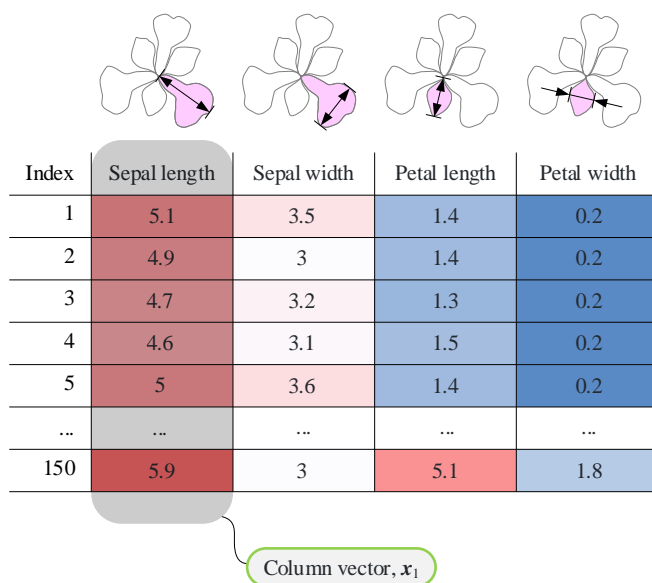


图 22. 鸢尾花数据，列向量代表数据特征

大家可能会问， $\mathbf{x}_1$ 、 $\mathbf{x}_2$ 、 $\mathbf{x}_3$ 、 $\mathbf{x}_4$ 这四个向量到底意味着什么？有没有什么办法可视化这四个列向量？怎么量化它们之间的关系？答案会在本书后续揭晓。

### 特殊列向量

**全零列向量** (zero column vector)  $\mathbf{0}$ ，是指每个元素均为 0 的列向量：

$$\mathbf{0} = [0 \ 0 \ \dots \ 0]^T \quad (5)$$

代码 `numpy.zeros((4, 1))` 可以生成  $4 \times 1$  全 0 列向量。

**全 1 列向量** (all-ones column vector)  $\mathbf{1}$ ，是指每个元素均为 1 的列向量：

$$\mathbf{1} = [1 \ 1 \ \dots \ 1]^T \quad (6)$$

代码 `numpy.ones((4, 1))` 可以生成  $4 \times 1$  全 1 列向量。

再次强调，一般情况，本书默认向量为列向量，除非具体说明。

## 1.4 向量长度：模，欧氏距离， $L^2$ 范数

**向量长度** (length of a vector) 又叫做**向量模** (vector norm)、**欧几里得距离** (Euclidean distance)、**欧几里得范数** (Euclidean norm) 或  **$L^2$  范数** ( $L^2$ -norm)。

定义向量  $\mathbf{a}$ :

$$\mathbf{a} = [a_1 \ a_2 \ \cdots \ a_n]^T \quad (7)$$

向量  $\mathbf{a}$  的模为:

$$\|\mathbf{a}\| = \|\mathbf{a}\|_2 = \sqrt{a_1^2 + a_2^2 + \cdots + a_n^2} = \left( \sum_{i=1}^n a_i^2 \right)^{\frac{1}{2}} \quad (8)$$

注意,  $\|\mathbf{a}\|_2$  下角标 2, 代表  $L^2$  范数; 没有特殊说明,  $\|\mathbf{a}\|$  默认代表  $L^2$  范数。 $L^2$  范数是  $L^p$  范数的一种, 本书后续还要介绍其他各种范数。

## 二维向量的模

特别地, 对于如下二维向量:

$$\mathbf{a} = [a_1 \ a_2]^T \quad (9)$$

二维向量  $\mathbf{a}$  的  $L^2$  范数为:

$$\|\mathbf{a}\| = \sqrt{a_1^2 + a_2^2} \quad (10)$$

图 23 给出向量  $\mathbf{a}$  和  $\mathbf{b}$ , 它们的模可以这样计算得到:

$$\begin{aligned} \|\mathbf{a}\| &= \sqrt{4^2 + 3^2} = \sqrt{25} = 5 \\ \|\mathbf{b}\| &= \sqrt{(-3)^2 + 4^2} = \sqrt{25} = 5 \end{aligned} \quad (11)$$

