

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

1.3 向量加减法

向量加法、减法是线性代数的核心运算之一，本节将介绍它们的基本定义、运算规则及重要性质。在几何视角下，我们介绍了平行四边形法则和三角形法则，分别用于理解向量加法的几何意义。平行四边形法则通过构造平行四边形来表示两个向量之和，而三角形法则通过首尾相连的方法求和。此外，三维线性无关的向量加法可用平行六面体法则直观表示，该方法适用于三个向量的加法运算，并展示了结合律的几何含义。在向量减法部分，我们将减法理解为加上相反向量的过程。

向量加法

对于两个维数相同的向量，**向量加法** (vector addition) 是指将它们对应位置的分量分别相加，得到一个新的向量。

比如，给定如下两个 n 维列向量

$$\mathbf{a} = \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_n \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix} \quad (1)$$

向量 \mathbf{a} 、 \mathbf{b} 之和为对应位置分量相加，结果为相同维数向量

$$\mathbf{a} + \mathbf{b} = \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_n \end{bmatrix} + \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix} = \begin{bmatrix} a_1 + b_1 \\ a_2 + b_2 \\ \vdots \\ a_n + b_n \end{bmatrix} \quad (2)$$

如下代码展示如下用 NumPy 完成向量加法。代码很简单，请大家自行注释学习。

代码 1. 向量加法 |  LA_01_03_01.ipynb

```
## 初始化
import numpy as np

## 定义向量
a_vec = np.array([4, 1])
b_vec = np.array([1, 2])

## 向量加法
a_plus_b = a_vec + b_vec
```



RGB 颜色向量为例

在 RGB 颜色模式中，红色向量 e_1 、绿色向量 e_2 、蓝色向量 e_3 代表了三种基本色。向量加法可以直观地表现为颜色的混合。

如图 1 所示，我们可以利用向量加法规则，通过不同的组合方式获得新的颜色。

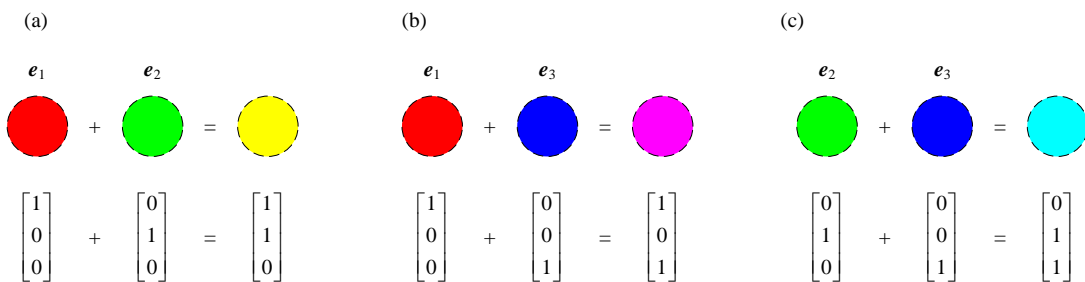


图 1. 向量加法：混合两种基本色

例如，如图 1 (a) 所示，红色向量 e_1 、绿色向量 e_2 相加，得到黄色向量，即

$$e_1 + e_2 = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix} \quad (3)$$

如图 1 (b)，红色向量 e_1 、蓝色向量 e_3 相加，得到品红；如图 1 (c)，绿色向量 e_2 、蓝色向量 e_3 相加，得到青色向量。

请大家自己写出这两个向量加法的运算过程。

如图 2 所示，当我们将红色向量 e_1 、绿色向量 e_2 、蓝色向量 e_3 都相加，即 $e_1 + e_2 + e_3$ ，就会得到白色向量，即

$$e_1 + e_2 + e_3 = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \quad (4)$$

这意味着在 RGB 色彩空间中，三种基色的光混合后形成白光。这种运算不仅符合向量加法的规则，也展现了 RGB 颜色模式中光的叠加原理。

$$\begin{array}{c}
 \mathbf{e}_1 \quad \mathbf{e}_2 \quad \mathbf{e}_3 \\
 \text{red circle} + \text{green circle} + \text{blue circle} = \text{white circle} \\
 \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}
 \end{array}$$

