

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

1.5 单位向量



本节你将掌握的核心技能：

- ▶ 向量单位化：掌握将任意非零向量转化为单位向量的标准方法，即除以其 L2 范数。
- ▶ 二维单位向量、单位圆关系：明确 2 维单位向量的终点落在单位圆上。
- ▶ 三维单位向量、单位球关系：理解 3 维单位向量终点分布于单位球面。
- ▶ 向量表达形式：掌握“长度 × 方向向量”的表示方法，便于将方向与大小分离理解。
- ▶ 极坐标：在二维中用极径、极角描述向量位置，理解直角坐标与极坐标互换。
- ▶ 球坐标：球坐标与三维直角坐标的互换方法。

单位向量在向量内积、标量投影、向量投影、极坐标、球坐标、标准正交系、规范正交系、特征值分解、奇异值分解等知识点都有重要的戏份；因此，我们特地拿出一节来介绍单位向量。

非零向量单位

向量单位化 (vector normalization)，也叫向量归一化，是一种用于调整非零向量长度的方法，使向量长度变为 1，同时保持原来的方向。

向量单位化是一种特殊的向量标量乘法。

任意非零 n 维列向量 \mathbf{a} 的单位化操作，是该向量乘以其向量长度 (大小、 L^2 范数、模、欧几里得范数) 的倒数，即

$$\hat{\mathbf{a}} = \frac{\mathbf{a}}{\|\mathbf{a}\|} = \frac{1}{\sqrt{a_1^2 + a_2^2 + \dots + a_n^2}} \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_n \end{bmatrix} \quad (1)$$

结果 \hat{a} 叫做**方向向量** (direction vector)，即**单位向量** (unit vector)，也叫**归一化向量** (normalized vector)。在强调方向性时，本书常用方向向量一词。

方向向量 \hat{a} 和原向量 a 方向相同。

方向向量 \hat{a} 的长度为 1，即 $\|\hat{a}\|=1$ 。

将不同非零向量单位化后并列展示，有助于直观比较它们的方向。特别是有了向量内积之后，我们会发现两个单位向量的内积就是两者夹角的余弦值；这是下一节要讲的内容。

⚠ 注意，零向量 $\mathbf{0}$ 通常被认为没有方向，因为它的模为零，无法确定唯一的方向；但在某些数学背景下，也可以认为零向量 $\mathbf{0}$ 与任意方向平行。

2 维单位向量

对于任意 2 维非零向量 $a = [a_1, a_2]^T$ ，其单位向量为

$$\hat{a} = \frac{a}{\|a\|} = \begin{bmatrix} \frac{a_1}{\sqrt{a_1^2 + a_2^2}} \\ \frac{a_2}{\sqrt{a_1^2 + a_2^2}} \end{bmatrix} \quad (2)$$

举个例子，给定 2 维列向量 $a = [3, 4]^T$ ，它的单位向量为

$$\hat{a} = \frac{a}{\|a\|} = \frac{1}{\sqrt{3^2 + 4^2}} \times \begin{bmatrix} 3 \\ 4 \end{bmatrix} = \begin{bmatrix} 3/5 \\ 4/5 \end{bmatrix} \quad (3)$$

如图 1 所示， $a = [3, 4]^T$ 单位化后，长度变为 1，方向不变。

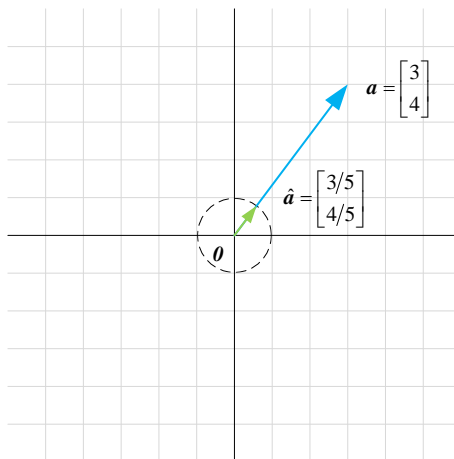


图 1.2 维向量 a 的单位化

如图 2 所示，2 维非零向量单位化后，若起点位于原点，终点恰好**单位圆** (unit circle) 圆周上。

显然, $e_1 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$ 、 $e_2 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$ 也都是单位向量, 分别代表 x_1 轴正方向、 x_2 轴正方向; 而 $-e_1 = \begin{bmatrix} -1 \\ 0 \end{bmatrix}$ 代表 x_1 轴负方向, $-e_2 = \begin{bmatrix} 0 \\ -1 \end{bmatrix}$ 代表 x_2 轴负方向。这四个向量的终点也都在单位圆圆周上。

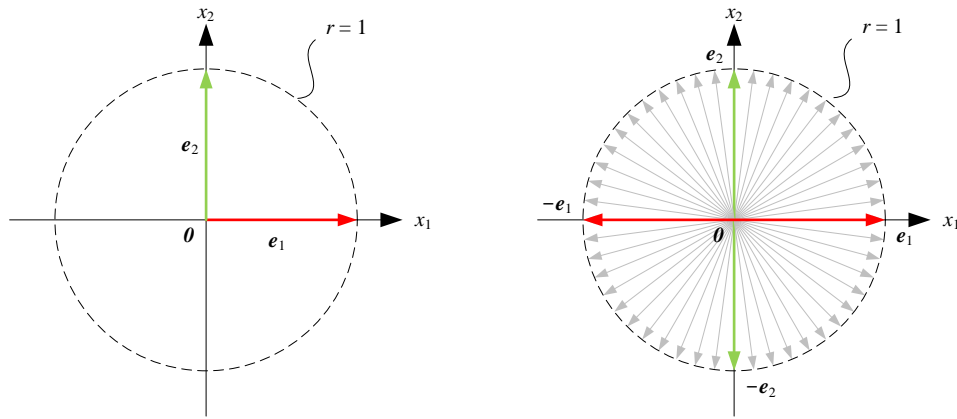


图 2. 平面上的单位向量起点位于原点, 终点位于单位圆圆周上

长度 × 方向向量

反过来看, 非零向量 a 可以写成“长度 × 方向向量”的形式, 即

$$a = \underbrace{\|a\|}_{\text{Length}} \underbrace{\hat{a}}_{\text{Direction}} \quad (4)$$