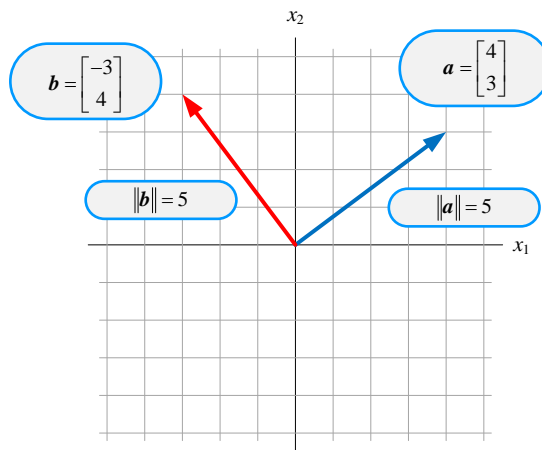


图 23. 向量  $\mathbf{a}$  和  $\mathbf{b}$  的模

二维向量  $\mathbf{a}$  和横轴夹角可以通过反正切求解:

$$\theta_a = \arctan\left(\frac{a_2}{a_1}\right) \quad (12)$$

本书下一章会介绍如何用夹角余弦值反求这个夹角。

Bk4\_Ch1\_01.py 绘制图 23 所示向量。matplotlib.pyplot.quiver() 绘制箭头图。



```
# Bk4_Ch1_01.py

import numpy as np
import matplotlib.pyplot as plt

def draw_vector(vector, RGB):
    array = np.array([[0, 0, vector[0], vector[1]]])
    X, Y, U, V = zip(*array)
    plt.quiver(X, Y, U, V, angles='xy', scale_units='xy', scale=1, color = RGB)

fig, ax = plt.subplots()

draw_vector([4,3], np.array([0,112,192])/255)
draw_vector([-3,4], np.array([255,0,0])/255)

plt.ylabel('$x_2$'); plt.xlabel('$x_1$'); plt.axis('scaled')
ax.set_xlim([-5, 5]); ax.set_ylim([-5, 5])
ax.grid(linestyle='--', linewidth=0.25, color=[0.5,0.5,0.5])
plt.show()
```

函数 numpy.linalg.norm() 默认计算  $L^2$  范数; 也可以用 numpy.sqrt(np.sum(a\*\*2)) 计算向量  $\mathbf{a}$  的  $L^2$  范数。Bk4\_Ch1\_02.py 计算图 23 中向量  $\mathbf{a}$  和  $\mathbf{b}$  模。



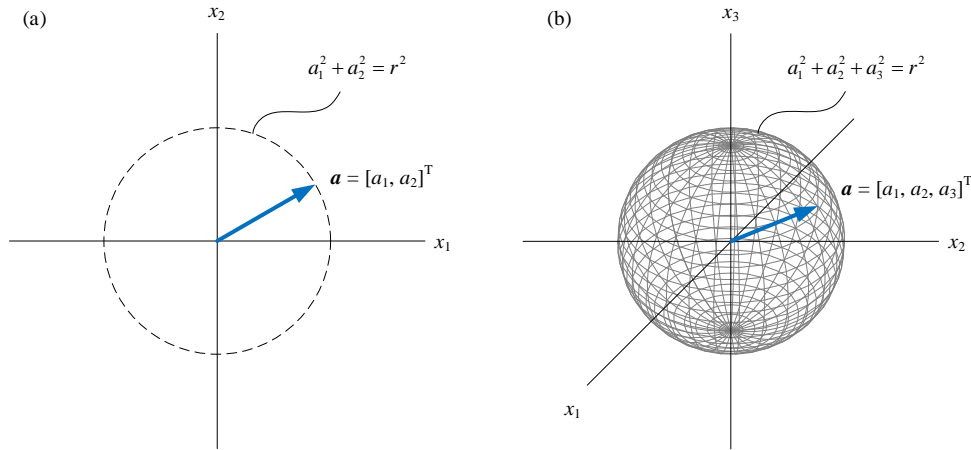
```
# Bk4_Ch1_02.py

import numpy as np

# define two column vectors
a = np.array([[4], [3]])
b = np.array([[-3], [4]])

# calculate L2 norm
a_L2_norm = np.linalg.norm(a)
b_L2_norm = np.linalg.norm(b)
```