图 2. 向量加法：混合三种基本色

几何视角：平行四边形法则

平行四边形法则 (parallelogram law for vector addition) 是一种几何方法，用于计算两个向量的加法。

这种方法通过对齐两个向量的起点，将它们作为相邻边构造平行四边形，其对角线从起点 (原点) 指向对角顶点，即为向量加法的结果。

图 3 所示为利用平行四边形法则求解两个向量之和，其中每个子图中非零向量 a 、 b 之间表现出不同的关系。

图 3 (a) 为单位正方形；图 3 (b) 为正方形；图 3 (c) 为矩形；图 3 (d) 的平行四边两条边平行于横轴；图 3 (e) 的两条边平行于纵轴。

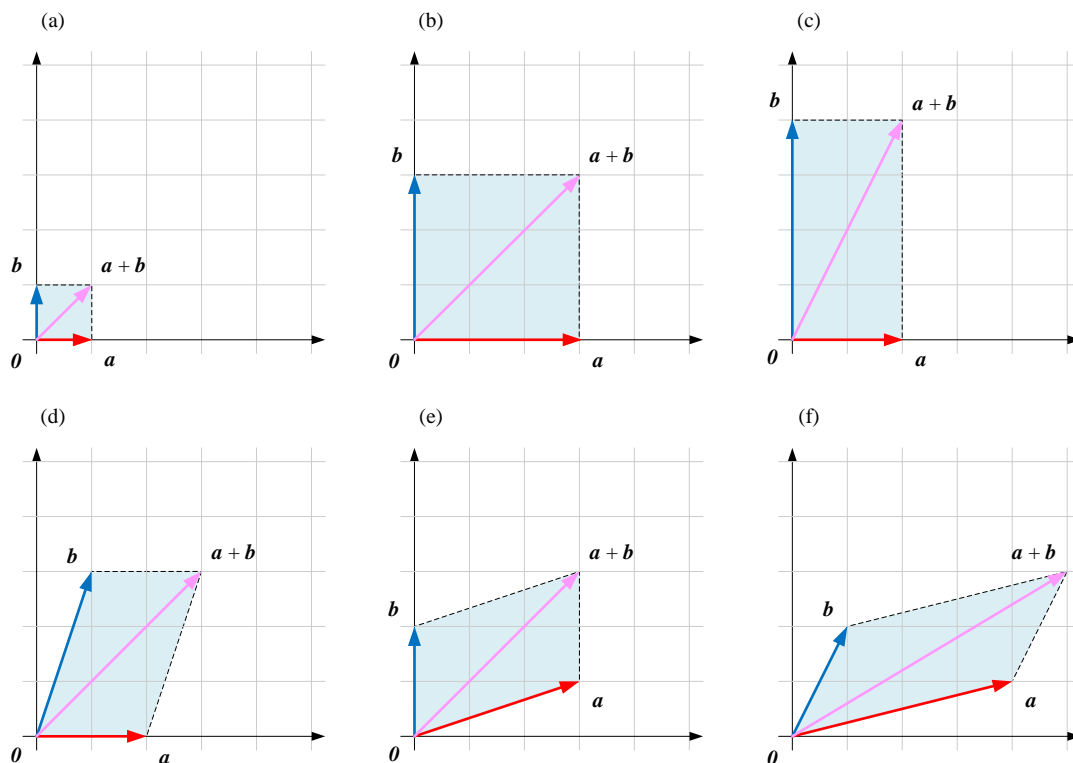


图 3. 平行四边形法则的几个例子

特别地，如图 4 所示，如果非零向量 a 、 b 共线 (collinear)，即它们位于同一条直线上， a 、 b 方向同向或反向。两者之和 $a + b$ 也在这条直线上，无法构成平行四边形。

此外，我们一再强调非零向量的情况。如果向量 a 、 b 中有任何一个是零向量 0 ，那么它们的和仍然落在非零向量所在的直线上，无法构成平行四边形。

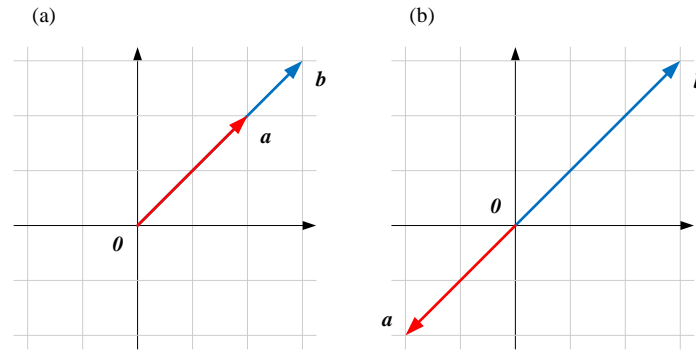


图 4. a 、 b 共线

代码 2 用平行四边形法则可视化向量加法。下面让我们聊聊其中关键词句。

a 导入了两个工具，`numpy` 和 `matplotlib.pyplot`，其中 `numpy` 主要用于处理数组和向量计算，而 `matplotlib.pyplot` 用于绘制图形。

b 用 `numpy.array()` 函数定义了两个向量 `a_vec` 和 `b_vec`。

c 计算了向量 `a_vec` 和 `b_vec` 的和，存储在 `a_plus_b` 变量中。这个计算利用了 `numpy` 数组的逐分量相加特性，即 `a_vec + b_vec` 会自动对两个数组的对应分量执行加法。

d 调用 `plt.figure(figsize=(6,6))`，这个函数用于创建一个新的绘图窗口，并设定图的大小。参数 `figsize=(6,6)` 表示图的宽度和高度都为 6 英寸。

e 使用 `plt.quiver()` 画出第一个向量 `a_vec`。

`plt.quiver(x, y, u, v, ...)` 这个函数用于绘制向量，参数 `(x, y)` 指定向量的起点，这里是 `(0,0)`，表示从原点开始。

参数 `(u, v)` 代表向量的方向和长度，即 `a_vec[0]` 和 `a_vec[1]`，即向量的 `x` 和 `y` 分量。

`angles='xy'` 表示按照标准二维坐标系绘制。

`scale_units='xy'` 让向量的长度按照坐标轴的刻度进行缩放。

`scale=1` 让向量保持实际大小，不被缩放。

`color='r'` 让向量变成红色，`label='a'` 让它在图例中标记为 `a`。

f 绘制向量 `b_vec`，语句和 **e** 几乎完全一致。

g 画出 `a_plus_b` 向量，即 `a_vec + b_vec` 的结果向量。

为了帮助观察向量相加的几何特性，**h** 使用 `plt.plot()` 画出了一个虚线的平行四边形辅助线。

`plt.plot(x_values, y_values, linestyle)` 用于绘制一条连接 `(x_values[0], y_values[0])` 和 `(x_values[1], y_values[1])` 的线。

第一条辅助线的 `x` 坐标是 `[b_vec[0], a_plus_b[0]]`，即 `[1,5]`，`y` 坐标是 `[b_vec[1], a_plus_b[1]]`，即 `[2,3]`，这条线连接 `b_vec` 的终点和 `a_plus_b` 的终点。

`'k--'` 表示黑色虚线。

i 绘制另外一条平行四边形辅助线。

j 中这些代码用来装饰图像。

`plt.gca().set_aspect('equal', adjustable='box')` 让坐标轴比例保持一致，确保 x 和 y 方向的缩放比例相同，否则图形可能会被拉伸。

`plt.xlim(0, 5)` 设置 x 轴的范围为 [0,5]，`plt.ylim(0, 5)` 设置 y 轴的范围为 [0,5]，这样整个图形都能完整显示在这个范围内。

`plt.axhline(0, color='black', linewidth=0.5)` 和 `plt.axvline(0, color='black', linewidth=0.5)` 画出了 x 轴和 y 轴的基准线，颜色为黑色，线宽 0.5。

`plt.grid(color='gray', alpha=0.8, linestyle='-', linewidth=0.25)` 让背景显示灰色网格线，使图形更加清晰，`alpha=0.8` 让网格线有一定透明度，`linestyle='-'` 让它是实线，`linewidth=0.25` 让它比较细。

