1.2 坐标系

为了可视化向量，我们需要借助坐标系。通过引入平面和三维直角坐标系，我们能够将抽象的向量具体化，使其在几何空间中具有明确的位置和方向。本节从最基本的实数数轴入手，理解其作为一维欧几里得空间的本质。接着，引入了平面直角坐标系，阐述了笛卡尔坐标系的起源及其在几何与代数之间的桥梁作用。通过向量的表示，我们理解了如何利用坐标来描述空间中的点及其位置关系。此外，本节还探讨了 numpy.meshgrid() 的应用，它能够生成规则的坐标网格数据，为数值计算和绘图提供基础。随后，我们进一步拓展至三维直角坐标系，并强调了右手定则对空间方向的规范。结合 RGB 颜色模式，我们直观展示了三维坐标系中的向量关系，同时引出了向量分解的概念。最后，我们讨论了向量的长度计算，介绍了欧几里得范数及其计算方法，并提供了相关代码示例。

实数数轴

在介绍直角坐标系之前，让我们先了解**实数数轴**这个概念。

如图 1 所示，**实数数轴** (real line, real axis) 是一条无限延伸的直线，它为我们提供了表示实数 (标量) 位置的基本工具。

原点 (0, origin)、正方向 (箭头指向)、单位长度 (1) 是数轴重要的三要素。原点左侧为负数，原点右侧为正数。

实数数轴向左右无限延伸，分别趋向于**负无穷** (negative infinity, $-\infty$)、**正无穷** (positive infinity, $+\infty$)。

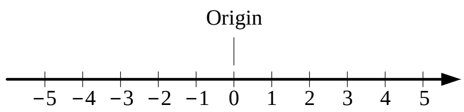

图 1. 实数数轴

实数数轴也叫**一维欧几里得空间** (one-dimensional Euclidean space)，记作 $\mathbb{R}$，也就是实数集合。

平面直角坐标系

笛卡尔坐标系 (Cartesian Coordinates)，也叫**平面直角坐标系** (rectangular coordinate system)，是一种通过一组数值来唯一确定点在空间中位置的系统。

笛卡尔 (René Descartes) 在 17 世纪提出了坐标系的概念，从而开创性地将代数与几何紧密结合在一起。他认为，几何图形不仅可以用直观的形状来描述，还可以用数字和符号表达。

传说笛卡尔在床上思考数学问题时，观察到天花板上的苍蝇，他意识到可以用两个垂直的数轴来描述苍蝇的位置。这一灵感促使他发明了笛卡尔坐标系，将几何问题转换为代数方程来求解。

正是这种方法，使得几何问题能够转化为代数问题，通过求解方程来寻找曲线的交点、计算距离、研究图形的性质等。笛卡尔的坐标系不仅为传统几何学注入了代数的严密性，也为后来的解析几何奠定了基础，使得复杂的几何形体可以用简单的代数运算来处理，从而极大地推动了数学的发展。

如图 2 (a) 所示，当两个数轴在原点垂直相交时，便构成了**平面直角坐标系**，也叫做**二维欧几里得空间** (two-dimensional Euclidean space)，记作 $\mathbb{R}^2$ 。

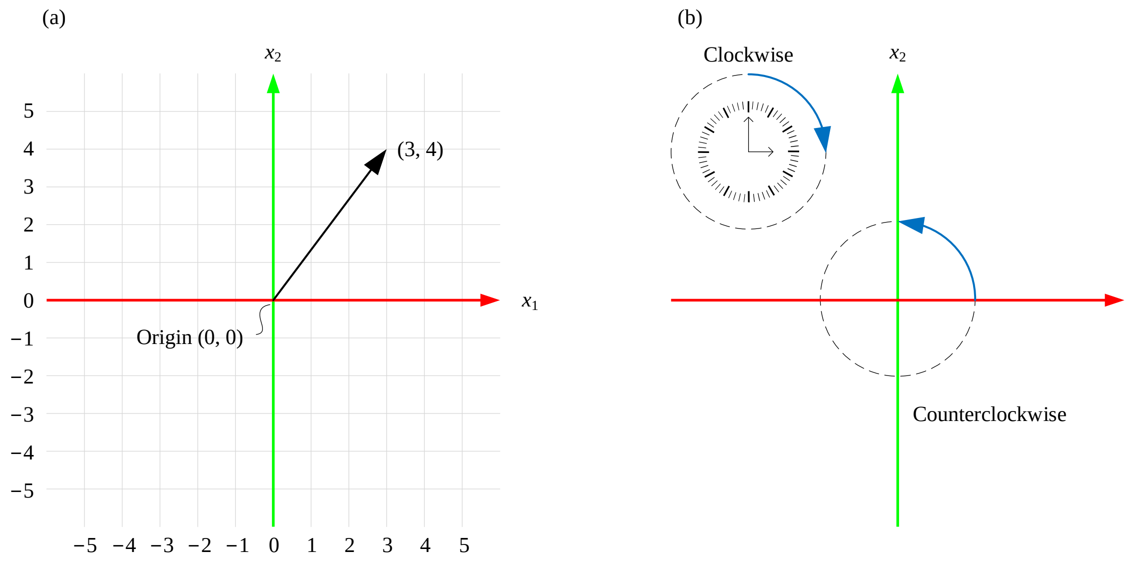

图 2. 平面直角坐标系

在这个坐标系中，每一个点坐标由元组 (x, y) 或 (x_1, x_2) 表示，本书常用 (x_1, x_2)；其中 x_1 表示水平方向的位置，x_2 表示竖直方向的位置。