值得一提的是, 和  $\|\mathbf{a}\|$  等长的二维向量, 如果起点重合, 它们的终点位于同一个圆上, 如图 24 (a) 所示。

图 24. 等  $L^2$  范数向量

### 三维向量的模

类似地，对于三维向量：

$$\mathbf{a} = [a_1 \ a_2 \ a_3]^T \quad (13)$$

三维向量  $\mathbf{a}$  的  $L^2$  范数为：

$$\|\mathbf{a}\| = \sqrt{a_1^2 + a_2^2 + a_3^2} \quad (14)$$

图 24 (b) 所示为起点相同的等长三维向量终点位于同一正圆球面上。

### 单位向量

长度为 1 的向量被称作**单位向量** (unit vector)。

向量  $\mathbf{a}$  除以自身的模得到  **$\mathbf{a}$  方向上的单位向量** (unit vector in the direction of vector  $\mathbf{a}$ ):

$$\hat{\mathbf{a}} = \frac{\mathbf{a}}{\|\mathbf{a}\|} \quad (15)$$

$\hat{\mathbf{a}}$  读作 “vector a hat”。`a/numpy.linalg.norm(a)` 可以计算向量  $\mathbf{a}$  方向上的单位向量。

图 25 (a) 所示平面直角坐标系，起点位于原点的单位向量  $\mathbf{x} = [x_1, x_2]^T$  终点位于**单位圆** (unit circle) 上：

$$\|\mathbf{x}\| = \sqrt{x_1^2 + x_2^2} = 1 \Rightarrow x_1^2 + x_2^2 = 1 \quad (16)$$

如图 25 (b) 所示平面直角坐标系中， $\mathbf{e}_1$  ( $\mathbf{i}$ ) 和  $\mathbf{e}_2$  ( $\mathbf{j}$ ) 分别为沿着  $x_1$  (水平) 和  $x_2$  (竖直) 方向的单位向量：

$$\mathbf{e}_1 = \mathbf{i} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \quad \mathbf{e}_2 = \mathbf{j} = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \quad (17)$$

显然， $\mathbf{e}_1$  和  $\mathbf{e}_2$  相互垂直。

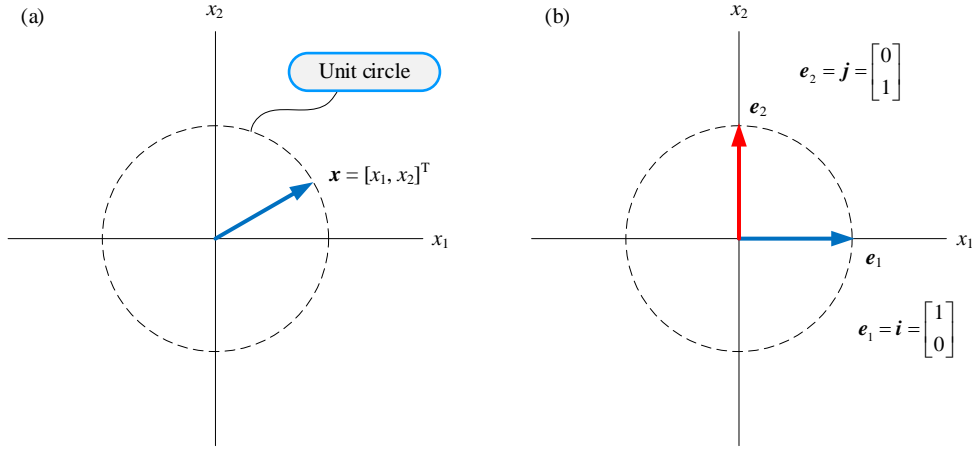


图 25. 单位向量

图 23 给出向量  $\mathbf{a}$  和  $\mathbf{b}$  可以用  $\mathbf{e}_1$  和  $\mathbf{e}_2$  表达：

$$\begin{aligned} \mathbf{a} &= 4\mathbf{e}_1 + 3\mathbf{e}_2 \\ \mathbf{b} &= -3\mathbf{e}_1 + 4\mathbf{e}_2 \end{aligned} \quad (18)$$

