

Notes's LR  
ML

终极笔记

# Dot product

Change basis

坐标 X 固定  
基底为 a

## 2.1.1 矩阵表示的空间映射

回顾第1章，在默认基底  $(e_1, e_2, e_3, \dots, e_n)$  所构成的  $R^n$  空间中，矩阵  $A$  与列向量  $x$  的乘法  $Ax$ ，其本质就是变换原始向量的基底。将默认基底中的各个基向量  $(e_1, e_2, e_3, \dots, e_n)$  分别对应地变换为矩阵  $A$  的各列，由矩阵  $A$  的各列充当目标向量新的“基向量”，再结合原始向量的坐标，最终得到目标向量在目标空间中的新位置。

因此可以概况地说：由于矩阵乘法的作用，原始向量的空间位置甚至其所在空间的维度和形态都发生了改变，这便是矩阵乘法的空间映射作用。

更直白地说，通过乘法运算，矩阵把向量的基底进行了变换，旧的基底  $\begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix}$  被变成了新的基底  $\begin{bmatrix} a \\ c \\ d \end{bmatrix}$ 。映射前由旧的基底分别乘以对应的坐标值  $(x, y)$  来表示其空间位置，而乘法运算之后，由于旧的基底被映射到了新的基底，那么向量自然而然应该用新的基底去分别乘以对应坐标值  $(x, y)$  来描述改变之后的空间位置： $x \begin{bmatrix} 1 \\ 0 \end{bmatrix} + y \begin{bmatrix} 0 \\ 1 \end{bmatrix} \Rightarrow x \begin{bmatrix} a \\ c \\ d \end{bmatrix} + y \begin{bmatrix} b \\ d \end{bmatrix} = \begin{bmatrix} ax+by \\ cx+dy \end{bmatrix}$ ，如图 1.9 所示。

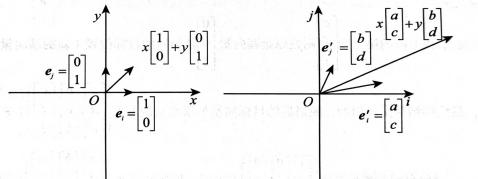


图 1.9 在矩阵的作用下，向量的基底发生了变换

因此，从这个例子中可以总结出矩阵所发挥的重要作用：在指定矩阵的乘法作用下，原始空间中的向量被映射转换到了目标空间中的新坐标，向量的空间位置由此发生了变化，甚至在映射之后，目标空间的维数相较于原始空间都有可能发生改变。那么，具体这些空间位置的改变有什么规律可言？在 1.4 节中会聚焦这一点，继续挖掘其背后更深层次的内涵。

## 1.1.7 向量间的乘法：内积和外积

向量间的乘法分为内积和外积两种形式，首先来介绍向量的内积运算。参与内积运算的两个向量必须维数相等，运算规则是先将对应位置上的元素相乘，然后合并相加，向量内积的最终运算结果是一个标量。

例如，两个  $n$  维向量  $u$  和  $v$  进行内积运算的规则为：

$$u \cdot v = \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ \vdots \\ u_n \end{bmatrix} \cdot \begin{bmatrix} v_1 \\ v_2 \\ v_3 \\ \vdots \\ v_n \end{bmatrix} = u_1 v_1 + u_2 v_2 + u_3 v_3 + \dots + u_n v_n$$

路过。。  
只在 v 投影

这个定义看上去好像没有什么特殊含义，但是内积的另一种表示方法  $u \cdot v = |u||v|\cos \theta$  所包含的物理意义就十分清晰了，它表示向量  $u$  在向量  $v$  方向上的投影长度乘向量  $v$  的模长，如图 1.3 所示。如果  $v$  是单位向量，内积就可以直接描述为向量  $u$  在向量  $v$  方向上的投影长度。

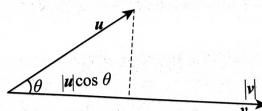


图 1.3 向量内积的几何表示

$$Ax = \begin{bmatrix} a_{11} & a_{12} & a_{13} & \dots & a_{1n} \\ a_{21} & a_{22} & a_{23} & \dots & a_{2n} \\ a_{31} & a_{32} & a_{33} & \dots & a_{3n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & a_{m3} & \dots & a_{mn} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_n \end{bmatrix} = x_1 \begin{bmatrix} a_{11} \\ a_{21} \\ a_{31} \\ \vdots \\ a_{m1} \end{bmatrix} + x_2 \begin{bmatrix} a_{12} \\ a_{22} \\ a_{32} \\ \vdots \\ a_{m2} \end{bmatrix} + x_3 \begin{bmatrix} a_{13} \\ a_{23} \\ a_{33} \\ \vdots \\ a_{m3} \end{bmatrix} + \dots + x_n \begin{bmatrix} a_{1n} \\ a_{2n} \\ a_{3n} \\ \vdots \\ a_{mn} \end{bmatrix}$$