一个列向量 $\boldsymbol{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$ 可以看作是平面上，起点位于 (0, 0) 终点位于 (x_1, x_2) 的向量；显而易见，零向量 $\boldsymbol{0}$ 位于坐标系的原点。

一般情况下，没有特殊说明，我们默认向量的起点位于原点。这样，我们可以通过坐标值唯一地表示向量，并利用代数方法进行计算和推导。

当然，在某些应用场景中，我们也可能将向量平移到不同的位置，例如在下一节关于向量加法的讨论中，我们会将一个向量的起点与另一个向量的终点对齐，以构造“首尾相接”的几何图像，从而直观地理解向量相加的过程。

大家可能已经注意到，图 2 中，x_1 轴的颜色被设置为红色，而 x_2 轴的颜色为绿色。这样设计的目的是我们将用“红绿”平面上的散点来代表红光和绿光混合得到的不同颜色来讲解 2 维向量。

此外，在平面直角坐标系中，图 2 (b) 所示，从 x_1 轴的正方向旋转 90 度到 x_2 轴的正方向，这个方向是**逆时针** (counterclockwise)，角度定义为正；换个角度来看，(在不超过 180 度范围内) 逆时针来看，x_1 轴的正方向领先于 x_2 轴的正方向。

这意味着我们定义了一个右手系的标准方向，使得向量的方向性和旋转性都符合“逆时针为正”的约定。

这一顺序至关重要，因为它影响到许多几何和代数计算，例如行列式、叉积和角度方向的判断，确保空间关系的一致性和正确性。

创建二维网格数据

函数 numpy.meshgrid() 来自 NumPy，用于生成坐标网格数据；而坐标网格数据常用于数值计算、绘图和向量化运算。

函数 numpy.meshgrid() 接受一维数组作为输入，生成两个或多个坐标矩阵，使得在多维空间中，每个点的坐标可以通过这些矩阵索引获得。

如图 3 所示，在二维平面上，numpy.meshgrid() 可以创建一个 x_1 轴和一个 x_2 轴的坐标网格；这个函数输出的是网格点处 x_1、x_2 的坐标值 (都是二维数组)。