其中, $\|a\|$ 提供向量大小 (长度、模、L2 范数、欧几里得范数); 方向向量 \hat{a} 提供方向。

比如, 2 维列向量 $a = [3, 4]^T$ 可以写成

$$\begin{bmatrix} 3 \\ 4 \end{bmatrix} = 5 \times \begin{bmatrix} 3/5 \\ 4/5 \end{bmatrix} \quad (5)$$

其中, 5 为向量长度; $[3/5, 4/5]^T$ 为方向向量 (单位向量)。

代码 1 计算单位向量, 请大家自行学习。此外, 请修改代码判断向量 a 是否为零向量。

代码 1. 单位向量 |  LA_01_05_01.ipynb

```

## 初始化
import numpy as np

## 定义向量 a
a = np.array([3, 4])

## 计算向量的长度
norm_a = np.linalg.norm(a)

## 计算单位向量
unit_a = a / norm_a

## 长度  $\times$  方向向量
norm_a * unit_a

```



单位圆、同心圆

如图 3 所示，当向量的起点位于原点，如果一个向量的长度（大小、 L^2 范数、欧几里得范数、模）小于 1（蓝色），那么它的终点落在单位圆内部；如果向量长度大于 1（红色），则终点位于单位圆外部。

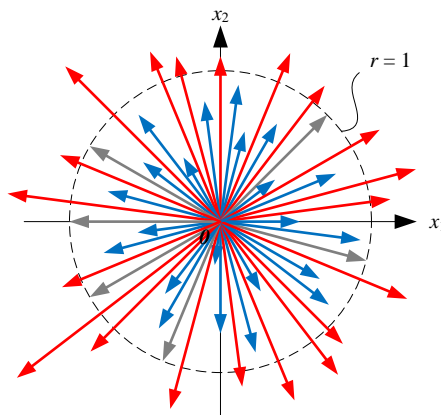


图 3. 终点位于单位圆上、内、外的向量

如图 4 所示，所有从原点出发且长度相同（比如 r ）的向量，其终点都会落在以该向量长度（ r ）为半径的圆周上。

而当向量的长度不同，它们的终点会分布在不同半径的圆上，从而形成一组同心圆。

我们将在本章向量范数一节中展开讲解。

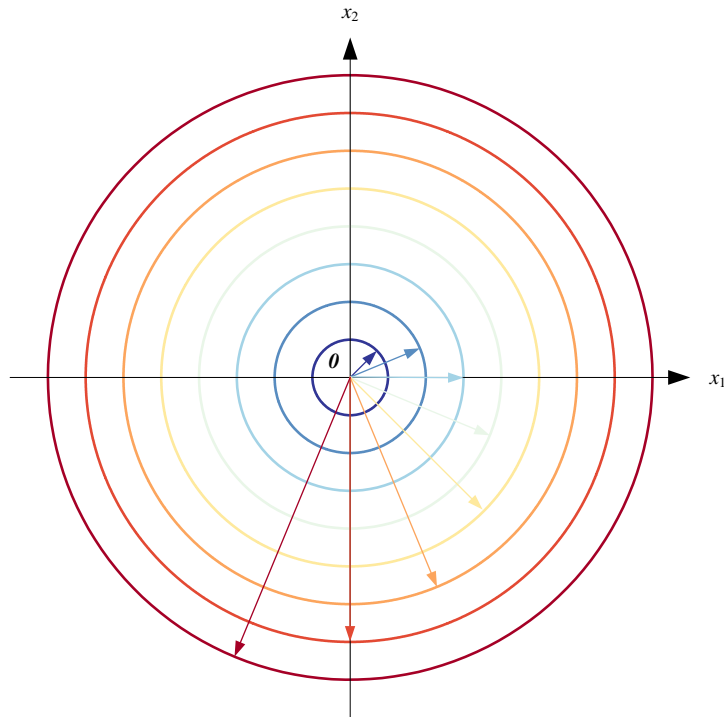


图 4. 不同向量长度的同心圆

投影

在直角坐标系中，横坐标 a_1 实际上是向量 $\mathbf{a} = \begin{bmatrix} a_1 \\ a_2 \end{bmatrix}$ 在水平方向上的投影，而纵坐标 a_2 是 \mathbf{a} 在竖直方向上的投影，而向量的长度为 $\|\mathbf{a}\|$ 。

