

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

8.3 旋转



本节你将掌握的核心技能：

- ▶ 矩阵乘法实现绕原点平面旋转。
- ▶ 旋转矩阵为正交矩阵，行列式为 1。
- ▶ 连续平面旋转：满足交换律。
- ▶ 格拉姆矩阵和向量内积关系。
- ▶ 交换旋转、缩放的次序会影响结果，矩阵乘法一般不满足交换律。
- ▶ 矩阵乘法实现三维绕轴旋转。

旋转变换在机器学习算法、计算机图形学、机器人、物理学等领域中都有广泛应用。本书后文中，大家会在谱分解、主成分分析、奇异值分解等话题中看到旋转的应用场景。旋转通常用矩阵乘法运算完成。本节先讲解平面旋转，再讲解三维旋转。

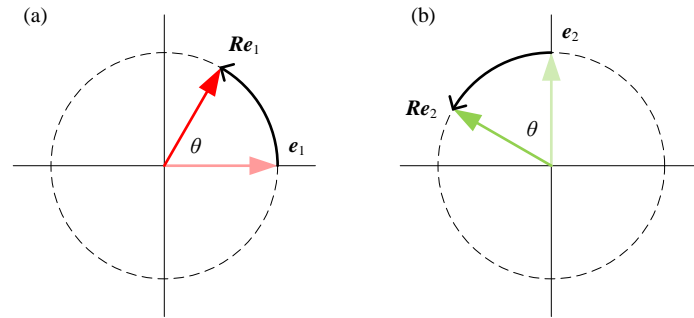
平面旋转

平面绕原点逆时针旋转角度 θ 对应的变换矩阵为

$$\mathbf{R}(\theta) = \begin{bmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{bmatrix} \quad (1)$$

在平面上，给定一个列向量 \mathbf{x} ，绕原点逆时针旋转角度 θ 后的新坐标为

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \quad (2)$$

图 1. R 对 e_1 、 e_2 的作用

R 对 e_1 作用的结果为

$$Re_1 = \begin{bmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} \cos \theta \\ \sin \theta \end{bmatrix} \quad (3)$$

上式相当于提取了 R 的第一列列向量。

如图 1 (a) 所示, $\begin{bmatrix} \cos \theta \\ \sin \theta \end{bmatrix}$ 是 e_1 绕原点逆时针旋转 θ 的新位置。

R 对 e_2 作用

$$Re_2 = \begin{bmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} -\sin \theta \\ \cos \theta \end{bmatrix} \quad (4)$$

上式相当于提取了 R 的第二列列向量。

如图 1 (b) 所示, $\begin{bmatrix} -\sin \theta \\ \cos \theta \end{bmatrix}$ 是 e_2 绕原点逆时针旋转 θ 的新位置。

图 2 给出两个例子。

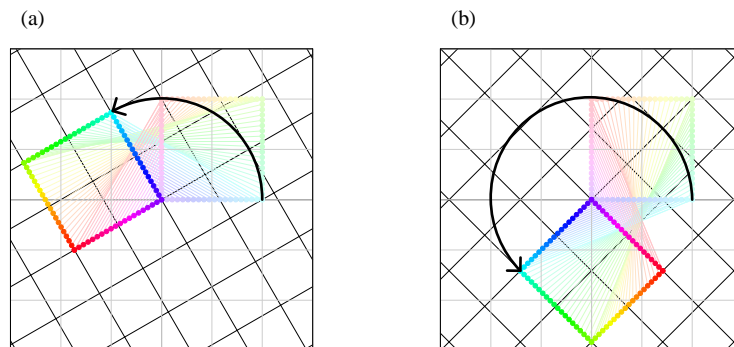


图 2. 旋转的两个例子

图 2 (a) 对应的旋转矩阵为

$$\mathbf{R} = \begin{bmatrix} \cos \frac{2\pi}{3} & -\sin \frac{2\pi}{3} \\ \sin \frac{2\pi}{3} & \cos \frac{2\pi}{3} \end{bmatrix} = \begin{bmatrix} -\frac{1}{2} & -\frac{\sqrt{3}}{2} \\ \frac{\sqrt{3}}{2} & -\frac{1}{2} \end{bmatrix} \quad (5)$$

图 2 (b) 对应的旋转矩阵为

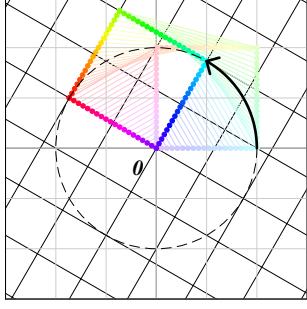
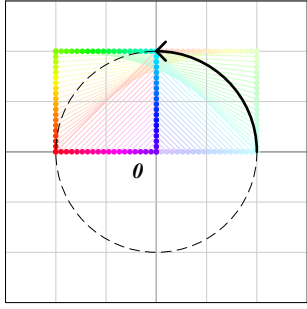
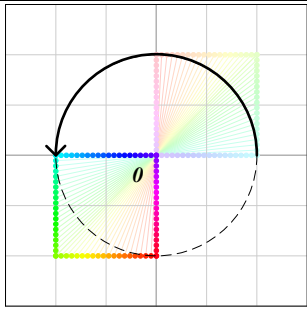
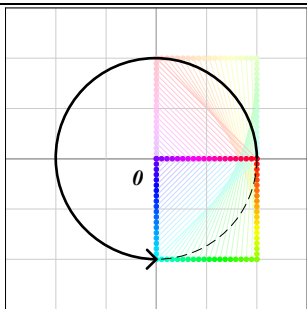
$$\mathbf{R} = \begin{bmatrix} \cos\left(-\frac{3\pi}{4}\right) & -\sin\left(-\frac{3\pi}{4}\right) \\ \sin\left(-\frac{3\pi}{4}\right) & \cos\left(-\frac{3\pi}{4}\right) \end{bmatrix} = \begin{bmatrix} -\frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} \\ -\frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} \end{bmatrix} \quad (6)$$