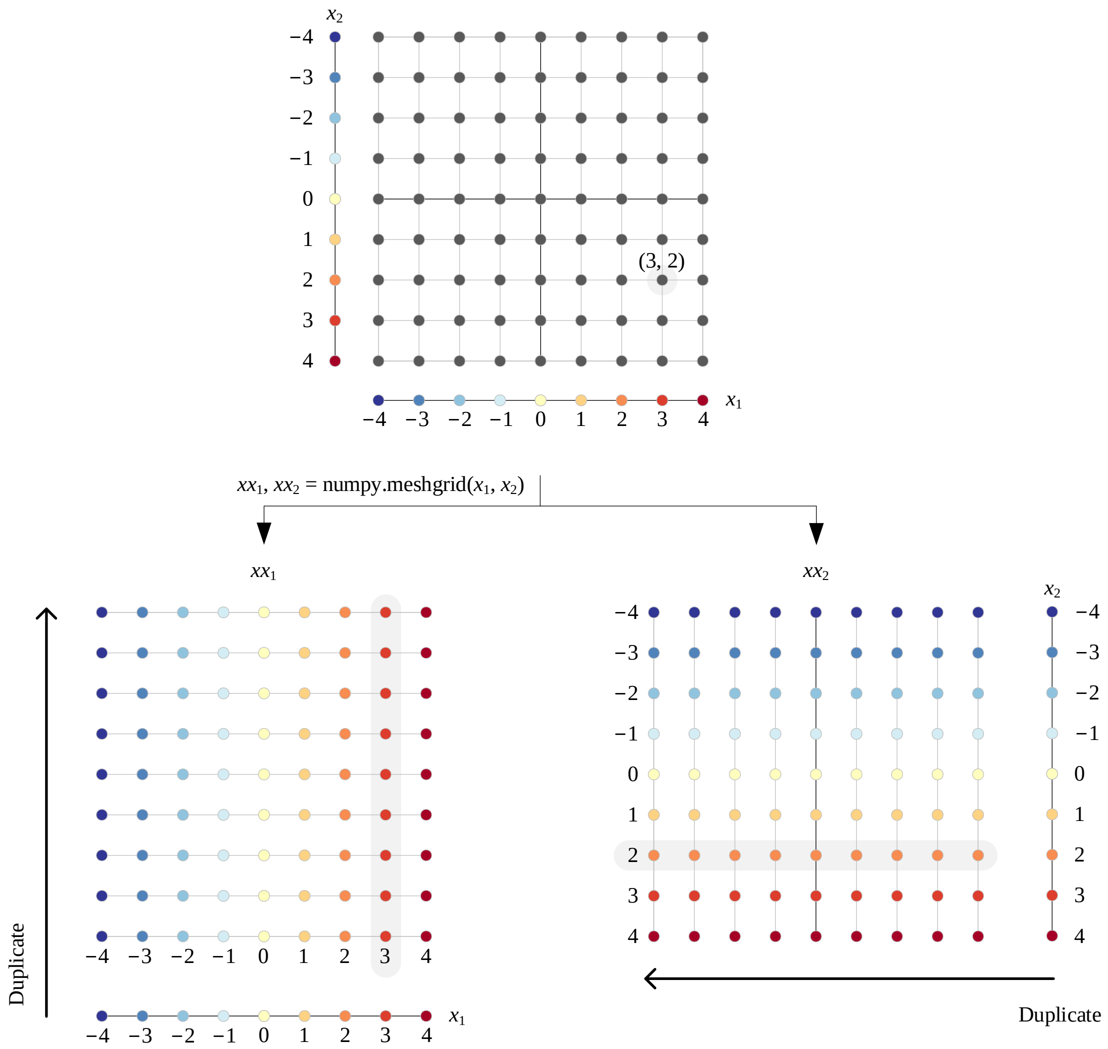

图 3. 用 numpy.meshgrid() 创建二维网格数据

代码 1 生成图 3 中二维网格数据。

代码 1. 用 numpy.meshgrid() 创建二维网格数据 | LA_01_02_01.ipynb

```
import numpy as np

# 生成 x1 和 x2 的一维数组，范围 [-4, 4]，步长为 1
x1 = np.arange(-4, 5, 1)
x2 = np.arange(-4, 5, 1)

# 生成网格
xx1, xx2 = np.meshgrid(x1, x2)
```

下面逐句解释代码 1。

ⓐ 导入 NumPy 库，并为其指定别名 np，这样在后续代码中就可以用 np 来调用 NumPy 提供的函数，而不需要每次都输入完整的 numpy，从而提高代码的简洁性和可读性。

ⓑ 用 numpy.arange() 函数创建一维数组。numpy.arange(start, stop, step) 函数会生成一个从 start 到 stop 之间的等间距 (step) 数组，但 stop 本身不会包含在结果中。

ⓒ 用 numpy.meshgrid() 函数，该函数的作用是接收两个一维数组 x1、x2，并生成两个二维数组 xx1、xx2，用于存储网格中每个点的坐标值。xx1 代表网格中所有点的 x1 坐标，xx2 代表网格中所有点的 x2 坐标。

如图 3 所示，从几何上来看，xx1 相当于将一维数组 x1 在每一行上复制多次；而 xx2 相当于将一维数组 x2 在每一列上复制多次，从而形成一个规则的二维网格。

需要注意的是，在纵向 numpy.meshgrid()生成的 xx2 数组中，负数位于上方，正数位于下方，这与常见的直角坐标系图示相反。通常在数学图像中，纵轴的正方向向上，而 numpy.meshgrid() 生成的 xx2 采用数组索引方式，导致数值顺序自上而下递增。

把 2 维向量放到平面直角坐标系中

当向量数量较少时，或者我们关注向量的起点、终点的具体位置时，我们常用**箭头** (arrow, quiver) 来代表向量。

而向量数量众多，或者我们主要关注向量的分布和趋势时，则倾向于使用**散点图** (scatter plot) 来简化表示；此时向量的起点被默认为坐标原点，而终点则由散点的位置来指示。

如图 4 (a) 所示，红色向量为 $\begin{bmatrix} 1 \\ 0 \end{bmatrix}$，表示纯红色 (光)；绿色向量为 $\begin{bmatrix} 0 \\ 1 \end{bmatrix}$，表示纯绿色 (光)。