最后，`plt.legend()` 让图例显示出来，它会自动使用 `plt.quiver()` 中 `label` 指定的名称，帮助区分不同的向量。

代码 2. 平行四边形法则 |  LA_01_03_02.ipynb

```

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

## 定义向量
b a_vec = np.array([4, 1])
b b_vec = np.array([1, 2])

## 计算向量和
c a_plus_b = a_vec + b_vec

## 可视化
d plt.figure(figsize=(6,6))

# 绘制向量 a
e plt.quiver(0, 0, a_vec[0], a_vec[1],
            angles='xy', scale_units='xy',
            scale=1, color='r', label="a")

# 绘制向量 b
f plt.quiver(0, 0, b_vec[0], b_vec[1],
            angles='xy', scale_units='xy',
            scale=1, color='b', label="a")

# 绘制向量 a + b
g plt.quiver(0, 0, a_plus_b[0], a_plus_b[1],
            angles='xy', scale_units='xy',
            scale=1, color='pink', label="a + b")

# 画出平行四边形的另外两条边
h plt.plot([b_vec[0], a_plus_b[0]],
           [b_vec[1], a_plus_b[1]], 'k--')

i plt.plot([a_vec[0], a_plus_b[0]],
           [a_vec[1], a_plus_b[1]], 'k--')

# 装饰
j plt.gca().set_aspect('equal', adjustable='box')
plt.xlim(0, 5)
plt.ylim(0, 5)
plt.axhline(0, color='black', linewidth=0.5)
plt.axvline(0, color='black', linewidth=0.5)
plt.grid(color='gray', alpha=0.8, linestyle='-', linewidth=0.25)
plt.legend()

```

RGB 颜色混合中的平行四边形法则

在 RGB 色彩空间中，我们可以通过平行四边形法则直观地理解图 1 中三个向量加法。

如图 5 (a) 所示，我们有红色向量 e_1 、绿色向量 e_2 ，它们的起点在原点对齐。

根据平行四边形法则，我们可以将红色向量 e_1 、绿色向量 e_2 作为平行四边形的两条邻边，然后从它们的起点出发，画出平行四边形的另外两条边。平行四边形的对角线从起点指向加法结果的方向，我们得到黄色向量 $e_1 + e_2$ 。

类似地，我们可以通过平行四边形法则计算得到品红、青色。

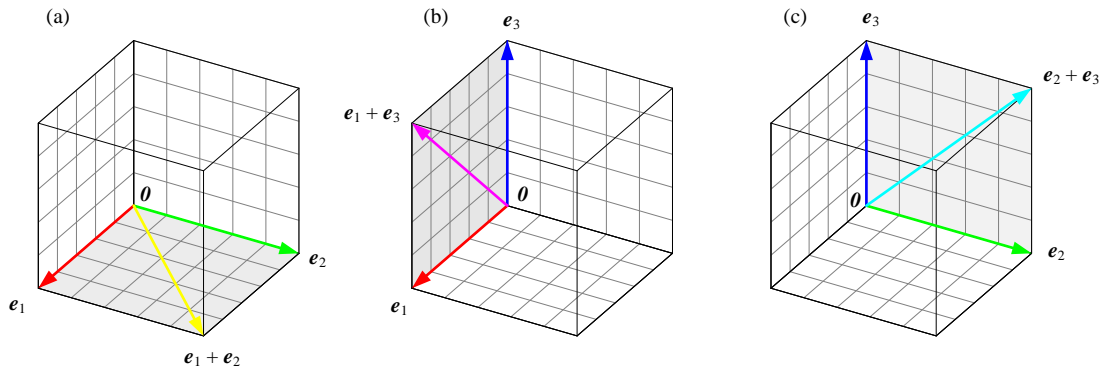


图 5. 平行四边形法则：混合两种基本色

零向量 $\mathbf{0}$ 是向量**加法单位元** (additive identity)，满足

$$\mathbf{a} + \mathbf{0} = \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_n \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} = \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_n \end{bmatrix} \quad (5)$$

加法单位元是指与任何向量相加后不改变该向量的特殊向量。

上式表达的是，任意向量加上零向量 $\mathbf{0}$ ，结果仍然是它本身。

几何上，零向量 $\mathbf{0}$ 可以被看作加法的起点，意味着所有向量的加法都可以视为从原点 $\mathbf{0}$ 出发，沿着不同的方向依次叠加。

例如，在 RGB 色彩空间中，零向量 $\mathbf{0}$ 对应于黑色 (即没有任何光)。如果从黑色 $\mathbf{0}$ 开始，依次加上红色向量 e_1 、绿色向量 e_2 ，那么最终可以抵达黄色向量。

这种理解方式直观地说明了加法是一个从原点不断推进的过程，而零向量则提供了这一过程的初始位置。

平行六面体

三个向量的加法可以通过几何方式理解为**平行六面体** (parallelepiped) 的构造。几何上，平行六面体是一个由六个平行四边形构成的立体结构。

假设有三个向量，每个向量都从原点出发，我们可以先计算前两个向量的和，并在空间中找到它们的合成向量。然后再将第三个向量加到这个结果上，最终的终点就代表三个向量相加的结果。几何上，在三维空间里会形成一个平行六面体。