而这个投影实际上就是本章后续要介绍的**标量投影** (scalar projection)。

简单来说，标量投影是指一个向量在另一个向量方向上的投影长度 (标量、坐标值)。

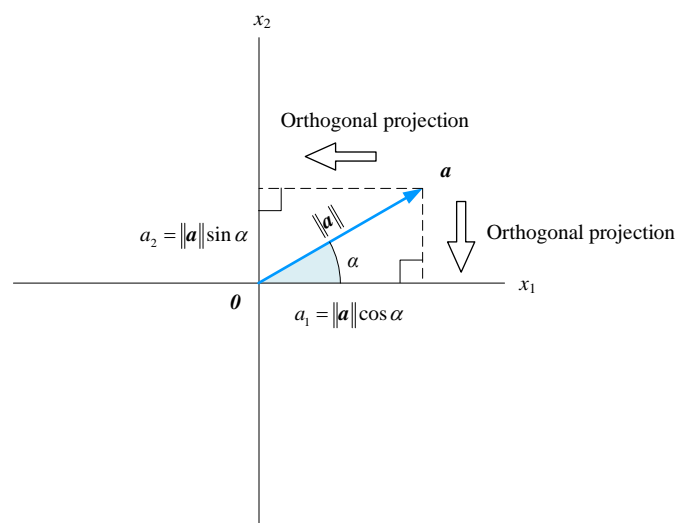


图 5. 向量在水平、竖直方向的标量投影

就好比，在正午时分，头顶的阳光直射下来，斜杆的影子正好垂直投射在水平地面上，影子长度就是斜杆在地面的正交投影。

如图 5 所示，利用勾股定理，我们可以得到

$$\begin{cases} a_1 = \|a\| \cos \alpha \\ a_2 = \|a\| \sin \alpha \end{cases} \quad (6)$$

如图 6 (a) 所示，**正弦** (sine) 表示一个角对应的直角三角形中，对边长度与斜边长度的比值，也可以理解为单位圆上该角对应点的纵坐标值。

如图 6 (b) 所示，**余弦** (cosine) 表示一个角对应的直角三角形中，邻边长度与斜边长度的比值，也可以理解为单位圆上该角对应点的横坐标值。图 6 还展示了其他几个常用三角函数。

α 为向量 a 和水平正方向的夹角，即

$$\alpha = \arctan\left(\frac{a_2}{a_1}\right) \quad (7)$$

其中， \arctan 为**反正切** (arctangent)，是**正切** (tangent, 简作 \tan) 的反函数；简单来说，表示已知正切值求对应角度。

如图 6 (c) 所示，正切 $\tan \alpha$ 表示一个角 α 对应的直角三角形中，对边长度与邻边长度的比值，也可以理解为单位圆上该角对应点的纵坐标与横坐标的比值。

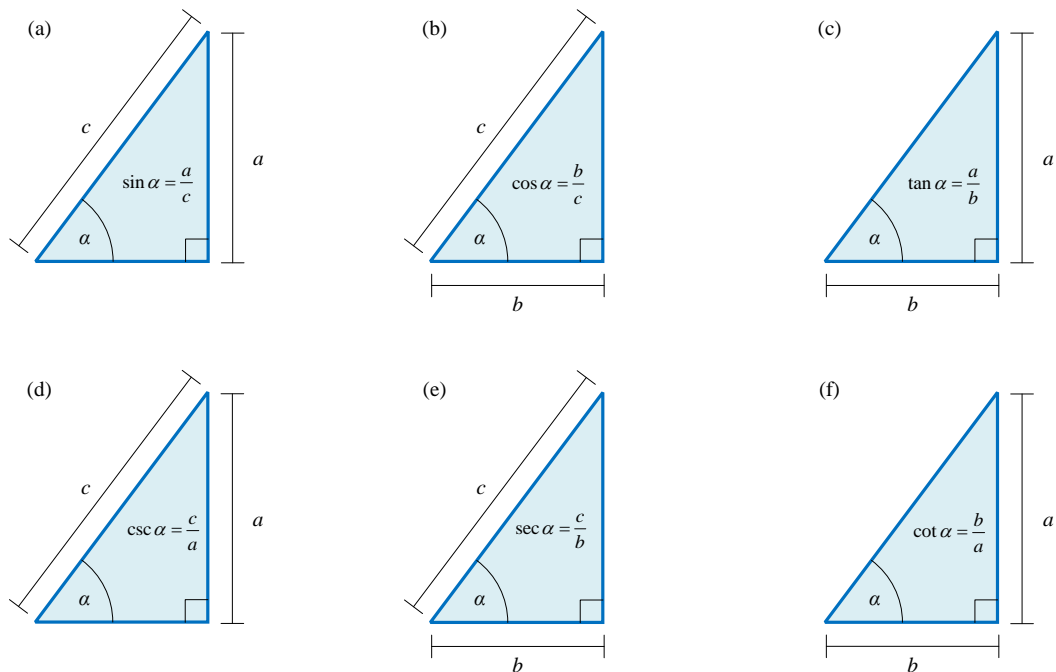


图 6. 常用三角函数

这样，列向量 a 可以写成“长度 \times 方向向量”这种形式

$$\mathbf{a} = \begin{bmatrix} a_1 \\ a_2 \end{bmatrix} = \begin{bmatrix} \|\mathbf{a}\| \cos \alpha \\ \|\mathbf{a}\| \sin \alpha \end{bmatrix} = \|\mathbf{a}\| \begin{bmatrix} \cos \alpha \\ \sin \alpha \end{bmatrix} \quad (8)$$

$\|\mathbf{a}\|$ 为标量，代表向量长度；而 $\begin{bmatrix} \cos \alpha \\ \sin \alpha \end{bmatrix}$ 为方向向量（单位向量），代表 \mathbf{a} 的方向。

极坐标

这实际上给了我们另外一种描述平面上点的位置的思路——**极坐标** (polar coordinate system)。

也就是说，平面上某点的坐标可以长度（极径， r ）、方向（极角， θ ）来表示。

极坐标系的原点称为**极点** (pole)。

极轴 (polar axis) 从极点出发的一条参考射线，通常与笛卡尔坐标系中横轴正半轴重合。

极径 (radial distance) 表示给定某点到极点的距离，一般用 r 表示。对于向量 \mathbf{a} ， r 对应 $\|\mathbf{a}\|$ 。