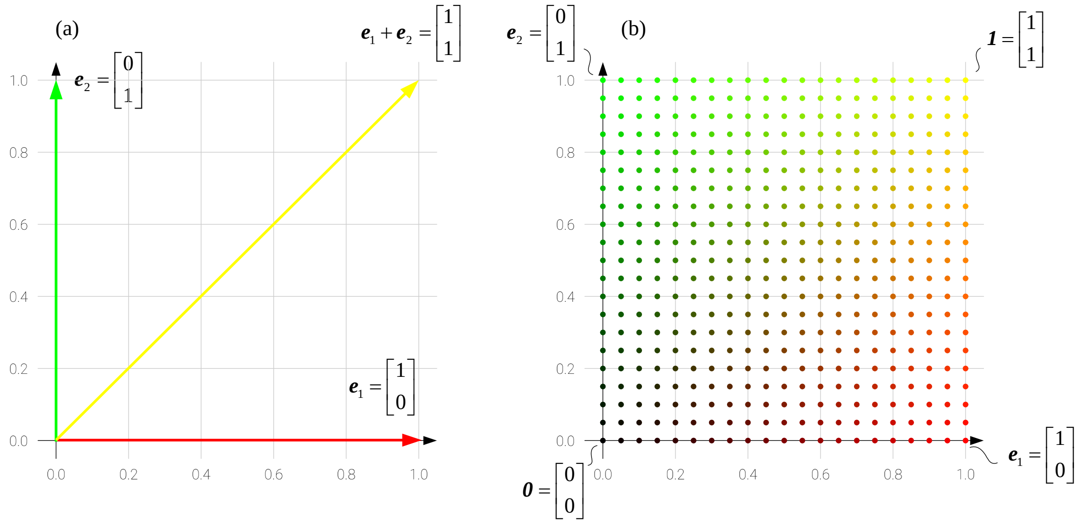

图 4. “红绿”平面上的单位向量和规则散点

如图 4 (a) 所示，在红绿平面 (x_1x_2 平面) 上，原点 $\begin{bmatrix} 0 \\ 0 \end{bmatrix}$ (零向量 $\boldsymbol{0}$) 代表纯黑色。

上一节提过，向量所有分量均为 0 的向量叫做**零向量** (zero vector)，记作 $\boldsymbol{0}$ (粗体，斜体)。**零向量**的长度为 0。与之相反，并非所有分量都为零的向量统称为**非零向量** (nonzero vector)。**非零向量**的长度大于 0。

黄光的向量则为 $\begin{bmatrix} 1 \\ 1 \end{bmatrix}$，表示纯红光和纯绿光的混合。这里，相信大家已经看到了**向量加法** (vector addition)，即 $\begin{bmatrix} 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix}$。这是本章后续要介绍的知识点。

黄光是**全 1 列向量**。上一节提过，**全 1 列向量**是一个所有分量均为 1 的列向量。

值得注意的是，在图 3 (a) 中，**全 1 列向量**恰为**单位正方形** (unit square) 的对角线。单位正方形是指边长为 1 的正方形。

⚠ RGB 颜色模式，各个颜色分量的取值范围为 [0, 1]。红绿平面仅仅是 $\mathbb{R}^2$ 极小的一部分。哪怕在 RGB 颜色空间中，红绿平面也仅仅是 RGB 正方体的一个立面而已。

向量分解

红绿平面 (x_1x_2 平面) 上，$\begin{bmatrix} 1 \\ 0 \end{bmatrix}$ 和 $\begin{bmatrix} 0 \\ 1 \end{bmatrix}$ 是两个很特殊的向量，我们给它俩特殊的记号 $\boldsymbol{e}_1$、$\boldsymbol{e}_2$

$$\boldsymbol{e}_1 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \boldsymbol{e}_2 = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \tag{1}$$

“红绿”平面上，单位向量 $\boldsymbol{e}_1$ 代表纯红色向量，指向 x_1 轴正方向；单位向量 $\boldsymbol{e}_2$ 代表纯绿色向量，指向 x_2 轴正方向。

很明显，$\boldsymbol{e}_1$ 和 $\boldsymbol{e}_2$ 的长度为 1；我们管长度为 1 的向量叫做**单位向量** (unit vector)。

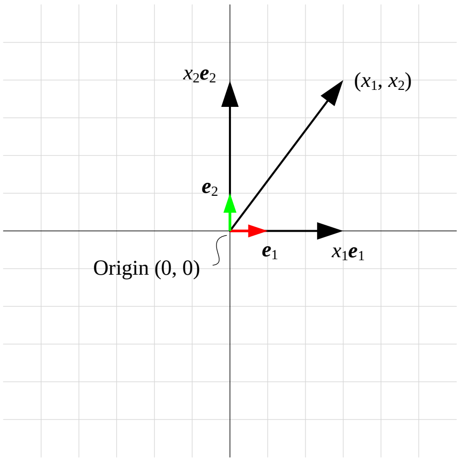

图 5. 把二维向量写成两个分量之和

如图 5 所示，这个平面上任意向量 $\boldsymbol{x} = [x_1, x_2]^{\mathrm{T}}$ 可以写成

$$\boldsymbol{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = x_1 \begin{bmatrix} 1 \\ 0 \end{bmatrix} + x_2 \begin{bmatrix} 0 \\ 1 \end{bmatrix} = x_1\boldsymbol{e}_1 + x_2\boldsymbol{e}_2 \tag{2}$$