图 6 所示三个向量相加的不同情形，每个子图中三个向量关系略有差别。

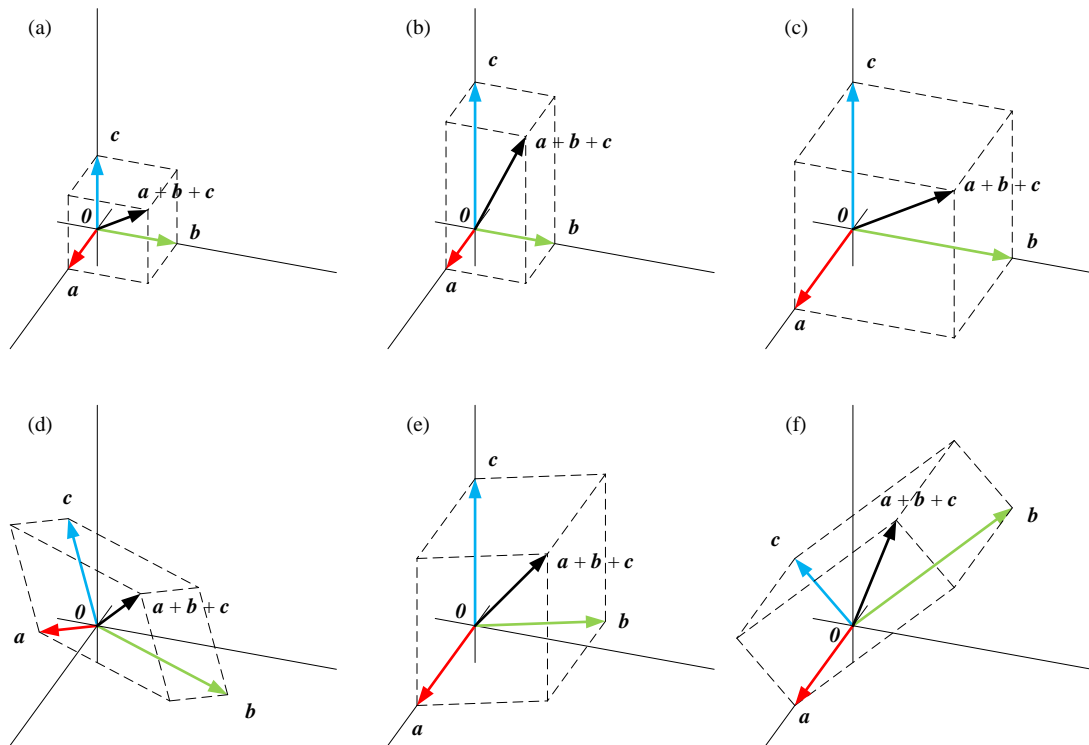


图 6. 用平行六面体理解三个（不共面）向量相加

当然也有例外的情况。如图 7 (a) 所示，如果三个向量恰好位于同一个平面上，即**共面** (coplanar)，它们的合成向量所形成的几何结构会退化为一个平行四边形，而不会真正形成三维的平行六面体。

特别地，如图 7 (b) 所示，如果三个向量首尾相接围成一个封闭的三角形，那么它们的合成向量将恰好为零向量 $\mathbf{0}$ ；这意味着从起点 $\mathbf{0}$ 出发，按照向量的顺序移动，最终会回到起点 $\mathbf{0}$ 。

这实际上体现的是三角形法则，马上要讲到的话题。

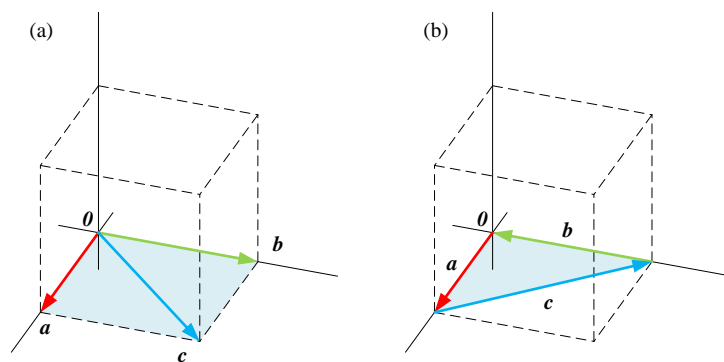


图 7. 三个向量共面

RGB 颜色混合中的平行六面体法则

回到 RGB 颜色空间。如图 8 所示，红色向量 \mathbf{e}_1 、绿色向量 \mathbf{e}_2 、蓝色向量 \mathbf{e}_3 从零向量 $\mathbf{0}$ 出发，分别对应平行六面体的三条相邻边。

这三个向量之和 $e_1 + e_2 + e_3$ 对应于该平行六面体的对角线，该对角线的起点与三个向量的起点相同，而终点则是平行六面体的对角顶点。值得注意的是，图 8 这个平行六面体是单位正方体。

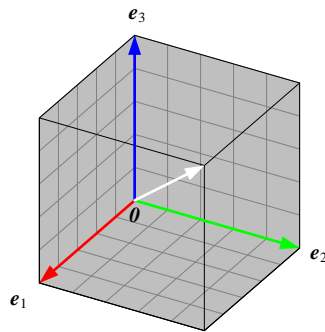


图 8. 平行六面体（单位立方体）计算红色向量 e_1 、绿色向量 e_2 、蓝色向量 e_3 三者之和

两个平行四边形

对于三个不共面向量加法的情况，平行六面体法则实际上包含两个平行四边形法则。

具体来说，首先使用平行四边形法则将两个向量相加，得到一个新的向量；然后再以该新向量和第三个向量为边，重复使用平行四边形法则，最终得到三个向量的和。这样，平行六面体法则实际上是平行四边形法则的两次迭代应用。

如图 9 所示，红色向量 e_1 、绿色向量 e_2 、蓝色向量 e_3 三者之和可以通过两个平行四边形法则来拆解。

首先，使用红色向量 e_1 、绿色向量 e_2 构造第一个平行四边形，它的对角线即为它们的和，为黄色向量 $e_1 + e_2$ 。

接着，以黄色向量 $e_1 + e_2$ 和蓝色向量 e_3 为边，再次应用平行四边形法则，构造第二个平行四边形。这次的对角线就是三个向量的总和，即白色向量 $(e_1 + e_2) + e_3$ 。

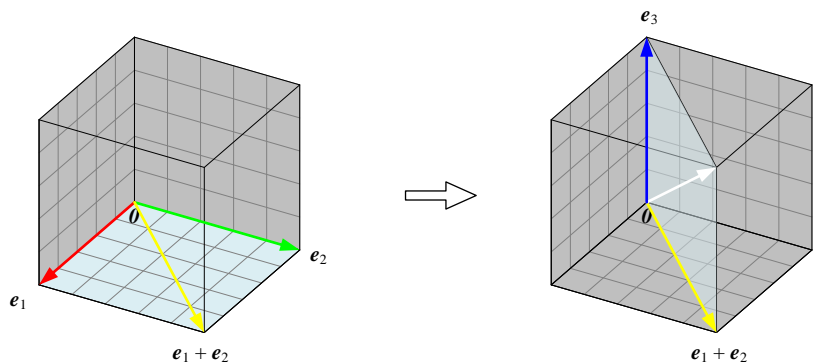


图 9. 将平行六面体拆解成两个平行四边形， $(e_1 + e_2) + e_3$

几何视角：三角形法则

三角形法则 (triangle law of vector addition) 是一种几何方法，用于计算两个向量的加法。它将第二个向量的起点与第一个向量的终点相连，从第一个向量的起点指向第二个向量的终点的有向线段就是向量加法的结果。