(18) 用到的便是向量加减法，这是下一节要介绍的内容。

图 23 这个平面就是用  $\mathbf{e}_1$  和  $\mathbf{e}_2$  张成的。白话说， $\mathbf{e}_1$  和  $\mathbf{e}_2$  张成  $\mathbb{R}^2$  这个平面， $\mathbb{R}^2$  平面上的点都可以表达成：

$$\mathbf{x} = x_1\mathbf{e}_1 + x_2\mathbf{e}_2 \quad (19)$$

而上式中的  $(x_1, x_2)$  向量终点在平面上的坐标点。从集合运算角度， $\mathbf{x} \in \mathbb{R}^2$ 。

### 三维直角坐标系

三维直角坐标系中， $\mathbf{e}_1$  ( $\mathbf{i}$ )、 $\mathbf{e}_2$  ( $\mathbf{j}$ ) 和  $\mathbf{e}_3$  ( $\mathbf{k}$ ) 分别为沿着  $x_1$ 、 $x_2$  和  $x_3$  单位向量：

$$\mathbf{e}_1 = \mathbf{i} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \quad \mathbf{e}_2 = \mathbf{j} = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}, \quad \mathbf{e}_3 = \mathbf{k} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \quad (20)$$

$\mathbf{e}_1$  ( $\mathbf{i}$ )、 $\mathbf{e}_2$  ( $\mathbf{j}$ ) 和  $\mathbf{e}_3$  ( $\mathbf{k}$ ) 两两相互垂直。

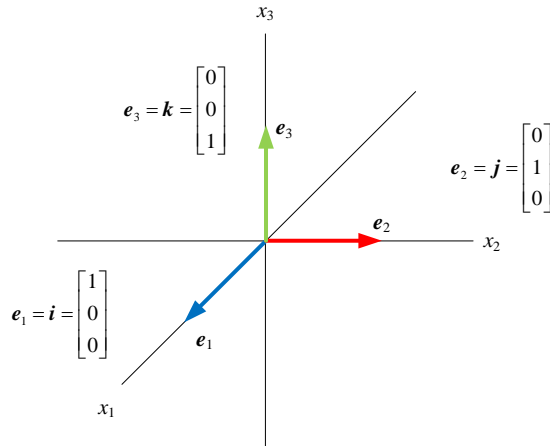


图 26. 三维空间单位向量

同理，图 26 这个三维空间是用  $e_1$ 、 $e_2$ 、 $e_3$  张成的。白话说， $e_1$ 、 $e_2$ 、 $e_3$  撑起  $\mathbb{R}^3$  这个三维空间，空间的点都可以表达成：

$$\mathbf{x} = x_1 \mathbf{e}_1 + x_2 \mathbf{e}_2 + x_3 \mathbf{e}_3 \quad (21)$$

而上式中的  $(x_1, x_2, x_3)$  向量终点在平面上的坐标点。

此外，大家可能已经注意到， $e_1$  可以用不同的形式表达，比如：

$$\mathbf{e}_1 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \quad \mathbf{e}_1 = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \quad \mathbf{e}_1 = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \quad \mathbf{e}_1 = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \quad (22)$$

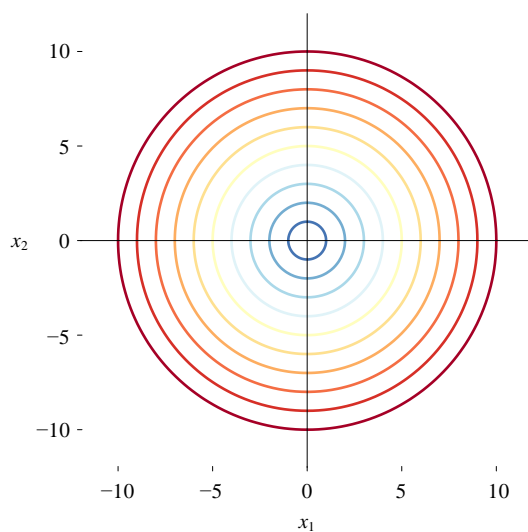
上式中几个  $e_1$  等价，相当于在不同维度空间中的  $e_1$ 。这些  $e_1$  之间是从低维到高维，或从高维到低维投影。本书后续将在投影部分深入探讨。