表 1 所示为几个常见的旋转角度对应的旋转矩阵、图形变换。

? 请大家分析表 1 中每个旋转矩阵，计算行列式、计算逆矩阵、对 \mathbf{e}_1 、 \mathbf{e}_2 作用。

表 1. 几个不同旋转角度

逆时针旋转	矩阵	图形
0 度 相当于没变化	$\begin{bmatrix} \cos 0 & -\sin 0 \\ \sin 0 & \cos 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$ <p>单位矩阵</p>	
30 度	$\begin{bmatrix} \cos \frac{\pi}{6} & -\sin \frac{\pi}{6} \\ \sin \frac{\pi}{6} & \cos \frac{\pi}{6} \end{bmatrix} = \begin{bmatrix} \frac{\sqrt{3}}{2} & -\frac{1}{2} \\ \frac{1}{2} & \frac{\sqrt{3}}{2} \end{bmatrix}$	
45 度	$\begin{bmatrix} \cos \frac{\pi}{4} & -\sin \frac{\pi}{4} \\ \sin \frac{\pi}{4} & \cos \frac{\pi}{4} \end{bmatrix} = \begin{bmatrix} \frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} \\ \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \end{bmatrix}$	

60 度	$\begin{bmatrix} \cos \frac{\pi}{3} & -\sin \frac{\pi}{3} \\ \sin \frac{\pi}{3} & \cos \frac{\pi}{3} \end{bmatrix} = \begin{bmatrix} \frac{1}{2} & -\frac{\sqrt{3}}{2} \\ \frac{\sqrt{3}}{2} & \frac{1}{2} \end{bmatrix}$	
90 度	$\begin{bmatrix} \cos \frac{\pi}{2} & -\sin \frac{\pi}{2} \\ \sin \frac{\pi}{2} & \cos \frac{\pi}{2} \end{bmatrix} = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}$	
180 度	$\begin{bmatrix} \cos \pi & -\sin \pi \\ \sin \pi & \cos \pi \end{bmatrix} = \begin{bmatrix} -1 & 0 \\ 0 & -1 \end{bmatrix}$	
270 度	$\begin{bmatrix} \cos \frac{3\pi}{2} & -\sin \frac{3\pi}{2} \\ \sin \frac{3\pi}{2} & \cos \frac{3\pi}{2} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}$	

逆矩阵：逆操作

对旋转矩阵 $R(\theta)$ 求逆

$$R^{-1}(\theta) = \begin{bmatrix} \cos \theta & \sin \theta \\ -\sin \theta & \cos \theta \end{bmatrix} \quad (7)$$

图 2 所示为旋转及逆操作。容易发现，(7) 这个矩阵实际上表示表示绕原点顺时针旋转 θ 度。

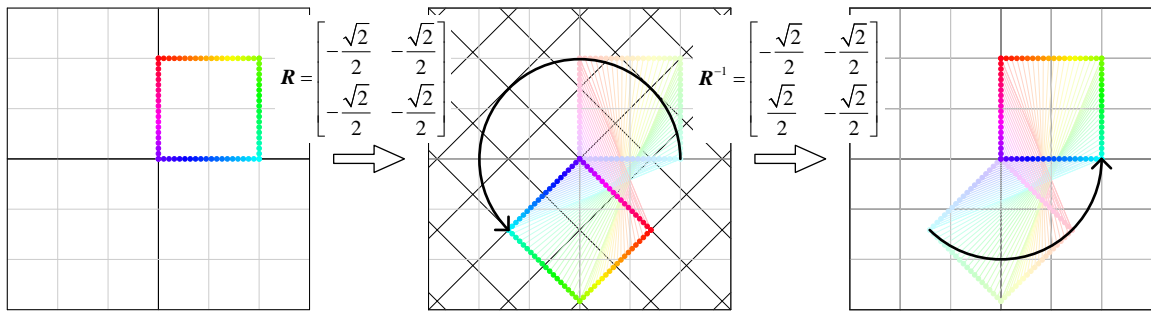


图 3. 旋转的逆运算

顺时针旋转 θ 对应逆时针旋转 $-\theta$ ，对应的旋转矩阵为

$$\begin{bmatrix} \cos(-\theta) & -\sin(-\theta) \\ \sin(-\theta) & \cos(-\theta) \end{bmatrix} = \begin{bmatrix} \cos \theta & \sin \theta \\ -\sin \theta & \cos \theta \end{bmatrix} \quad (8)$$