图 3 中用平行四边形法则完成的加法计算也可以用三角形法则完成，如图 10 所示。

和前文一样，当向量 a 、 b 共线时，它们位于同一直线上，无论方向相同还是相反，其和仍然落在该直线上，因此无法围成三角形。

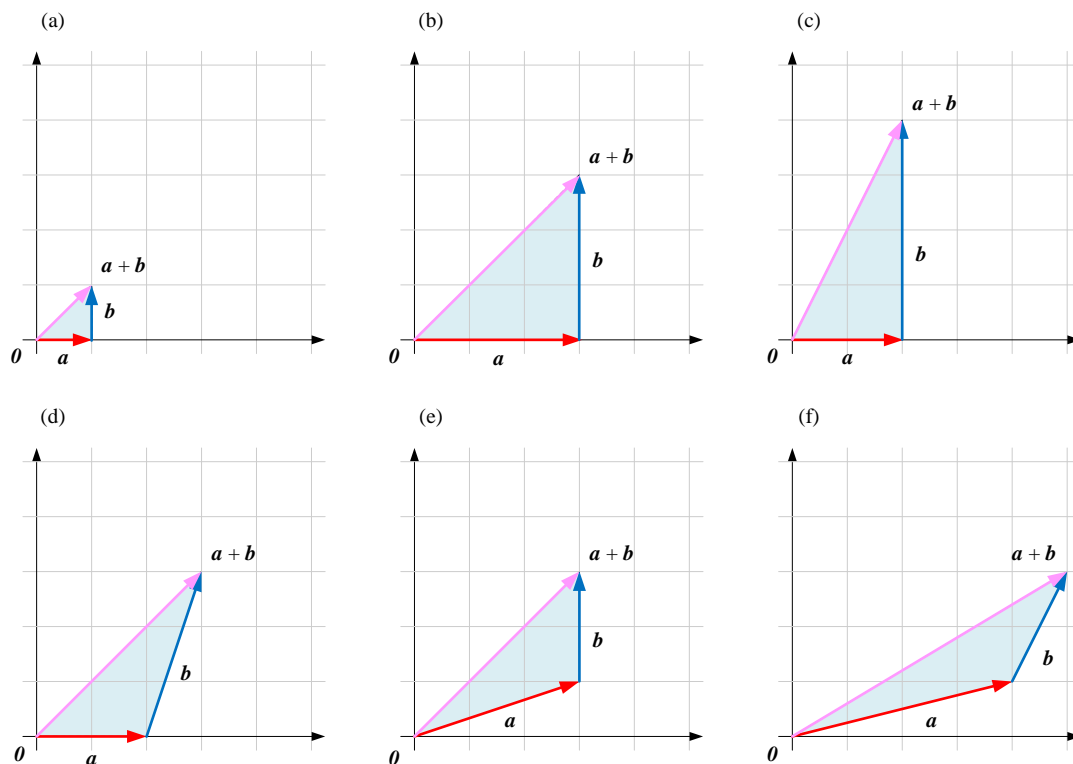


图 10. 三角形法则计算向量加法 $a + b$ 的几个例子



代码文件 LA_01_03_03.ipynb 用三角形法则可视化向量加法，代码和 LA_01_03_02.ipynb 类似，请大家自行学习。

RGB 颜色混合的三角形法则

在 RGB 色彩空间中，向量加法可以通过三角形法则直观地理解。

例如，我们希望将红色向量 e_1 、绿色向量 e_2 相加，得到黄色向量。

根据三角形法则，我们可以将绿色向量 e_2 的起点移动到红色向量 e_1 的终点，首尾相连后，从红色向量 e_1 的起点到绿色向量 e_2 的终点的有向线段便是黄色向量。