### 等距线：同心圆

如图 27 所示，向量  $\mathbf{x}$  的模长度  $\|\mathbf{x}\|$  取得不同数值时，我们可以得到一系列同心圆：

$$\|\mathbf{x}\| = \sqrt{x_1^2 + x_2^2} = c \quad (23)$$

这个向量模就是欧几里得距离，也叫欧氏距离。图 27 中等高线这就是我们在本系列丛书《数学要素》一册中讲到的等距线。

图 27. 相等  $L^2$  范数向量终点位于一系列同心圆等高线上

Bk4\_Ch1\_03.py 绘制图 27。



```
# Bk4_Ch1_03.py
import matplotlib.pyplot as plt
import numpy as np

x1 = np.linspace(-10, 10, num=201);
x2 = x1;

xx1, xx2 = np.meshgrid(x1, x2)
p = 2
zz = ((np.abs((xx1))**p) + (np.abs((xx2))**p))**(1./p)

fig, ax = plt.subplots(figsize=(12, 12))

ax.contour(xx1, xx2, zz, levels = np.arange(11), cmap='RdYlBu_r')

ax.axhline(y=0, color='k', linewidth = 0.25)
ax.axvline(x=0, color='k', linewidth = 0.25)
ax.set_xlim(-12, 12)
ax.set_ylim(-12, 12)
ax.spines['top'].set_visible(False)
ax.spines['right'].set_visible(False)
ax.spines['bottom'].set_visible(False)
ax.spines['left'].set_visible(False)
ax.set_xlabel('$x_1$')
ax.set_ylabel('$x_2$')
ax.set_aspect('equal', adjustable='box')
plt.show()
```

## 1.5 加减法：对应位置元素分别相加减

从数据角度看，两个等长列向量相加，结果为对应位置元素分别相加，得到长度相同的列向量，比如下例：

$$\mathbf{a} + \mathbf{b} = \begin{bmatrix} -2 \\ 5 \end{bmatrix} + \begin{bmatrix} 5 \\ -1 \end{bmatrix} = \begin{bmatrix} -2+5 \\ 5-1 \end{bmatrix} = \begin{bmatrix} 3 \\ 4 \end{bmatrix} \quad (24)$$

两个等长列向量相减，则是对应元素分别相减，得到等长列向量，比如下例：

$$\mathbf{a} - \mathbf{b} = \begin{bmatrix} -2 \\ 5 \end{bmatrix} - \begin{bmatrix} 5 \\ -1 \end{bmatrix} = \begin{bmatrix} -2-5 \\ 5-(-1) \end{bmatrix} = \begin{bmatrix} -7 \\ 6 \end{bmatrix} \quad (25)$$

该原理也适用于等长行向量加减法。

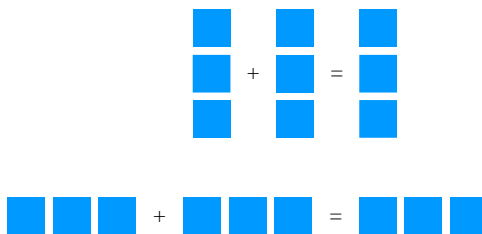
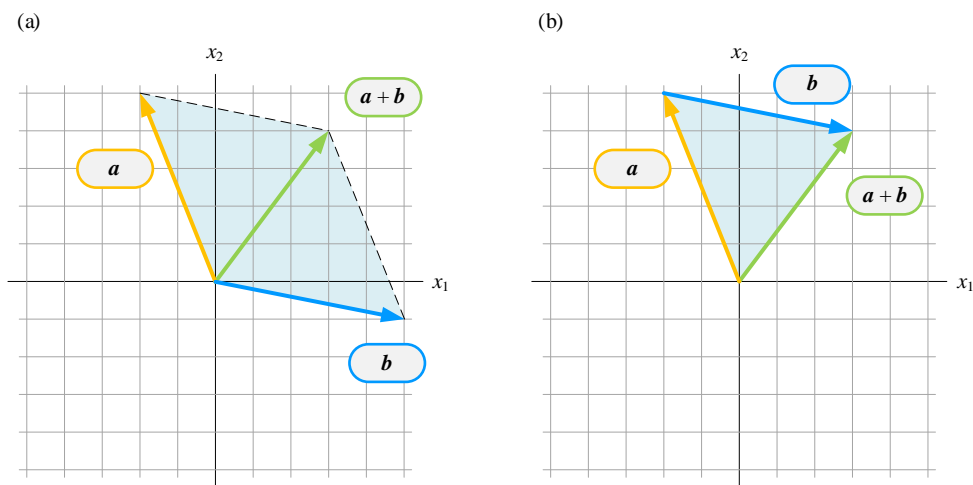