(8) 和 (7) 矩阵一样。

行列式

二维旋转矩阵的行列式为

$$\det(\mathbf{R}(\theta)) = \begin{vmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{vmatrix} = \cos^2 \theta + \sin^2 \theta = 1 \quad (9)$$

这说明平面旋转不改变几何形状的面积；这个（绕原点）旋转也不改变向量先后顺序。

正交矩阵：规范正交基

比较(8) 和 (7)，我们发现旋转矩阵的逆矩阵 \mathbf{R}^{-1} 是其转置矩阵 \mathbf{R}^T ；

$$\mathbf{R}^{-1} = \mathbf{R}^T \quad (10)$$

也就是说

$$\mathbf{R} @ \mathbf{R}^T = \mathbf{R}^T @ \mathbf{R} = \mathbf{I} \quad (11)$$

这意味着 \mathbf{R} 为**正交矩阵** (orthogonal matrix)。

把旋转矩阵 \mathbf{R} 按列向量展开写成如下形式

$$\mathbf{R} = \begin{bmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{bmatrix} = \begin{bmatrix} \cos \theta \\ \sin \theta \end{bmatrix} \begin{bmatrix} -\sin \theta \\ \cos \theta \end{bmatrix} = [\mathbf{r}_1 \quad \mathbf{r}_2] \quad (12)$$

我们发现的每一列都是单位向量（方向向量）， L^2 向量范数（向量长度、向量大小、欧几里得范数）的平方为 1，即

$$\begin{aligned} \|\mathbf{r}_1\|_2^2 &= \mathbf{r}_1^T @ \mathbf{r}_1 = 1 \\ \|\mathbf{r}_2\|_2^2 &= \mathbf{r}_2^T @ \mathbf{r}_2 = 1 \end{aligned} \quad (13)$$



请大家自行计算 (13)。



请大家回顾本书前文向量范数、矩阵乘法和内积关系等内容。

此外，(13) 列向量正交，即

$$\mathbf{r}_2^T @ \mathbf{r}_1 = \mathbf{r}_1^T @ \mathbf{r}_2 = 0 \quad (14)$$

这意味着 $\mathbf{R} = [\mathbf{r}_1, \mathbf{r}_2]$ 为**规范正交基** (orthonormal basis)。比如，表 1 中规范正交基的网格都是单位正方形。

特别地，当 θ 为 0 时， $\mathbf{R} = \mathbf{I}$ ，即

$$\begin{bmatrix} \cos 0 & -\sin 0 \\ \sin 0 & \cos 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = [\mathbf{e}_1 \quad \mathbf{e}_2] \quad (15)$$

这时我们得到的是**标准正交基** (standard basis, natural basis)。



请大家回顾本书前文基底、规范正交基、标准正交基等内容。

格拉姆矩阵

有了 (13)、(14)，让我们看一个“有趣”的矩阵乘法

$$\mathbf{R}^T \mathbf{R} = \begin{bmatrix} \mathbf{r}_1^T \\ \mathbf{r}_2^T \end{bmatrix} @ [\mathbf{r}_1 \quad \mathbf{r}_2] = \begin{bmatrix} \mathbf{r}_1^T @ \mathbf{r}_1 & \mathbf{r}_1^T @ \mathbf{r}_2 \\ \mathbf{r}_2^T @ \mathbf{r}_1 & \mathbf{r}_2^T @ \mathbf{r}_2 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = \mathbf{I} \quad (16)$$

我们管 $\mathbf{R}^T \mathbf{R}$ 叫做 \mathbf{R} 的**格拉姆矩阵** (Gram matrix)。

在 $\mathbf{R}^T \mathbf{R}$ 这个格拉姆矩阵的对角线上，我们看到列向量自身的内积，结果等价列向量 L^2 范数的平方。

而 $\mathbf{R}^T \mathbf{R}$ 非对角线元素则对应 \mathbf{R} 的成对向量内积。



本书后文将会专门讲解格拉姆矩阵。

连续旋转

如果进行两次绕原点逆时针旋转，先旋转 α 、后旋转 β ，则总旋转矩阵为

$$\mathbf{R}(\beta)\mathbf{R}(\alpha) = \begin{bmatrix} \cos \beta & -\sin \beta \\ \sin \beta & \cos \beta \end{bmatrix} \begin{bmatrix} \cos \alpha & -\sin \alpha \\ \sin \alpha & \cos \alpha \end{bmatrix} \quad (17)$$

使用三角恒等式简化上式得到

$$\mathbf{R}(\beta)\mathbf{R}(\alpha) = \begin{bmatrix} \cos(\alpha + \beta) & -\sin(\alpha + \beta) \\ \sin(\alpha + \beta) & \cos(\alpha + \beta) \end{bmatrix} \quad (18)$$

反过来，先旋转 β 、后旋转 α ，则总旋转矩阵为

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

$$R(\alpha)R(\beta) = \begin{bmatrix} \cos \alpha & -\sin \alpha \\ \sin \alpha & \cos \alpha \end{bmatrix} \begin{bmatrix} \cos \beta & -\sin \beta \\ \sin \beta & \cos \beta \end{bmatrix} = \begin{bmatrix} \cos(\alpha + \beta) & -\sin(\alpha + \beta) \\ \sin(\alpha + \beta) & \cos(\alpha + \beta) \end{bmatrix} \quad (19)$$