极角 (polar angle) 表示极径与极轴的夹角，一般用 θ 表示； θ 通常以弧度为单位，规定逆时针角度为正，范围一般为 $[0, 2\pi)$ 。

在极坐标系中，一个点的位置用 (r, θ) 表示。如图 7 所示，直角坐标系坐标和极坐标可以相互转换。

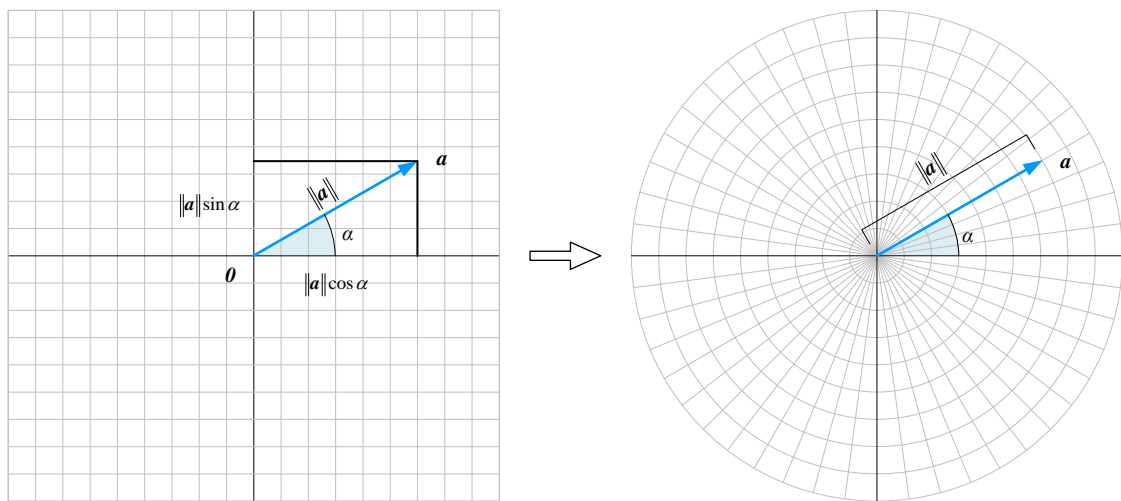


图 7. 平面直角坐标系到极坐标系

代码 2 完成平面直角坐标系坐标和极坐标相互转换。下面聊聊其中关键语句。

- a** 这行代码计算极径；`numpy.sqrt()` 是 n 开平方函数，它的作用是返回输入值的平方根。`x1**2` 表示 x_1 的平方，`x2**2` 表示 x_2 的平方。
- b** 用 `numpy.arctan2()` 计算反正切，结果为弧度。与 `numpy.arctan()` 不同，`numpy.arctan2()` 可以正确处理所有象限的角度，不需要额外考虑 x 为负数的情况。
- c** 用 `numpy.rad2deg()` 将输入的弧度值转换为角度值。

代码 2. 平面直角坐标和极坐标相互转换



LA_01_05_02.ipynb

```

## 初始化
import numpy as np

## 平面直角坐标转化为极坐标
x1, x2 = 3, 4

# 计算极径
a r = np.sqrt(x1**2 + x2**2)

# 计算极角
b theta = np.arctan2(x2, x1)
c np.rad2deg(theta)

## 极坐标转化为平面直角坐标
x1_ = r * np.cos(theta)
x2_ = r * np.sin(theta)

```



绘制正圆

代码 3 展示如何用 Python 在平面直角坐标系中绘制单位圆，如图 9 (a) 所示。

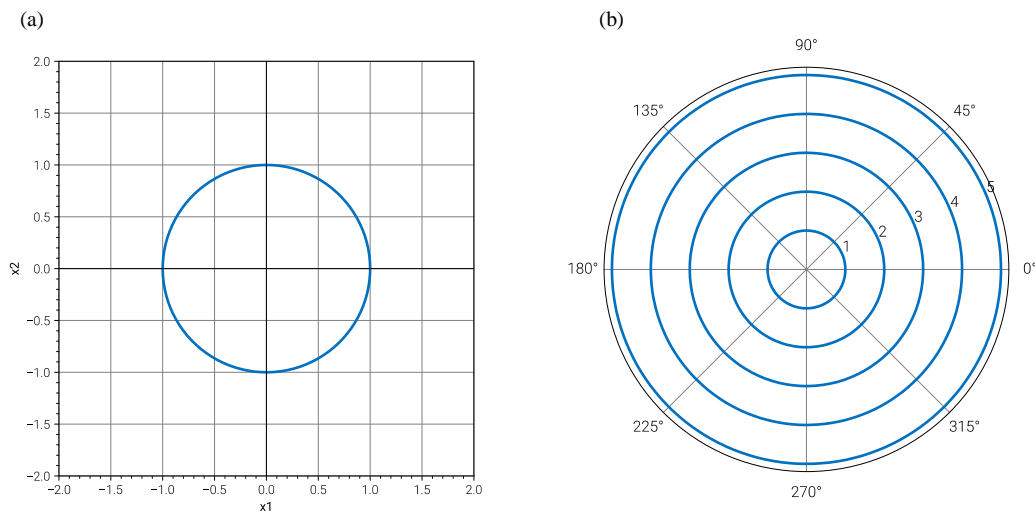


图 8. 可视化正圆

下面聊聊其中关键语句。

a 先生成一个叫做 `theta_array` 的数组。

`numpy.linspace(a, b, n)` 的意思是在从 `a` 到 `b` 的区间里，等间隔地取出 `n` 个数字。所以这里是从 0 到 $2 * \text{numpy.pi}$ 之间，也就是 0 到一个圆周的范围里，平均取出 300 个点。