同样，我们也可以换个角度，将红色向量 e_1 的起点移动到绿色向量 e_2 的终点，最终得到的仍然是相同的黄色向量。

这实际上体现的是**向量加法的交换律** (Commutative Law of Vector Addition)，即

$$e_1 + e_2 = e_2 + e_1 \quad (6)$$

类似地，我们可以使用三角形法则来直观地理解其他颜色的向量加法。

例如，将红色向量 e_1 、蓝色向量 e_3 相加，我们可以先将蓝色向量 e_3 的起点移动到红色向量 e_1 终点，得到的有向线段即为品红向量。

换个角度，我们也可以先移动红色向量 e_1 的起点到蓝色向量 e_3 的终点，同样得到品红向量，这也体现了向量加法的交换律。

同理，当绿色向量 e_2 、蓝色向量 e_3 相加，我们可以按照相同的方法操作，无论先移动哪个向量，最终都得到青色向量。

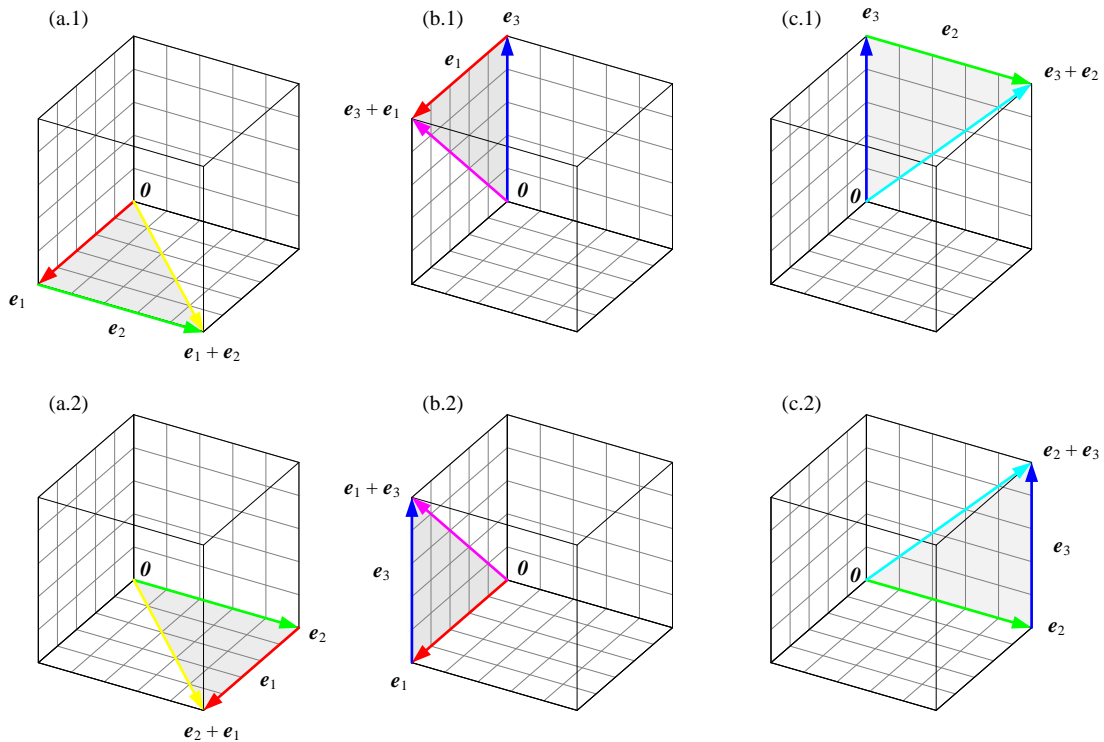


图 11. 向量加法交换律

多个向量相加：首尾顺序连接

在几何上，三角形法则也可以理解为首尾相接的过程。

假设有三个不共面向量，依次首尾相接，就是把第一个向量的终点作为第二个向量的起点，再把第二个向量的终点作为第三个向量的起点，最后形成一个从起点到最终终点的合成向量。

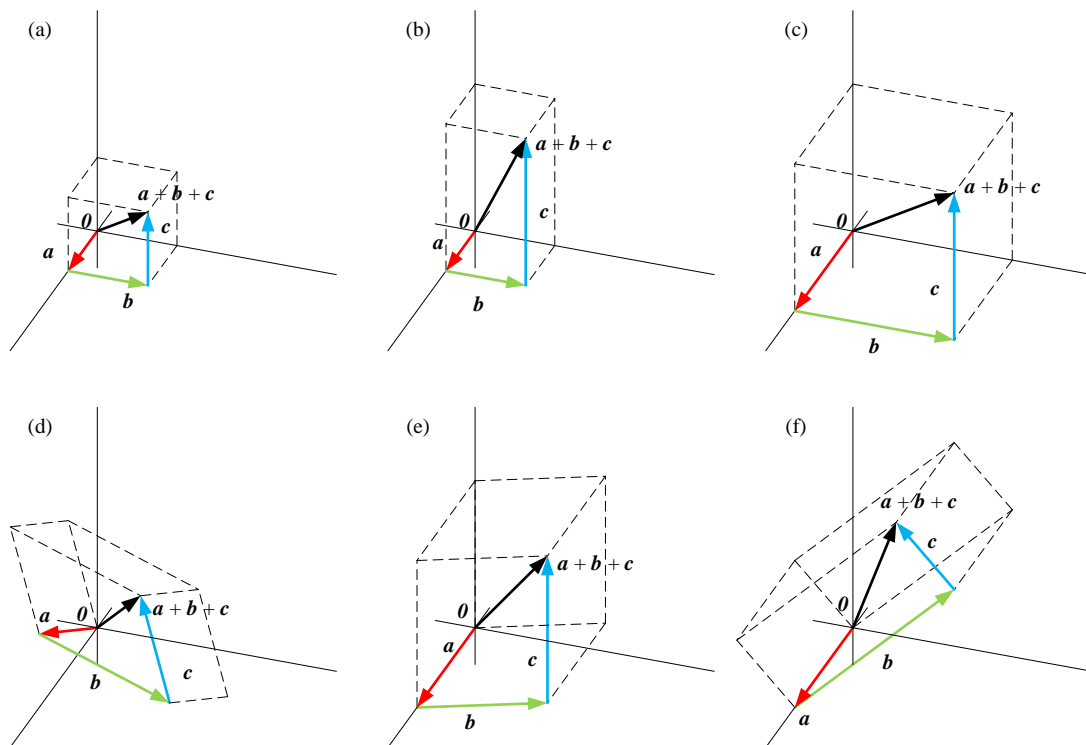
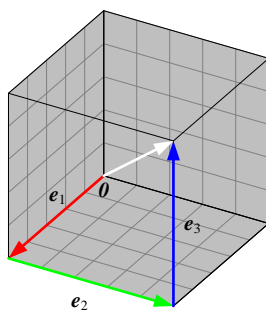


图 12. 用首尾相连的三角形法则三个向量相加

例如，在 RGB 色彩空间中，红色向量 e_1 、绿色向量 e_2 、蓝色向量 e_3 ，若按照三角形法则依次首尾相接，则最终的向量指向白色，即 $e_1 + e_2 + e_3$ 。

图 13. 首尾顺序连接计算红色向量 e_1 、绿色向量 e_2 、蓝色向量 e_3 三者之和

利用前文介绍的向量加法的结合律，向量加法 $e_1 + e_2 + e_3$ 可以写成两条路径，具体如图 14 所示。

第一条路径为 $(e_1 + e_2) + e_3$ ；先计算 $(e_1 + e_2)$ 得到黄色向量。然后再加上蓝色向量 e_3 ，最后得到白色向量。

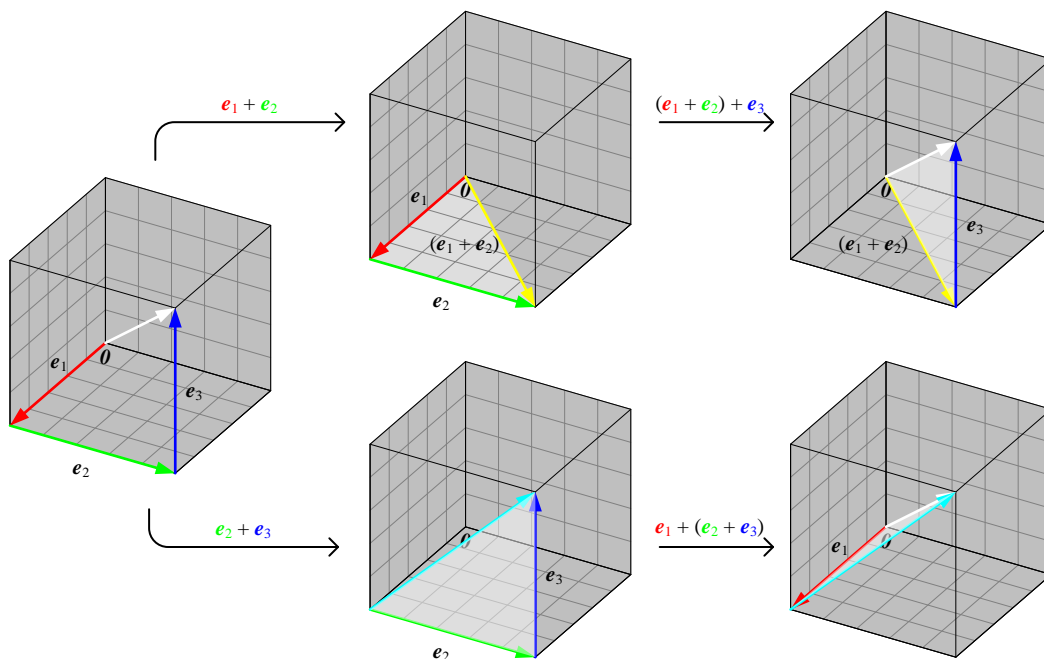
第二条路径为 $e_1 + (e_2 + e_3)$ ；先计算 $(e_2 + e_3)$ 得到青色向量。然后再加上红色向量 e_1 ，同样得到白色向量。