这说明平面绕原点旋转变换是可累加的，即多次旋转等价于单次旋转对应角度的累加。这也是矩阵乘法满足交换律的特殊情况。

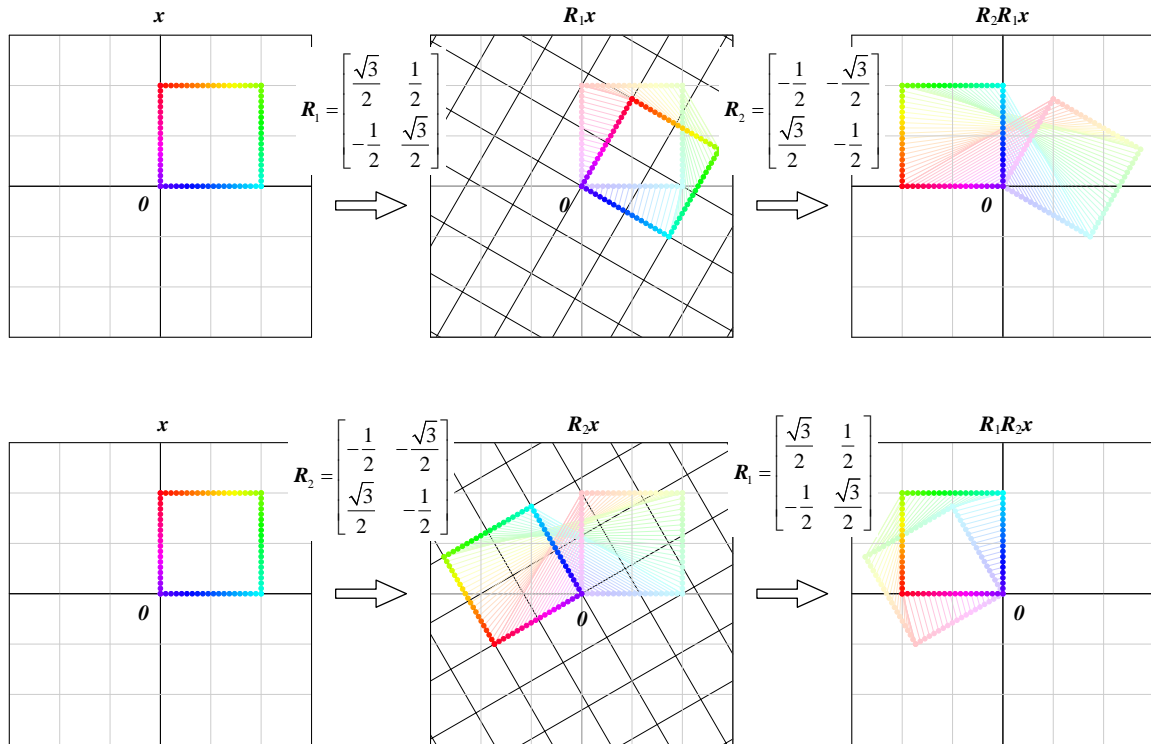


图 4. 两个 2×2 旋转矩阵连乘满足交换律

本书前文提到过矩阵乘法一般不满足交换律，即 $AB \neq BA$ 。

对比图 5、图 6，我们可以发现，先缩放、再旋转的结果不同于先旋转、再缩放。

⚠ 反复强调，对于矩阵乘法 $ABCx$ ，对列向量 x 的作用顺序从左向右，即 C 、 B 、 A 。

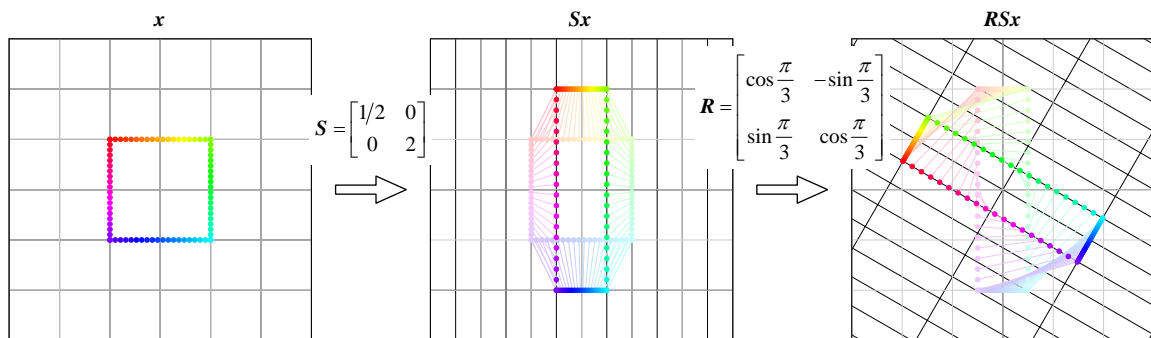


图 5. 先缩放，再旋转

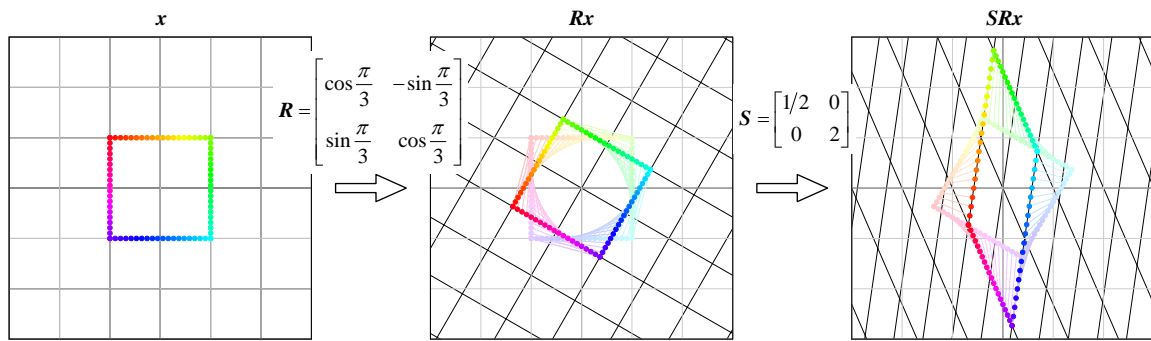


图 6. 先旋转, 再缩放

三维旋转

如图 7 所示, 在三维空间中, 几何体可以独立绕 x_1 、 x_2 、 x_3 轴分别独立旋转。

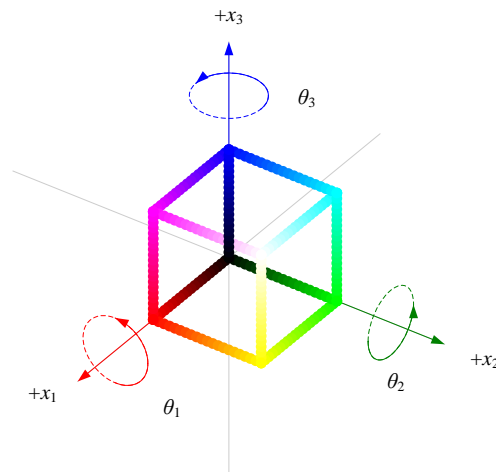


图 7. 几何体绕三轴独立旋转

在三维空间中, 绕各轴的旋转正负方向遵循右手定则。右手的拇指指向旋转轴的正方向。其余四指弯曲时指示的方向为正向旋转 (逆时针方向); 与正向旋转相反的方向为负向旋转 (顺时针方向)。

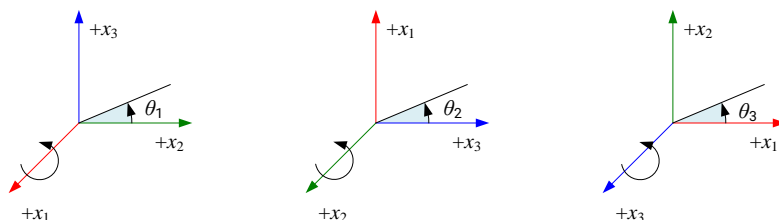


图 8. 三个旋转角度

绕 x_1 轴旋转

绕 x_1 轴逆时针旋转角度 θ_1 对应的变换矩阵为

$$\mathbf{R}_1 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos \theta_1 & -\sin \theta_1 \\ 0 & \sin \theta_1 & \cos \theta_1 \end{bmatrix} \quad (20)$$