这个式子本质上是向量分解；此外这个式子还用到了向量加法、标量乘法。这些都是本章后文要介绍的知识点。

看着 $\boldsymbol{e}_1$、$\boldsymbol{e}_2$ 交织出的方方正正的网格，仿佛整个平面都在它们的支撑下展开。这就是**张成** (span) 的力量，简单却无处不在，像无形的骨架托起了空间的每一寸。请大家记住“**张成**”这个概念，它就像空间的基石，后续会为我们打开理解线性世界的大门。

⚠ RGB 颜色模式中，对于任何颜色的 R、G、B 每个分量都要有具体值；比如，纯红色中蓝色分量为 0；纯绿色中蓝色分量也为 0。写成二维向量的形式仅仅是为了方便理解。几何角度来看，三维到二维相当于投影。

三维直角坐标系

在平面直角坐标系 $\mathbb{R}^2$ 的基础上，我们可以进一步扩展为三维直角坐标系。

具体来说，如果在 x_1x_2 平面的原点处垂直升起一条新的数轴 x_3，并且使 x_3 轴与 x_1 轴、x_2 轴两两垂直，同时原点对齐，我们便得到了三维直角坐标系，记作 $\mathbb{R}^3$。

而在三维空间中，点的位置则由有序三元组 (x, y, z) 或 (x_1, x_2, x_3) 表示，本书则常用 (x_1, x_2, x_3)。

一个向量 $\boldsymbol{x} = [x_1, x_2, x_3]^{\mathrm{T}}$ 可以看做是三维空间中起点位于 (0, 0, 0)、终点位于 (x_1, x_2, x_3) 的 3 维向量。