图 28. 数据角度看向量加法

### 几何视角

从几何角度看，**向量加法** (vector addition) 结果可以用**平行四边形法则** (parallelogram method) 或**三角形法则** (triangle method) 获得，具体如图 29 所示。



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)



图 29. 几何角度看向量加法

**向量减法** (vector subtraction), 向量  $a$  减去向量  $b$ , 可以将向量  $b$  换向得到  $-b$ ; 然后再计算向量  $-b$  与向量  $a$  的和, 即:

$$a - b = a + (-b) = \begin{bmatrix} -2 \\ 5 \end{bmatrix} - \begin{bmatrix} 5 \\ -1 \end{bmatrix} = \begin{bmatrix} -7 \\ 6 \end{bmatrix} \quad (26)$$

注意,  $a - b$  和  $b - a$  结果相反:

$$b - a = -(a - b) = \begin{bmatrix} 5 \\ -1 \end{bmatrix} - \begin{bmatrix} -2 \\ 5 \end{bmatrix} = \begin{bmatrix} 7 \\ -6 \end{bmatrix} \quad (27)$$

向量  $a$  减去向量  $b$ , 得到的结果  $a - b$  对应向量箭头, 指向  $a$  终点; 相反, 向量  $b$  减去向量  $a$  得到的结果  $b - a$ , 指向  $b$  终点。

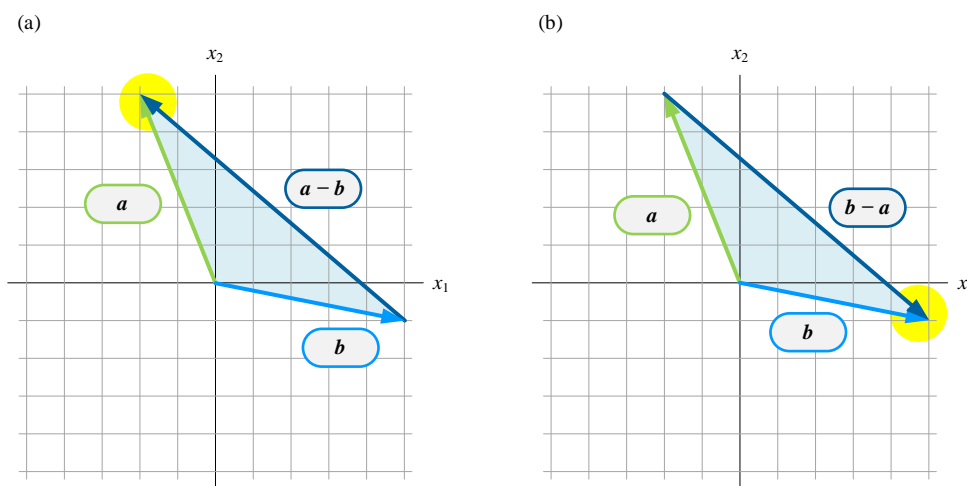


图 30. 几何角度向量减法

请大家注意以下性质:

$$\begin{aligned} a + b &= b + a \\ (a + b) + c &= a + (b + c) \\ a + (-a) &= 0 \end{aligned} \quad (28)$$

两个向量相同, 当且仅当两者大小方向均相同。如果两个向量的大小相同但是方向相反, 两者互为反向量。两个向量方向相同或相反, 则称向量平行。

Bk4\_Ch1\_04.py 计算本节向量加减法示例。



| # Bk4\_Ch1\_04.py

```
import numpy as np

# define two column vectors
a = np.array([[ -2], [ 5]])
b = np.array([[ 5], [-1]])

# calculate vector addition
a_plus_b = a + b
a_plus_b_2 = np.add(a,b)

# calculate vector subtraction
a_minus_b = a - b
a_minus_b_2 = np.subtract(a,b)

b_minus_a = b - a
b_minus_a_2 = np.subtract(b,a)
```