如图 9 所示，三维列向量 \mathbf{x} 绕 x_1 轴逆时针旋转角度 θ_1 之后的坐标为

$$\underbrace{\begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos \theta_1 & -\sin \theta_1 \\ 0 & \sin \theta_1 & \cos \theta_1 \end{bmatrix}}_{\mathbf{R}_1} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} z_1 \\ z_2 \\ z_3 \end{bmatrix} \quad (21)$$

展开上式之后，我们得到三个等式

$$\begin{cases} z_1 = x_1 \\ z_2 = x_2 \cdot \cos \theta_1 - x_3 \cdot \sin \theta_1 \\ z_3 = x_2 \cdot \sin \theta_1 + x_3 \cdot \cos \theta_1 \end{cases} \quad (22)$$

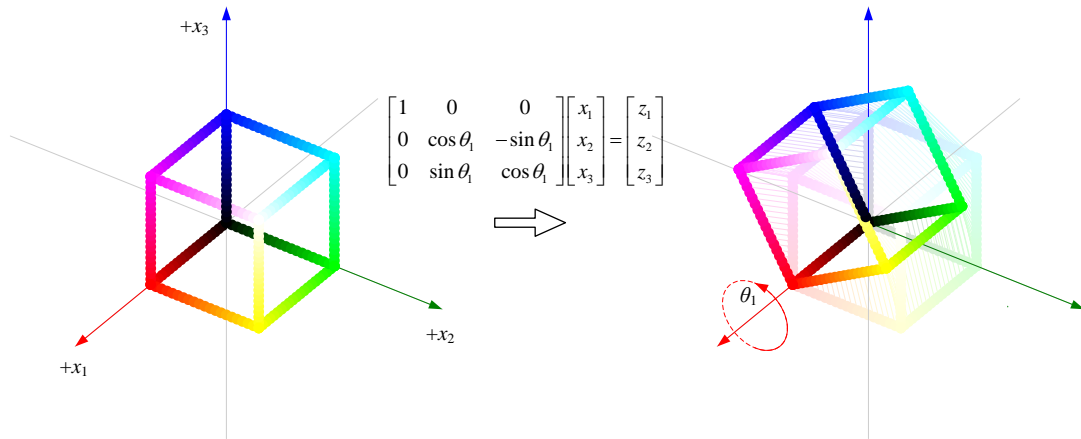


图 9. 绕 x_1 轴旋转，三维空间

\mathbf{R}_1 的逆矩阵为

$$\mathbf{R}_1^{-1} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos \theta_1 & \sin \theta_1 \\ 0 & -\sin \theta_1 & \cos \theta_1 \end{bmatrix} \quad (23)$$