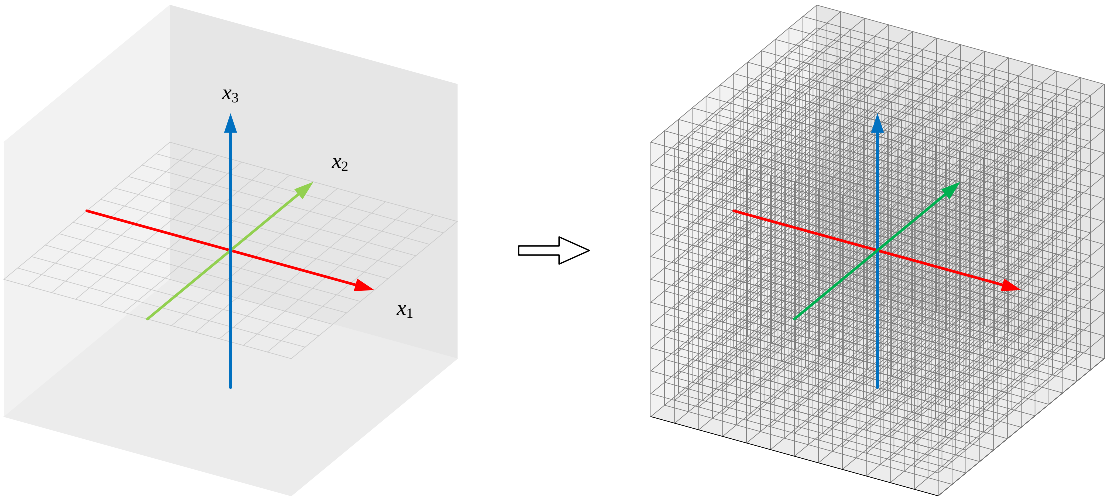

图 6. 三维直角坐标系的网格

本书采用的三维直角坐标系满足右手定则，即按照右手系的标准定义空间方向。

具体来说，如图 7 所示，握紧右手拳头，伸出四指，并使其指向 x_1 轴的正方向；此时，手心自然朝向 x_2 轴的正方向；拇指竖起，指向 x_3 轴的正方向。

和平面直角坐标系的右手定则一样，这种坐标系的定义确保了行列式、坐标变换、向量叉积、旋转方向等数学运算的一致性。

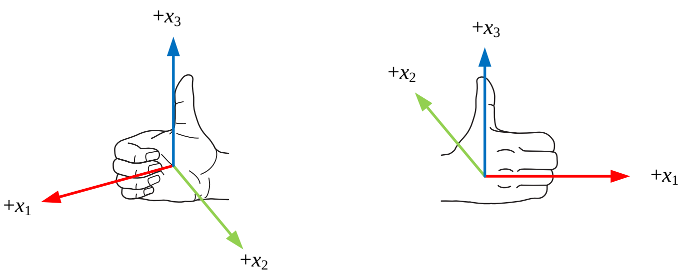

图 7. 右手系三维直角坐标系

图 8 展示如何用 numpy.meshgrid() 生成三维网格数据的原理；蓝色箭头代表一维数据的复制方向。

代码文件 LA_01_02_02.ipynb 利用 numpy.meshgrid() 生成三维数组，请大家自行学习。

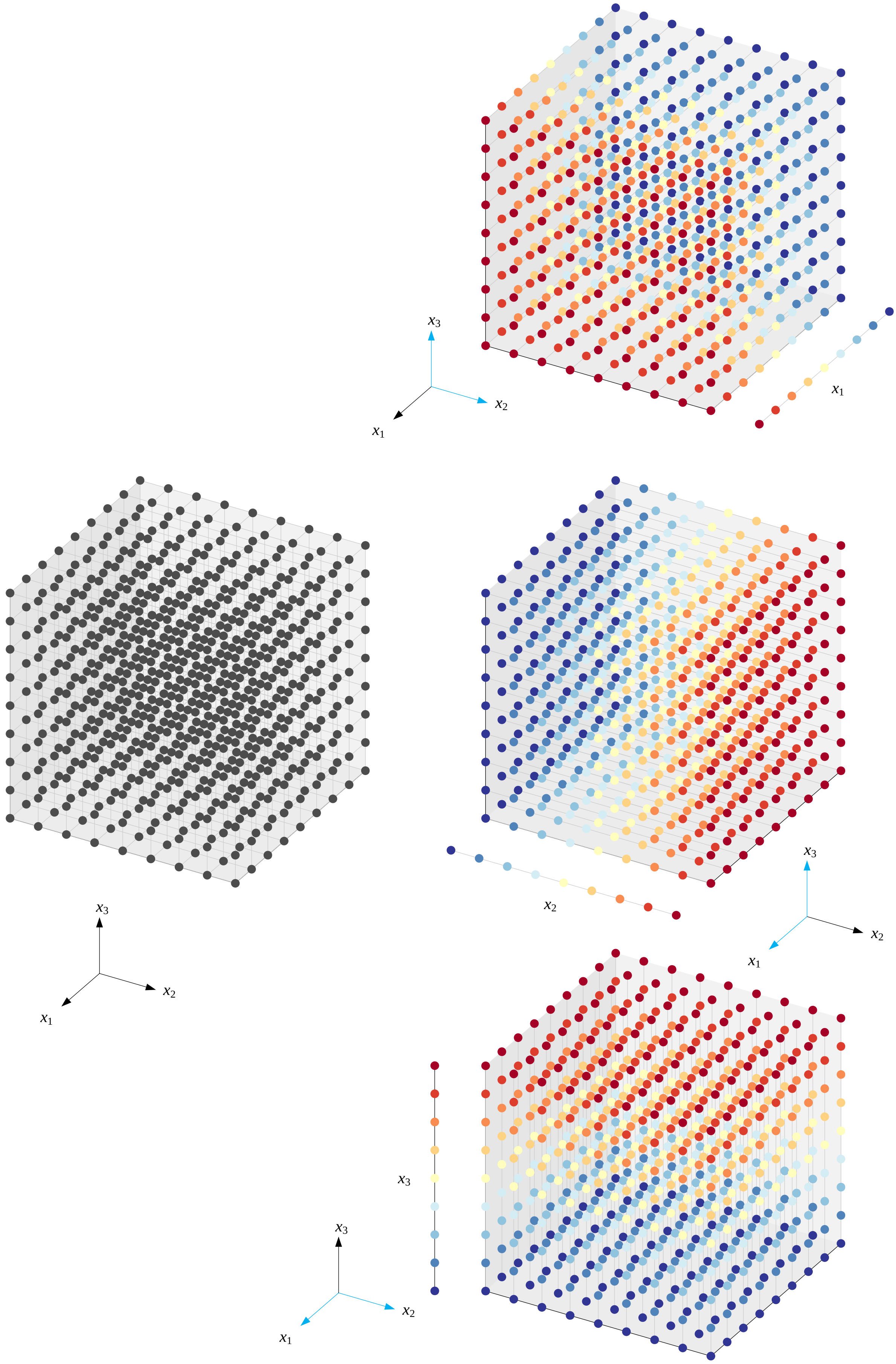

图 8. 用 numpy.meshgrid() 创建三维网格数据

欧几里得空间

欧几里得空间 (Euclidean space) 是一个定义了**欧几里得距离** (Euclidean distance)，即**直线距离** (straight-line distance)，的**向量空间** (vector space)，它满足欧几里得几何的公理和性质。

简单来说，**向量空间**是一个集合，其元素称为向量，并在该集合上定义了向量加法和标量乘法，这两种运算满足封闭性、交换律、结合律、分配律、存在零向量以及每个向量存在加法逆元等基本公理。本书后续将专门讲解向量空间这个概念。

实数数轴 $\mathbb{R}$、**平面直角坐标系** $\mathbb{R}^2$、**三维直角坐标系** $\mathbb{R}^3$ 都是欧几里得空间的特殊例子，分别对应一维、二维、三维欧几里得空间。