## 1.6 标量乘法：向量缩放

**向量标量乘法** (scalar multiplication of vectors) 指的是标量和向量每个元素分别相乘，结果仍为向量。

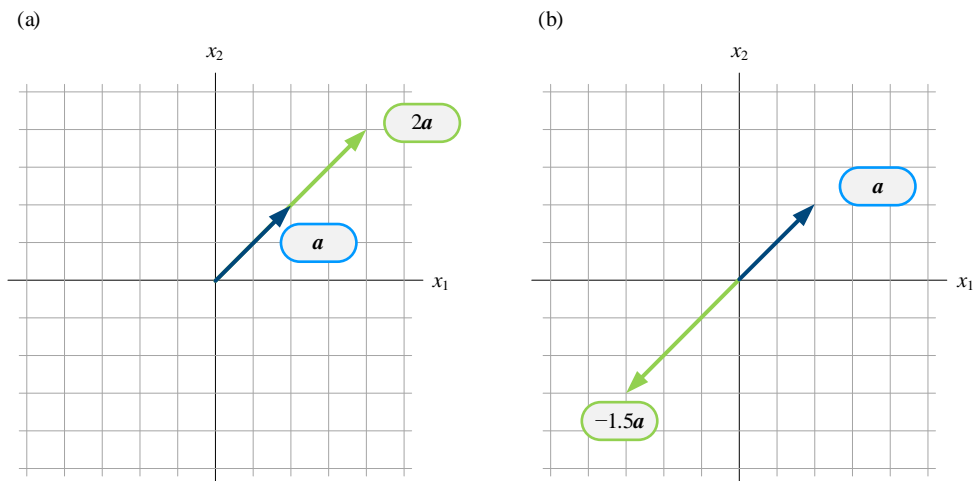


图 31. 向量标量乘法

从几何角度，标量乘法将向量按标量大小成比例缩放，向量方向同向或反向，如图 31 所示。

Bk4\_Ch1\_05.py 完成图 31 中运算。



```
# Bk4_Ch1_05.py
import numpy as np

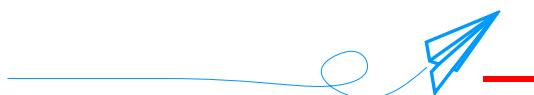
# define a column vector
a = np.array([[ 2], [ 2]])
```

```
b = 2*a
c = -1.5*a
```

请读者注意以下标量乘法性质：

$$\begin{aligned}
 (t+k)a &= ta + ka \\
 t(a+b) &= ta + tb \\
 t(ka) &= tka \\
 1a &= a \\
 -1a &= -a \\
 0a &= \mathbf{0}
 \end{aligned} \tag{29}$$

其中， $t$  和  $k$  为标量。



最后用四幅图来总结本章核心内容。学完本章，希望读者看到任何向量，可以试着从几何、数据两个角度来思考问题。

从几何角度，向量是既有长度又有方向的量。从数据角度，表格数据可以看做是矩阵。

矩阵的每一行向量是一个样本点，每一列向量代表一个特征。提到向量模、 $L^2$  范数、欧几里得距离，希望大家能够联想到正圆、正圆球。

用到向量加减法时，想到平行四边形法则和三角形法则。

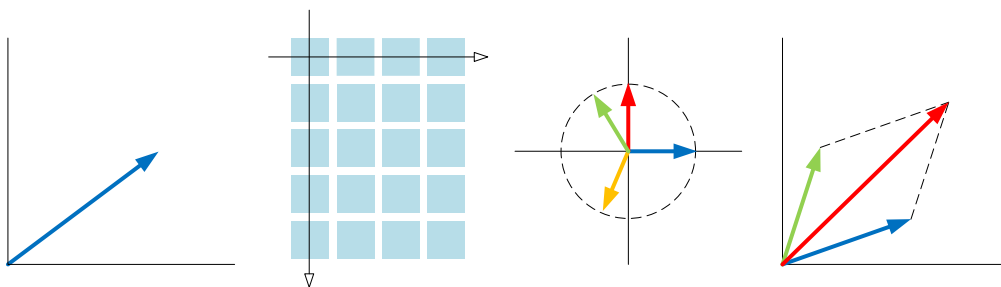


图 32. 总结本章重要内容的四副图