和二维旋转矩阵一样， \mathbf{R}_1 的逆对应的几何操作是绕 x_1 轴顺时针旋转角度 θ_1 。

计算 \mathbf{R}_1 的格拉姆矩阵

$$\mathbf{R}_1^T \mathbf{R}_1 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos \theta_1 & -\sin \theta_1 \\ 0 & \sin \theta_1 & \cos \theta_1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos \theta_1 & \sin \theta_1 \\ 0 & -\sin \theta_1 & \cos \theta_1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (24)$$

反过来

$$\mathbf{R}_1 \mathbf{R}_1^T = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos \theta_1 & \sin \theta_1 \\ 0 & -\sin \theta_1 & \cos \theta_1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos \theta_1 & -\sin \theta_1 \\ 0 & \sin \theta_1 & \cos \theta_1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (25)$$

这意味着 \mathbf{R}_1 也是正交矩阵。在三维空间中， \mathbf{R}_1 为规范正交基。

下面让我们回顾本书前文讲过的拉普拉斯展开 (Laplace expansion) 计算 (20) 行列式。

选择 (20) 第一行展开，因为这一行 0 最多，计算量最小。

$$\det(\mathbf{R}_1) = 1 \times \begin{vmatrix} \cos \theta_1 & -\sin \theta_1 \\ \sin \theta_1 & \cos \theta_1 \end{vmatrix} = 1 \quad (26)$$

这意味旋转过程，几何体体积不发生变换。这和我们图 9 的观察完全一致。

绕 x_2 、 x_3 轴旋转

绕 x_2 轴逆时针旋转角度 θ_2 对应的变换矩阵为


$$\mathbf{R}_2 = \begin{bmatrix} \cos \theta_2 & 0 & \sin \theta_2 \\ 0 & 1 & 0 \\ -\sin \theta_2 & 0 & \cos \theta_2 \end{bmatrix} \quad (27)$$

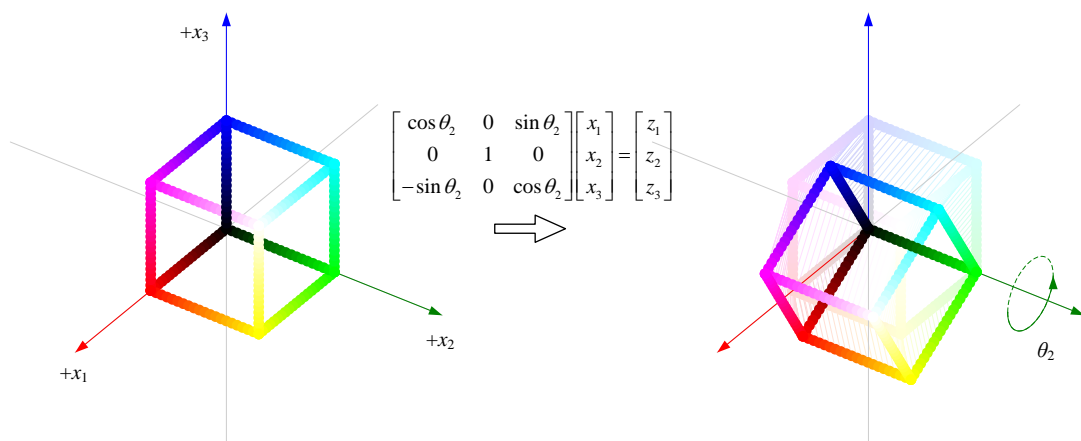
绕 x_2 轴逆时针旋转角度 θ_2 之后的坐标为

$$\underbrace{\begin{bmatrix} \cos \theta_2 & 0 & \sin \theta_2 \\ 0 & 1 & 0 \\ -\sin \theta_2 & 0 & \cos \theta_2 \end{bmatrix}}_{\mathbf{R}_2} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} z_1 \\ z_2 \\ z_3 \end{bmatrix} \quad (28)$$

展开之后，我们得到三个等式

$$\begin{cases} z_1 = x_1 \cdot \cos \theta_2 + x_3 \cdot \sin \theta_2 \\ z_2 = x_2 \\ z_3 = -x_1 \cdot \sin \theta_2 + x_3 \cdot \cos \theta_2 \end{cases} \quad (29)$$