在这些空间中，我们可以直观地定义点、线、平面以及它们之间的距离和角度，从而为几何和物理问题的研究提供了基础框架。

向量空间这个概念非常重要。这里大家先对向量空间有个初步印象，本书后续还会一而再、再而三地讲解向量空间。

RGB 颜色模式

图 4 展示的红绿平面是由红色、绿色分量织而成的色彩，它们仅仅是整个 RGB 色彩宇宙中的一小片领域。

然而，这个红绿平面仅仅是探索色彩无限可能性的起点，真正的色彩之旅，还需加上蓝色的维度，才能绘制出更加丰富多彩的世界，如图 9。

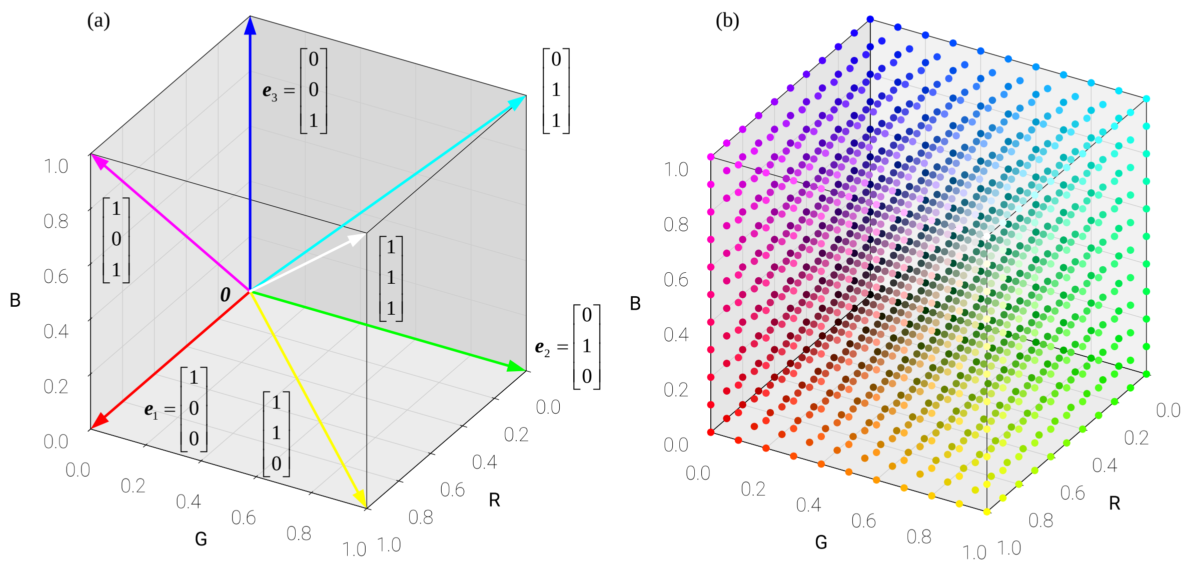

图 9. RGB 颜色空间中的箭头、散点

类似平面直角坐标系 $\boldsymbol{e}_1$、$\boldsymbol{e}_2$，三维直角坐标系也有自己的特殊单位向量

$$\boldsymbol{e}_1 = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \boldsymbol{e}_2 = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}, \boldsymbol{e}_3 = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \tag{3}$$

其中，单位向量 $\boldsymbol{e}_1$ 指向 x_1 轴正方向，单位向量 $\boldsymbol{e}_2$ 指向 x_2 轴正方向，单位向量 $\boldsymbol{e}_3$ 指向 x_3 轴正方向。

在图 9 (a) RGB 颜色空间中，$\boldsymbol{e}_1 = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$ 代表纯红色，$\boldsymbol{e}_2 = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$ 代表纯绿色，$\boldsymbol{e}_3 = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$ 代表纯蓝色。

通过足量混合，红色与绿色交融诞生了鲜艳的**黄色** (yellow)，$\begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix}$；红色与蓝色交织形成了深邃的**品红** (magenta)，$\begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix}$；绿色与蓝色相融则呈现出清新的**青色** (cyan)，$\begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix}$。

在这个空间中**零向量** $\boldsymbol{0} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$ 为黑色，而**全 1 列向量** $\boldsymbol{1} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$ 代表白色。

在图 9 (a) 中，**全 1 列向量**恰为**单位正方体** (unit cube) 的对角线。单位正方体是边长为 1 的正方体。

如图 9 (b) 所示，就像红、绿、蓝三基色可以组合成各种颜色，三维直角坐标系中的基向量 $\boldsymbol{e}_1$、$\boldsymbol{e}_2$、$\boldsymbol{e}_3$ 也可以组合成三维空间中的任意向量。

三维直角坐标系中任意向量 $\boldsymbol{x} = [x_1, x_2, x_3]^{\mathrm{T}}$ 可以写成

$$\boldsymbol{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = x_1 \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} + x_2 \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} + x_3 \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} = x_1\boldsymbol{e}_1 + x_2\boldsymbol{e}_2 + x_3\boldsymbol{e}_3 \tag{4}$$