`numpy.pi` 是 NumPy 提供的圆周率 π ， $2 * \text{np.pi}$ 就是一整圈的角度，用弧度来表示。这个数组就代表了从 0 到一圈的所有角度。

`numpy.cos()` 是 NumPy 提供的余弦函数，把前面生成的 `theta_array` 数组作为输入，组成一个新的数组 `x1_array`，这些值就是单位圆上点的横坐标。

类似地，`numpy.sin()` 是正弦函数，生成数组 `x2_array`，即单位圆上点的横坐标。

b 用 `matplotlib.pyplot.subplots()` (简作 `plt.subplots()`) 创建一个画图的画布 `fig`、轴对象 `ax`。我们可以在轴对象 `ax` 上画各种图。`figsize=(6,6)` 表示画布大小是 6 英寸 × 6 英寸，确保画出来是个正方形区域。

c 在轴对象 `ax` 上，`ax.plot()` 绘制线图。前两个参数分别是横坐标和纵坐标的数据，这里就是我们前面算出来的圆上的 300 个点的横轴、纵轴坐标。`'b'` 是颜色的缩写，表示蓝色 (blue)。`linewidth=1` 表示线的粗细是 1，属于比较细的线。最终它会把这些点依次连起来，看起来就是一个圆的轮廓 (放大看实际上是折线)。

d 是各种图像装饰，让我们逐个了解一下。

`ax.set_aspect('equal')` 设置坐标轴的比例，也就是说让横轴和纵轴的单位长度相同。加上这句后，圆看起来才会是圆，不会被拉扁。

`ax.grid(True)` 让图上显示网格线，像方格纸那样的背景。

`ax.set_xlim([-2, 2])` 这行代码是设置横轴的显示范围，也就是说横轴左边最远到 -2，右边最远到 2。


`ax.set_ylim([-2, 2])` 和上一行类似，不过是设置纵轴的显示范围。也是从 -2 到 2，保证图形是个正方形视图。

`ax.axhline(0, color='k', linewidth=0.5)` 画一条横向的直线，也就是横轴。第一个参数 0 表示在 $y=0$ 的位置画一条线，`color='k'` 表示这条线是黑色的，`linewidth=0.5` 表示线宽是 0.5。

`ax.axvline(0, color='k', linewidth=0.5)` 和上一行类似，是画纵向的直线，也就是纵轴。第一个参数 0 表示在 $x=0$ 的位置画一条黑线，线宽为 0.5。

`ax.set_xlabel('x1')` 给横轴加一个标签，写上 'x1'。

`ax.set_ylabel('x2')` 是给纵轴加一个标签，写上 'x2'。

代码 3. 在平面直角坐标系中绘制单位圆 |  LA_01_05_03.ipynb

```

## 初始化
import numpy as np
import matplotlib.pyplot as plt

## 生成数据
a theta_array = np.linspace(0, 2 * np.pi, 300) # 0 到 2π 之间均匀取点
  x1_array = np.cos(theta_array)
  x2_array = np.sin(theta_array)

## 可视化
b fig, ax = plt.subplots(figsize=(6,6))

# 绘制单位圆
c ax.plot(x1_array, x2_array, 'k', linewidth=1) # 黑色圆，线宽 1

# 设置 XY 轴比例相同
d ax.set_aspect('equal')
  ax.grid(True)

# 设置坐标轴范围
  ax.set_xlim([-2, 2])
  ax.set_ylim([-2, 2])

# 添加坐标轴
  ax.axhline(0, color='k', linewidth=0.5)
  ax.axvline(0, color='k', linewidth=0.5)

# 增加轴标签
  ax.set_xlabel('x1')
  ax.set_ylabel('x2')

```



LA_01_05_04.ipynb 展示如何用极坐标系表示多个同心圆，请大家自学。

3 维单位向量

对于任意非零 3 维向量 $\mathbf{a} = [a_1, a_2, a_3]^T$ ，其**方向向量**为

$$\hat{\mathbf{a}} = \frac{\mathbf{a}}{\|\mathbf{a}\|} = \begin{bmatrix} \frac{a_1}{\sqrt{a_1^2 + a_2^2 + a_3^2}} \\ \frac{a_2}{\sqrt{a_1^2 + a_2^2 + a_3^2}} \\ \frac{a_3}{\sqrt{a_1^2 + a_2^2 + a_3^2}} \end{bmatrix} \quad (9)$$

举个例子，给定 3 维列向量 $\mathbf{b} = [2, 3, 6]^T$ ，它的**方向向量**为

$$\hat{\mathbf{b}} = \frac{\mathbf{b}}{\|\mathbf{b}\|} = \frac{1}{\sqrt{2^2 + 3^2 + 6^2}} \times \begin{bmatrix} 2 \\ 3 \\ 6 \end{bmatrix} = \begin{bmatrix} 2/7 \\ 3/7 \\ 6/7 \end{bmatrix} \quad (10)$$

3 维列向量 $\mathbf{b} = [2, 3, 6]^T$ 也可以写成“长度 \times 方向向量”的形式，即

$$\mathbf{b} = 7 \times \begin{bmatrix} 2/7 \\ 3/7 \\ 6/7 \end{bmatrix} = \begin{bmatrix} 2 \\ 3 \\ 6 \end{bmatrix} \quad (11)$$