在几何上，这也意味着向量的合成路径可以不同，但它们的终点位置保持不变。

这种方法体现了向量加法的**向量加法的结合律** (Associative Law of Addition)，即：

$$(e_1 + e_2) + e_3 = e_1 + (e_2 + e_3)$$

(7)

图 14. $(e_1 + e_2) + e_3$ 和 $e_1 + (e_2 + e_3)$ 两条路径

将交换律、结合律结合在一起，我们可以得到获得白色向量的不同路径。

如图 15 所示，为了获得白色向量，我们可以使用 $e_3 + e_2 + e_1$ 。而这个加法也可以写成两种形式，即 $(e_3 + e_2) + e_1$ 和 $e_3 + (e_2 + e_1)$ 两条路径。

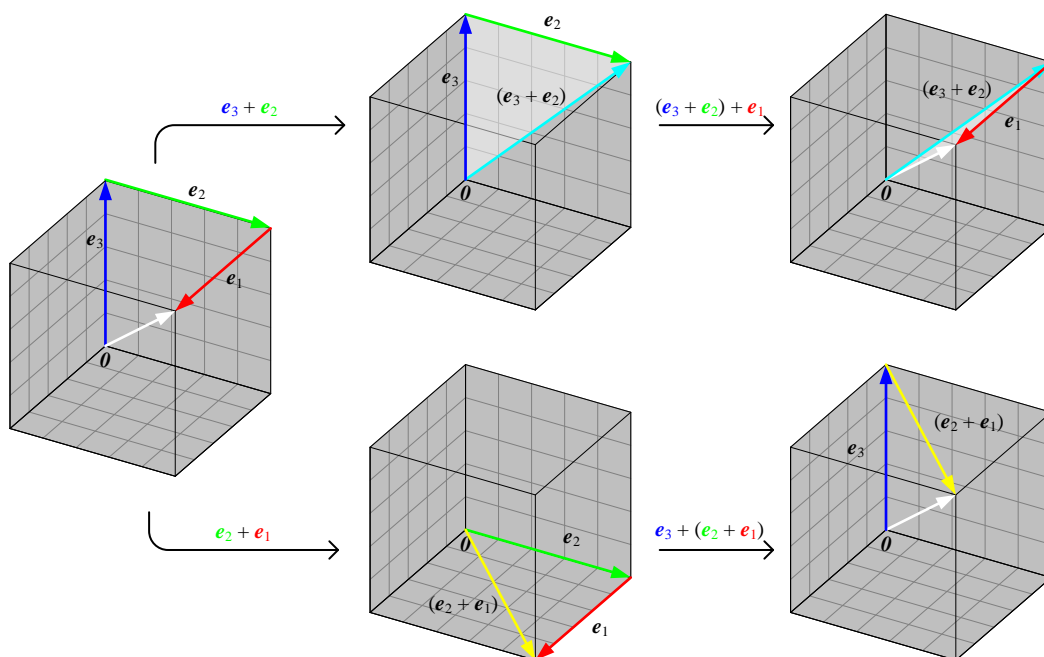


图 15. $(e_3 + e_2) + e_1$ 和 $e_3 + (e_2 + e_1)$ 两条路径

请大家思考，通过排列组合还有哪些路径可以红色向量 e_1 、绿色向量 e_2 、蓝色向量 e_3 得到白色向量。

减法

向量减法 (vector subtraction) 是对维数相同的两个向量对应分量减法运算。本节前文给定的 n 维 a 、 b 列向量之差为

$$a - b = \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_n \end{bmatrix} - \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix} = \begin{bmatrix} a_1 - b_1 \\ a_2 - b_2 \\ \vdots \\ a_n - b_n \end{bmatrix} \quad (8)$$

例如，在 RGB 色彩空间中，减去某个颜色向量意味着过滤掉这个颜色，比如下图三个例子。

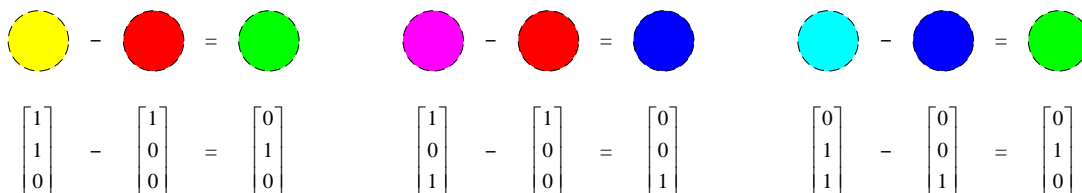


Figure 16 shows three examples of vector subtraction in RGB color space. Each example consists of a color circle, a minus sign, another color circle, an equals sign, and a third color circle. Below each example is a corresponding vector equation. The first example shows yellow minus red equals green, with the vector equation $\begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix} - \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$. The second example shows magenta minus red equals blue, with the vector equation $\begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix} - \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$. The third example shows cyan minus blue equals green, with the vector equation $\begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix} - \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$.