 请大家计算 (27) 矩阵的逆，并判断它为 正交矩阵，再用拉普拉斯展开计算它的行列式。

图 10. 绕 x_2 轴旋转，三维空间

绕 x_3 轴逆时针旋转角度 θ_2 对应的变换矩阵为

$$\mathbf{R}_3 = \begin{bmatrix} \cos \theta_3 & -\sin \theta_3 & 0 \\ \sin \theta_3 & \cos \theta_3 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (30)$$

绕 x_3 轴逆时针旋转角度 θ_3 之后的坐标为

$$\underbrace{\begin{bmatrix} \cos \theta_3 & -\sin \theta_3 & 0 \\ \sin \theta_3 & \cos \theta_3 & 0 \\ 0 & 0 & 1 \end{bmatrix}}_{\mathbf{R}_3} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} z_1 \\ z_2 \\ z_3 \end{bmatrix} \quad (31)$$

展开之后，我们得到三个等式

$$\begin{cases} z_1 = x_1 \cdot \cos \theta_3 - x_2 \cdot \sin \theta_3 \\ z_2 = x_1 \cdot \sin \theta_3 + x_2 \cdot \cos \theta_3 \\ z_3 = x_3 \end{cases} \quad (32)$$

? 同样请大家计算 (30) 矩阵的逆，并判断它为正交矩阵，再用拉普拉斯展开计算它的行列式。

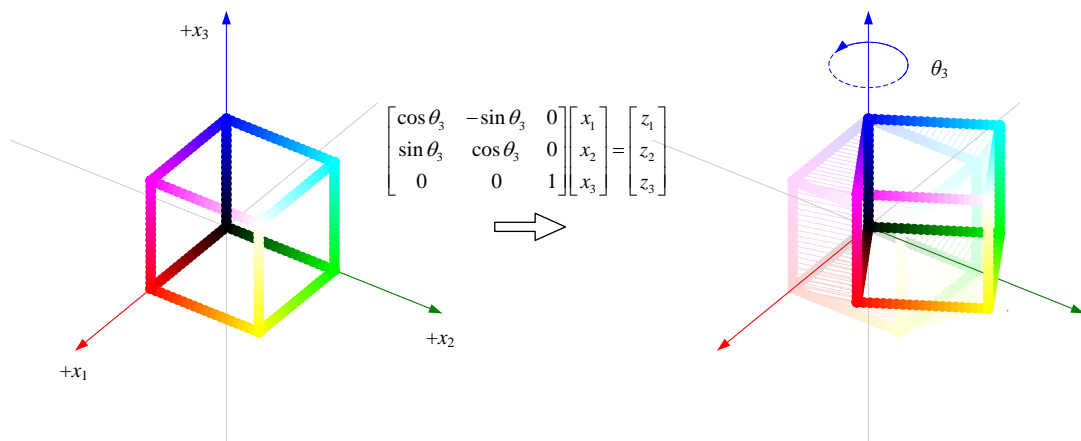


图 11. 绕 x_3 轴旋转，三维空间

⚠ 注意，三维旋转，绕不同轴的旋转先后顺序不能交换；只有绕同一轴旋转才能交换顺序，这一点类似平面旋转。

可视化三维绕轴旋转

代码 1 为可视化三维绕轴旋转的自定义函数。

a 只是把原始立方体的顶点数据赋值给一个变量 `colors`，用于后面设置每个点的颜色。单位正方体的为 RGB 空间，每个顶点的坐标就是颜色色号。

b 定义一个列表，列出立方体的 12 条边，每条边由两个点的索引构成。比如 `[0, 1]` 表示用第 0 个点和第 1 个点连成一条边。总共 8 个点，组合成 12 条边。

c 使用 `for` 循环画立方体原始状态所有边。`original_cube[edge]` 会取出边的两个顶点，然后 `zip(*...)` 会把它们的 `x`、`y`、`z` 坐标分别提取出来。前面的 `*` 是解包操作，告诉 `plot` 函数“把这些分别当作 `x`、`y`、`z` 坐标”。`color="gray"` 设置线为灰色，`linestyle="dotted"` 设置为虚线。

d 和 **c** 类似，绘制几何变换后的形状。

e 用 `scatter()` 绘制原始立方体顶点，每个顶点设置不同颜色。

f 和 **e** 类似，用 `scatter()` 绘制变换后立方体顶点

代码 1. 自定义可视化函数 |  LA_08_03_01.ipynb

```

## 可视化函数
def plot_cube(ax, original_cube, transformed_cube):
    colors = original_cube

    # 定义每个卦限的框线顶点对
    edges = [[0, 1], [1, 2], [2, 3], [3, 0],
             [0, 4], [1, 5], [2, 6], [3, 7],
             [4, 5], [5, 6], [6, 7], [7, 4]]

    # 绘制原始立方体框线
    for edge in edges:
        ax.plot(*zip(*original_cube[edge]),
                color="gray", linestyle="dotted")

    # 绘制变换后的立方体框线
    for edge in edges:
        ax.plot(*zip(*transformed_cube[edge]),
                color="black", linestyle="-")

    # 绘制顶点
    ax.scatter(original_cube[:,0], original_cube[:,1],
               original_cube[:,2],
               facecolor=colors, s=50, marker='o', edgecolor='k')

    ax.scatter(transformed_cube[:,0], transformed_cube[:,1],
               transformed_cube[:,2],
               facecolor=colors, s=50, marker='o', edgecolor='k')

    ax.set_xlim(-2, 2); ax.set_ylim(-2, 2); ax.set_zlim(-2, 2)
    ax.axis(False); ax.set_box_aspect([1, 1, 1])
    ax.view_init(elev=30, azimuth=30)
    ax.set_proj_type('ortho')
    ax.plot((-2,2),(0,0),(0,0),color = 'k')
    ax.plot((0,0),(-2,2),(0,0),color = 'k')
    ax.plot((0,0),(0,0),(-2,2),color = 'k')