3 维方向向量 (单位向量) 的若起点位于原点，终点位于**单位球** (unit sphere) 表面。

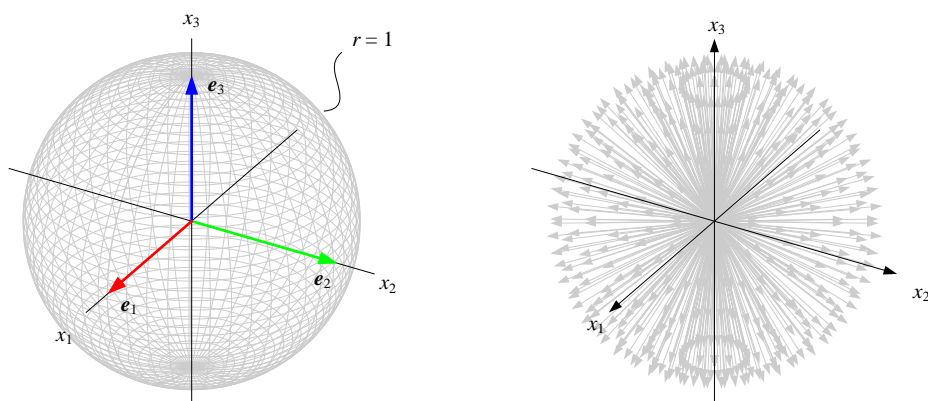


图 9. 三维空间中的单位向量终点位于单位球表面

图 10 所示为，RGB 空间中长度为 1 的单位向量，其中包括我们熟悉的基向量纯红色向量 \mathbf{e}_1 、纯绿色向量 \mathbf{e}_2 、纯蓝色向量 \mathbf{e}_3 。

所有这些向量的起点都在原点，它们的向量终点分布在一个半径为 1 的单位球的 1/8 球面上。

然而，白色对应的向量 $[1, 1, 1]^T$ 的长度为 $\sqrt{3}$ ，超出了单位球的范围。因此，白色向量并不位于该单位球面上，而是在其外部。

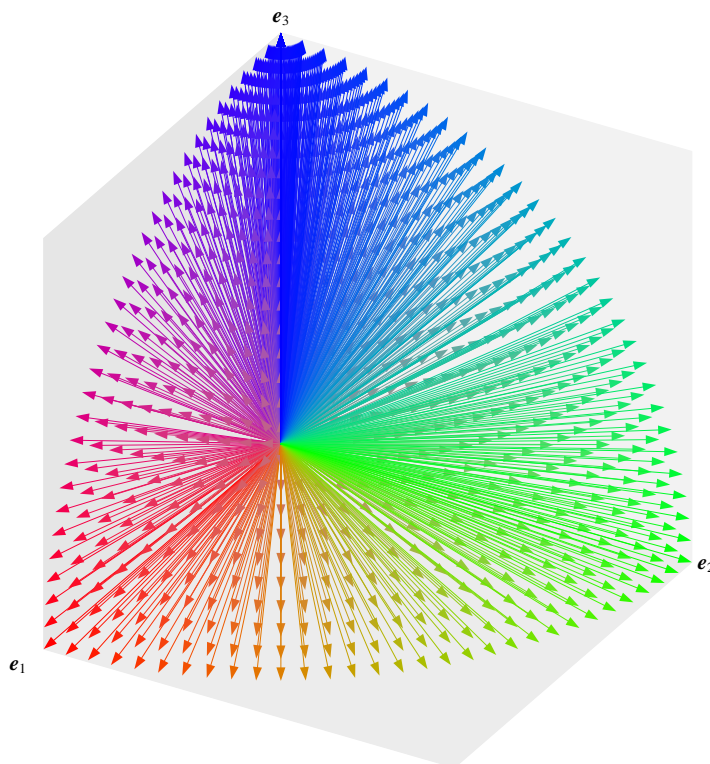


图 10. RGB 颜色空间中的单位向量

球坐标

三维直角坐标系中的单位向量和**球坐标** (spherical coordinate system) 系有着密切联系。

如图 11 所示，球坐标相当于由两个平面极坐标系构造。

球坐标系中定位点 P 用的是球坐标 (r, θ, φ) 。

其中， r 是 P 与原点 O 之间距离，也叫**径向距离** (radial distance)；

θ 是 OP 连线和 x_3 轴正方向夹角，叫做**极角** (polar angle)；球坐标 θ 取值范围为 $[0, \pi]$ 。这个角度相当于地球的“纬度角”。

OP 连线在 x_1x_2 平面投影线为 OH ， φ 是 OH 和 x_1 轴正方向夹角，叫做**方位角** (azimuth angle)。球坐标 φ 取值范围为 $[0, 2\pi]$ 。这个角度相当于地球的“经度角”。