这体现的是**向量分解** (vector decomposition)，这是本章后续要讨论的话题。

⚠ 注意，RGB 颜色空间因其有界性，严格来说不能称为向量空间，更像是一个色彩的三维立方体。不过，它依然能启发我们理解向量空间相关概念。

平面直角坐标系中的 $\boldsymbol{e}_1$ 是 2 维向量，三维直角坐标系中 $\boldsymbol{e}_1$ 是 3 维向量；以此类推，n 维空间中的 $\boldsymbol{e}_1$ 是 n 维向量。

必要时，为了区分，我们会增加上标把它们分别记作 $\boldsymbol{e}_1^{(2)}$、$\boldsymbol{e}_1^{(3)}$、$\boldsymbol{e}_1^{(4)}$、$\boldsymbol{e}_1^{(n)}$。比如，

$$\boldsymbol{e}_1^{(2)} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \quad \boldsymbol{e}_1^{(3)} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \quad \boldsymbol{e}_1^{(4)} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \quad \boldsymbol{e}_1^{(n)} = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \tag{5}$$

向量长度

有了直角坐标系，我们可以很容易计算得到向量长度。

利用勾股定理，n 维向量 $\boldsymbol{x}$ 的大小具体为

$$\|\boldsymbol{x}\| = \|\boldsymbol{x}\|_2 = \sqrt{x_1^2 + x_2^2 + \cdots + x_n^2} \tag{6}$$

举个例子，给定如下 2 维向量

$$\boldsymbol{x} = \begin{bmatrix} 3 \\ 4 \end{bmatrix} \tag{7}$$

这个向量的大小 (长度) 为

$$\|\boldsymbol{x}\| = \|\boldsymbol{x}\|_2 = \sqrt{3^2 + 4^2} = 5 \tag{8}$$

在线性代数中，向量的大小 (长度) 常称作 **L^2 范数** (L2 norm) 、**欧几里得范数** (Euclidean norm)、**模**。本章后续将专门介绍向量范数这一概念。

⚠ 注意，本书中绝对值用 $|-5|$，而向量大小运算符用 $\|\boldsymbol{x}\|$。$\|\boldsymbol{x}\|_2$ 下角标中的 2 代表 L^2 范数，常常省略作 $\|\boldsymbol{x}\|$。

3 维向量 $\boldsymbol{x}$ 的长度为

$$\|\boldsymbol{x}\| = \|\boldsymbol{x}\|_2 = \sqrt{x_1^2 + x_2^2 + x_3^2} \tag{9}$$

比如，3 维列向量 $[1, 2, 2]^{\mathrm{T}}$ 的长度 (大小、L^2 范数、模) 为

$$\|\boldsymbol{x}\| = \|\boldsymbol{x}\|_2 = \sqrt{1^2 + 2^2 + 2^2} = 3 \tag{10}$$

计算向量长度

代码 2 展示如何用 numpy.linalg.norm() 计算向量 (一维数组) 长度。

代码 2. 计算向量大小 | LA_01_02_03.ipynb

```python
## 初始化
import numpy as np

## 定义向量 (一维数组)
a = np.array([3, 4])
b = np.array([1, 2, 2])

## 计算向量长度 （L2范数）
length_a = np.linalg.norm(a)
np.sqrt(a[0]**2 + a[1]**2)

length_b = np.linalg.norm(b)
np.sqrt(b[0]**2 + b[1]**2 + b[2]**2)
```

下面聊聊代码 2。

ⓐ 用 numpy.array() 定义两个一维数组。

ⓑ 用 numpy.linalg.norm() 计算 a 的长度。

ⓒ 则手动验算 a 的长度。a[0] 表示 a 的第一个分量 3，a[1] 表示 a 的第二个分量 4。运算符**完成平方运算。函数 numpy.sqrt() 完成开平方运算；sqrt 代表 square root，即平方根。

ⓓ 用 numpy.linalg.norm() 计算 b 的长度。

ⓔ 手动验算 b 的长度。

请大家用 DeepSeek/ChatGPT 等工具完成本节如下习题。

Q1. 欧几里得几何的公理和性质有哪些？

Q2. 平面直角坐标系的四个象限指的是什么？

Q3. 三维直角坐标系八个卦限指的是什么？

Q4. 请 DeepSeek/ChatGPT 逐行注释下例，并学习绘制箭头图。
https://matplotlib.org/stable/gallery/images_contours_and_fields/quiver_simple_demo.html

Q5. 请 DeepSeek/ChatGPT 逐行注释下例，并学习绘制平面散点图。

https://seaborn.pydata.org/generated/seaborn.scatterplot.html

Q6. 请 DeepSeek/ChatGPT 逐行注释下例，并学习绘制三维散点图。

https://matplotlib.org/stable/gallery/mplot3d/scatter3d.html

Q7. 用 numpy.norm() 函数计算如下向量的长度，并用勾股定理验证。

► [2, 4, 4]

► [2, 3, 6]

► [1, 4, 8]

Q8. 请自学如何用 numpy.meshgrid() 创建三维数组。

Q9. 请自学如何对三维数组进行索引、切片。