```

代码 2 调用代码 1 的自定义函数，绘制图 12。代码 2 很简单，请大家逐行注释。

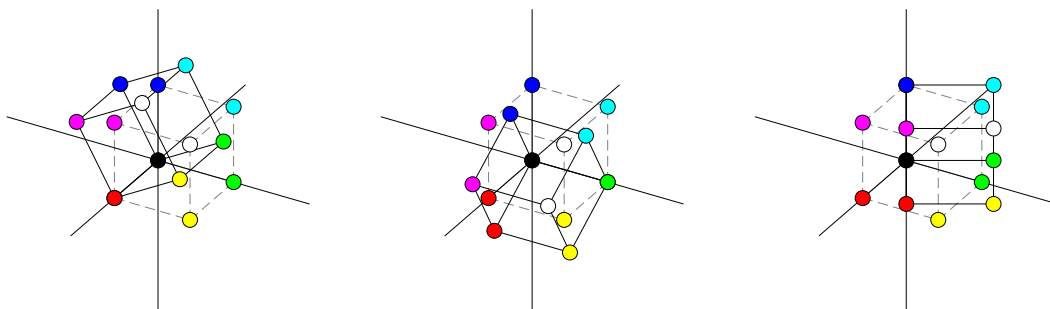


图 12. 三维绕轴旋转案例

代码 2. 三维绕轴旋转 | LA_08_03_01.ipynb

```

## 定义单位立方体的 8 个顶点坐标
cube = np.array([[0, 0, 0], [1, 0, 0], [1, 1, 0], [0, 1, 0],
                 [0, 0, 1], [1, 0, 1], [1, 1, 1], [0, 1, 1]])

## 绕 x1 轴旋转
fig = plt.figure(figsize=(5, 5))
ax = fig.add_subplot(111, projection='3d')

theta_1 = np.pi / 6
R_x1 = np.array([[1, 0, 0],
                 [0, np.cos(theta_1), -np.sin(theta_1)],
                 [0, np.sin(theta_1), np.cos(theta_1)]])

rotated_cube = cube @ R_x1.T
plot_cube(ax, cube, rotated_cube)

## 绕 x2 轴旋转
fig = plt.figure(figsize=(5, 5))
ax = fig.add_subplot(111, projection='3d')

theta_2 = np.pi / 6
R_x2 = np.array([[np.cos(theta_2), 0, np.sin(theta_2)],
                 [0, 1, 0],
                 [-np.sin(theta_2), 0, np.cos(theta_2)]])

rotated_cube = cube @ R_x2.T
plot_cube(ax, cube, rotated_cube)

## 绕 x3 轴旋转
fig = plt.figure(figsize=(5, 5))
ax = fig.add_subplot(111, projection='3d')

theta_3 = np.pi / 6
R_x3 = np.array([[np.cos(theta_3), -np.sin(theta_3), 0],
                 [np.sin(theta_3), np.cos(theta_3), 0],
                 [0, 0, 1]])

rotated_cube = cube @ R_x3.T
plot_cube(ax, cube, rotated_cube)

```



请大家用 DeepSeek/ChatGPT 等工具完成本节如下习题。

- Q1.** 请编写 Python 代码用矩阵乘法完成，给定点 (3, 4) 绕原点逆时针旋转 90 度，算出旋转后的坐标。
- Q2.** 证明二维旋转矩阵的逆矩阵是其转置矩阵。
- Q3.** 平面旋转，先旋转 30 度、再旋转 60 度，分别计算旋转矩阵，再计算复合几何变换矩阵。
- Q4.** 编写一个 Python 函数 `rotate_point(x1, x2, theta)`，输入点 (x1, x2) 和角度 `theta` (弧度)，返回平面绕原点旋转后的坐标。

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

Q5. 给定一组平面点坐标 $\text{points} = [(1, 0), (0, 1), (-1, 0), (0, -1), (0, 0)]$ ，编写 Python 代码逆时针旋转 45° 并输出旋转后的新坐标。

Q6. 编写 Python 代码，使用 matplotlib 绘制一个单位正方形，绕原点逆时针旋转 60° 后再绘制旋转后的单位正方形。

Q7. 写代码，比较如下两个不同顺序的三维旋转是否相同

► 先绕 x_1 旋转 90 度，再绕 x_2 轴旋转 90 度；

► 先绕 x_2 旋转 90 度，再绕 x_1 轴旋转 90 度；

Q8. 了解什么是罗德里格旋转 (Rodrigues' rotation)。

Q9. 了解什么是单位四元数 (Unit quaternion)，了解如何用它完成三维空间旋转。

Q10. 了解万向锁问题 (Gimbal Lock)。

Q11. 了解复数，并了解复数和旋转有什么联系。