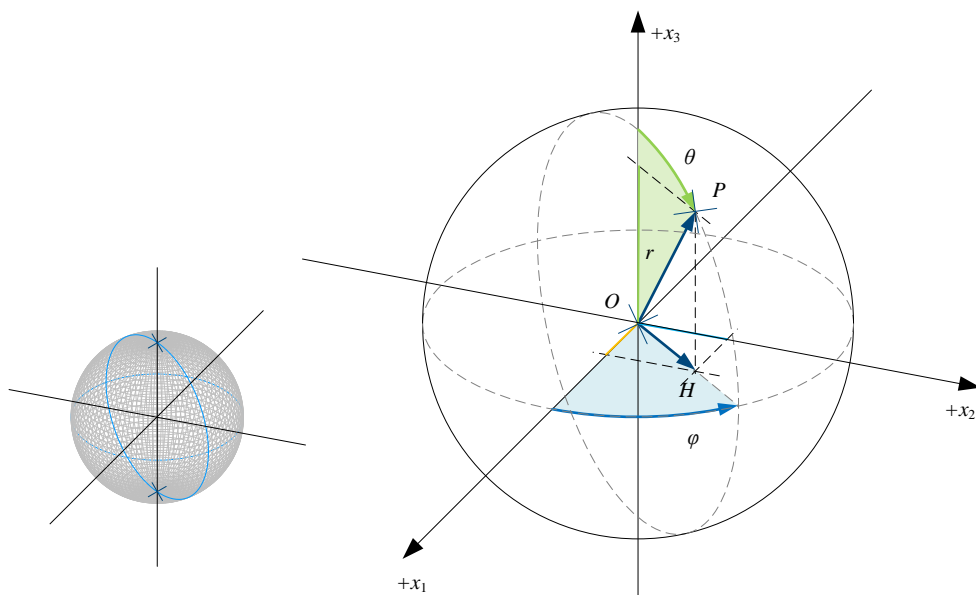


图 11. 球坐标系, 图片来自《可视之美》

下面, 让我们看看球坐标 (r, θ, φ) 和三维直角坐标 (x_1, x_2, x_3) 之间的转换。

如图 12 (b) 所示, 由于 x_3 垂直于 x_1x_2 平面, r 在 x_3 上的投影就是 P 点在 x_3 轴的坐标值, 即

$$x_3 = r \cos \theta \quad (12)$$

如图 12 (c) 所示, PO 在 x_1x_2 平面上的投影为

$$r \sin \theta \quad (13)$$

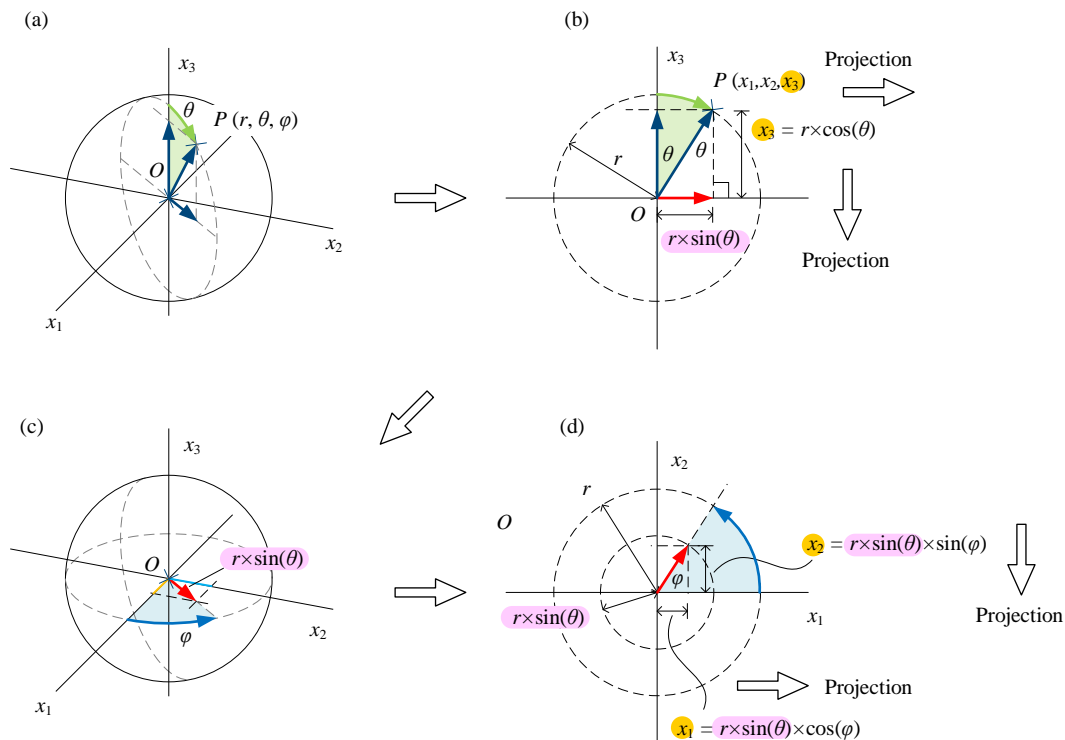


图 12. 从球坐标到三维直角坐标, 图片来自《可视之美》

然后，如图 12 (d) 所示， $r\sin\theta$ 向 x_1 轴投影便是 P 点在 x_1 轴的坐标值

$$x_1 = r \sin \theta \cos \varphi \quad (14)$$

$r\sin\theta$ 向 x_2 轴投影便是 P 点在 x_2 轴的坐标值

$$x_2 = r \sin \theta \sin \varphi \quad (15)$$

综合以上，我们可以得到 P 在三维直角坐标系中的坐标为

$$\begin{cases} x_1 = r \sin \theta \cos \varphi \\ x_2 = r \sin \theta \sin \varphi \\ x_3 = r \cos \theta \end{cases} \quad (16)$$

这样， P 点对应的向量为

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} r \sin \theta \cos \varphi \\ r \sin \theta \sin \varphi \\ r \cos \theta \end{bmatrix} = r \begin{bmatrix} \sin \theta \cos \varphi \\ \sin \theta \sin \varphi \\ \cos \theta \end{bmatrix} \quad (17)$$

对应的方向向量为

$$\begin{bmatrix} \sin \theta \cos \varphi \\ \sin \theta \sin \varphi \\ \cos \theta \end{bmatrix} \quad (18)$$