图 16. RGB 颜色空间的向量减法

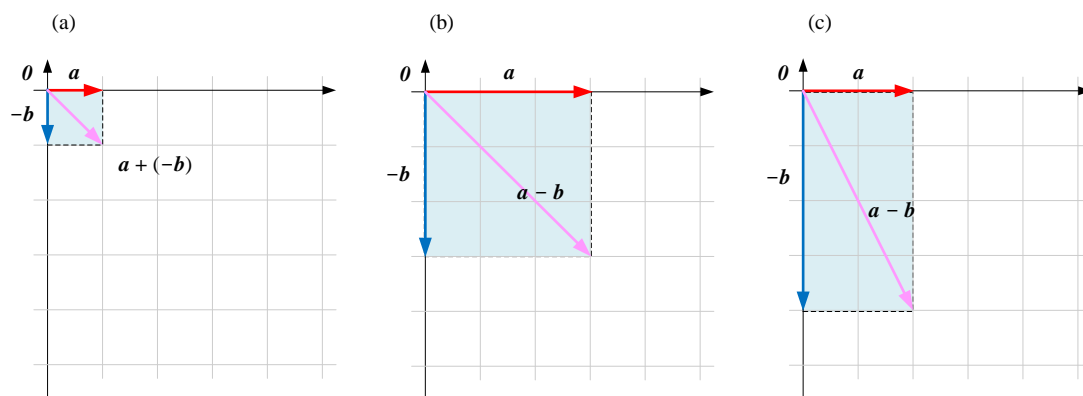
有了向量加法，向量减法的理解就变得直观且自然。上述向量减法可以写成如下加法形式

$$a - b = a + (-b) = \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_n \end{bmatrix} + \begin{bmatrix} -b_1 \\ -b_2 \\ \vdots \\ -b_n \end{bmatrix} = \begin{bmatrix} a_1 - b_1 \\ a_2 - b_2 \\ \vdots \\ a_n - b_n \end{bmatrix} \quad (9)$$

其中 $-b$ 被称为 b 的**反向量** (opposite vector)。

图 17 所示为利用平行四边形法则求解 a 和 $-b$ 向量之和。

几何上， $-b$ 与 b 具有相同的长度，但方向相反。因此，向量减法 $a - b$ 可以理解为从原点出发，先沿着 a 方向移动，然后再沿着 $-b$ 移动，最终到达 $a - b$ 的位置。

图 17. 平行四边形法则计算向量减法 $a - b$ 的几个例子

换个视角，令

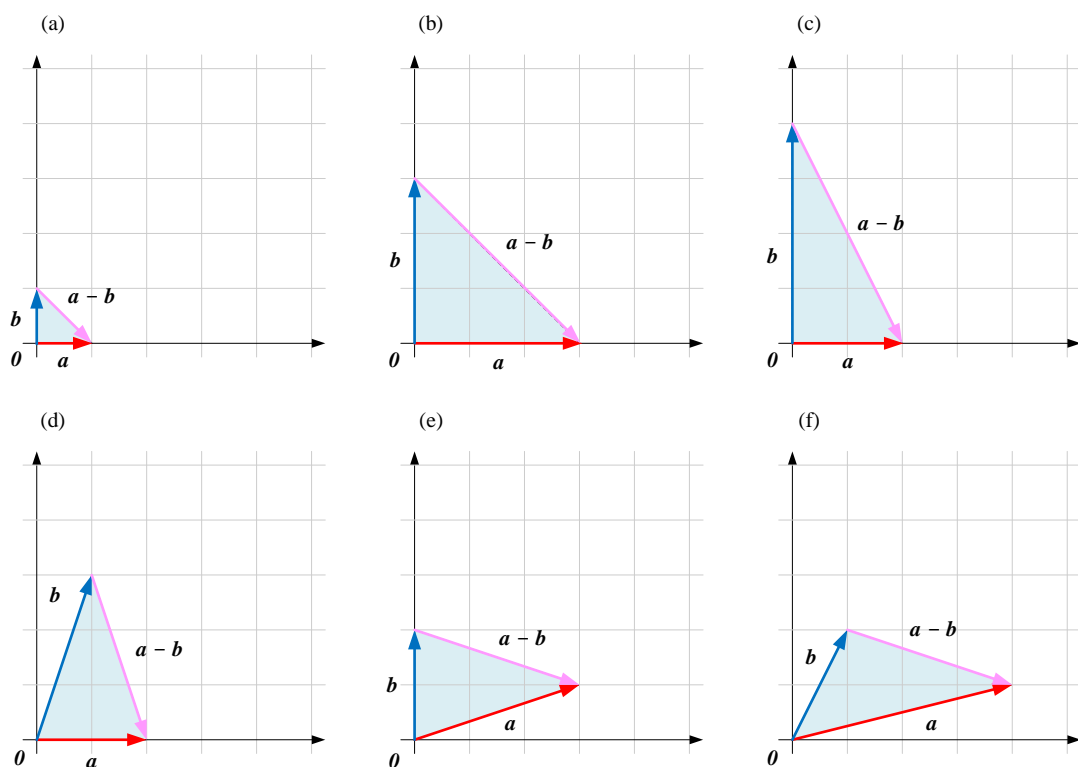
$$c = a - b \quad (10)$$

等式左右加上 b ，得到

$$\begin{aligned} b + c &= a \\ b + (a - b) &= a \end{aligned} \quad (11)$$

这意味着 b 和 $a - b$ 之和为 a 。

如图 18 所示，将 a 、 b 向量的起点对其在 0 ， $a - b$ 的结果是以 b 的终点为起点指向 a 的终点。这本质上体现的还是三角形法则。

图 18. 三角形法则计算向量减法 $a - b$ 的几个例子

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



请大家用 DeepSeek/ChatGPT 等工具完成本节如下习题。

Q1. 请用 `np.array()` 创建如下向量，并计算它们的加法。

$$\mathbf{a} = [1, 2, 3]^T, \mathbf{b} = [3, 4, 5]^T, \mathbf{c} = [5, 6, 7]^T$$

Q2. 请利用代码 2 可视化图 3 每个子图。

Q3. 请学习使用如下函数，修改代码 2，给平行四边形增加填充色。

https://matplotlib.org/stable/api/_as_gen/matplotlib.pyplot.fill.html

Q4. 请修改代码 1 完成向量减法运算 $\mathbf{a} - \mathbf{b}$ 、 $\mathbf{b} - \mathbf{a}$ 。

$$\mathbf{a} = [3, 2, 1]^T, \mathbf{b} = [5, 4, 3]^T$$

Q5. 请编写 Python 代码绘制图 8。

Q6. 请编写 Python 代码绘制图 13。

Q7. 请用平行四边形法则可视化图 3 (d)、(e)、(f) 子图中 $\mathbf{a} - \mathbf{b}$ 。

Q8. 请大家根据图 18 绘制 $\mathbf{b} - \mathbf{a}$ 。