反过来看，利用勾股定理， r 为

$$r = \sqrt{x_1^2 + x_2^2 + x_3^2} \quad (19)$$

这便是向量的长度。



LA_01_05_05.ipynb 完成三维直角坐标和球坐标的相互转换，请大家自行学习。

可视化单位球

代码 4 绘制图 13 (a) 单位球网格面。

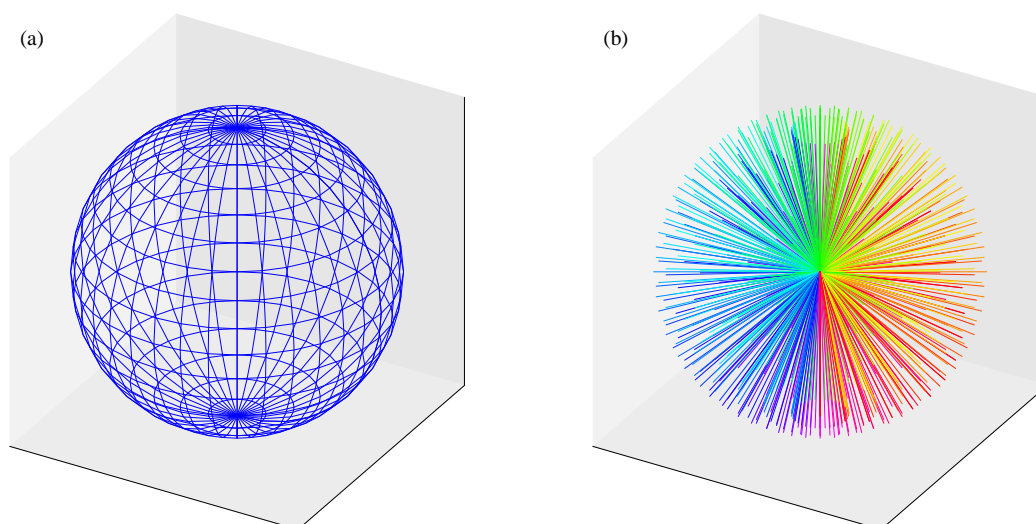


图 13. 可视化单位球

下面聊聊其中关键语句。

a 先用 `numpy.linspace()` 生成从 0 到 2π 均匀取 37 个点。这个数组代表“经度角”，也就是绕着地球转一圈的那个角度。

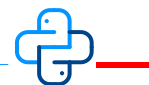
然后用 `numpy.linspace()` 生成从 0 到 π 取 19 个点。这个是“纬度角”，代表从北极到南极的半圈。

用 `numpy.meshgrid()` 函数生成一个二维网格。`u` 和 `v` 是两个一维数组，这个函数把它们扩展成二维数组 `uu` 和 `vv`，这样我们就能在一个二维角度网格上做计算了，相当于为球面每个点配备了一个经度角、纬度角。


b 把球面的每个球坐标转化成 x_1 轴、 x_2 轴、 x_3 轴坐标。

c 创建图对象。在图上添加一个三维子图。111 的意思是“1 行 1 列中的第 1 个子图”，`projection='3d'` 表示这个图是三维的。

d 用 `plot_wireframe()` 是绘制网格用的函数，传入的 `xx, yy, zz` 是每个网格点的位置。`color='blue'` 表示线的颜色是蓝色，`linewidth=0.5` 表示线的宽度是 0.5。



LA_01_05_06.ipynb 还绘制图 13 (b)，请大家自行学习。

代码 4. 在平面直角坐标系中绘制单位圆 |  LA_01_05_06.ipynb

```

## 初始化
import numpy as np
import matplotlib.pyplot as plt
from matplotlib.cm import hsv

## 创建球面数据
a u = np.linspace(0, 2 * np.pi, 37) # 生成经度角
  v = np.linspace(0, np.pi, 19)      # 生成纬度角
  uu, vv = np.meshgrid(u, v)        # 生成网格
  r = 1 # 球半径

# 计算球面坐标
b xx = r*np.cos(uu) * np.sin(vv)
  yy = r*np.sin(uu) * np.sin(vv)
  zz = r*np.cos(vv)

## 可视化
c fig = plt.figure(figsize=(6, 6))
  ax = fig.add_subplot(111, projection='3d')

# 绘制球面网格线
d ax.plot_wireframe(xx, yy, zz, color='blue', linewidth=0.5)

# 设置坐标轴范围
ax.set_xlim([-1, 1])
ax.set_ylim([-1, 1])
ax.set_zlim([-1, 1])

# 隐藏坐标轴刻度
ax.set_xticks([])
ax.set_yticks([])
ax.set_zticks([])
ax.set_aspect('equal', 'box')
ax.set_proj_type('ortho')

```



请大家用 DeepSeek/ChatGPT 等工具完成本节如下习题。

Q1. 逐行注释下例，并学习如何绘制等高线图。

https://matplotlib.org/stable/gallery/images_contours_and_fields/contour_demo.html

Q2. 学习绘制填充等高线。

https://matplotlib.org/stable/api/as_gen/matplotlib.pyplot.contourf.html

Q3. 请编写 Python 代码绘制正弦、余弦图像。

Q4. 请将 (4, 4) 平面直角坐标转换为极坐标。

Q5. 计算 $[4, 4]^T$ 的单位向量；并把 $[4, 4]^T$ 写成“长度 \times 方向向量”的形式。

Q6. 计算 $[1, 2, 2]^T$ 的单位向量；并把 $[1, 2, 2]^T$ 写成“长度 \times 方向向量”的形式。

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

Q7. 请将极坐标 $(2, \pi/3)$ 转化为平面直角坐标。

Q8. 请将球坐标 $(4, \pi/6, \pi/4)$ 转化为三维直角坐标系坐标。