在  $m \times n$  形状大小的矩阵  $A$  的作用下，原始的  $n$  维基向量  $\begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix}$  被映射成了新的  $m$  维基向量  $\begin{bmatrix} a_{11} \\ a_{21} \\ a_{31} \\ \vdots \\ a_{m1} \end{bmatrix}$ ，原始的  $n$  维基向量  $\begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$  被映射成了新的  $m$  维基向量  $\begin{bmatrix} a_{12} \\ a_{22} \\ a_{32} \\ \vdots \\ a_{m2} \end{bmatrix}$ ，…… 原始的  $n$  维基向量  $\begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$  被映射成了新的  $m$  维基向量  $\begin{bmatrix} a_{1n} \\ a_{2n} \\ a_{3n} \\ \vdots \\ a_{mn} \end{bmatrix}$ 。

从上面的推导结果中可以发现， $m \neq n$  这种情况最值得讨论，因为此时矩阵  $A$  是一个普通的矩阵，而不是之前所举的特殊方阵。在这种最一般的情况下，映射前后，列向量  $x$  的基向量维数都发生了变化：原始的  $n$  维列向量  $x$  被变成了  $n$  个  $m$  维列向量线性组合的形式，其最终的运算结果是一个  $m$  维的列向量。

由此可以看出，映射后的向量维数和原始向量维数的关系取决于矩阵维数  $m$  和  $n$  的关系：如果  $m > n$ ，那么映射后的目标向量维数就大于原始向量的维数；如果  $m < n$ ，那么目标向量的维数就小于原始向量的维数；如果  $m = n$ ，那么就是前面所举的方阵那种特殊情况了。

Dot product

$AX = B$

# 空间映射的实质

## 2.1.1 矩阵表示的空间映射

回顾第1章，在默认基底  $(e_1, e_2, e_3, \dots, e_n)$  所构成的  $R^n$  空间中，矩阵  $A$  与列向量  $x$  的乘法  $Ax$ ，其本质就是变换原始向量的基底。将默认基底中的各个基向量  $(e_1, e_2, e_3, \dots, e_n)$  分别对应地变换为矩阵  $A$  的各个列，由矩阵  $A$  的各列充当目标向量新的“基向量”，再结合原始向量的坐标，最终得到目标向量在目标空间中的新位置。

因此可以概况地说：由于矩阵乘法的作用，原始向量的空间位置甚至其所在空间的维度和形态都发生了改变，这便是矩阵乘法的空间映射作用。

在前面的内容中，反复讲了这样一个结论：矩阵的本质就是映射。对于一个  $m \times n$  的矩阵  $A$ ，矩阵乘法  $y = Ax$  的作用就是将向量从  $n$  维原始空间中的  $x$  坐标位置，映射到  $m$  维目标空间中的  $y$  坐标位置，这是正向映射的过程。

## 2.2.3 “矮胖”矩阵压缩空间：不存在逆映射

$$m \text{ 行 } n \text{ 列的矩阵 } A_{m \times n} \text{ 将向量 } x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_n \end{bmatrix} \text{ 从原始空间映射到目标空间中的 } y = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \\ y_m \end{bmatrix}, \text{ 其中, } n > m, \text{ 即映}$$

射的矩阵  $A_{m \times n}$  是一个列数大于行数的“矮胖”矩阵。

下面举一个实例来说明问题：假设矩阵  $A$  是一个  $2 \times 3$  的“矮胖”矩阵，那么在前文中讨论过，如果矩阵  $A$  的 3 个列向量共面但不共线，则该矩阵能将一个  $R^3$  空间压缩成一个二维平面；如果这 3 个列向量都满足共线的条件，则经过矩阵的映射作用最终将  $R^3$  空间压缩成一条直线。

这个映射过程的本质是将向量  $x$  所在的三维空间  $R^3$ ，映射到向量  $y$  所在的二维空间（或者甚至是维的空间）中，对应于压缩扁平化的操作。仔细想想生活中的“压扁了”这个概念：一个纸盒

子被一巴掌拍成了一张纸，也就是说，在这个过程中会有多个向量  $x$  会被转移到同一个向量  $y$  上去。

下面用一个实例来具体描述一下压缩这个过程，还是用 2.1 节中的矩阵： $A = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix}$ 。在这里，

我们聚焦的是映射后目标空间中的零向量  $y = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$ ，看看原始空间中会有多少个向量被映射到那里去。

换句话说，我们的目标就是去寻找所有满足  $Ax = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$  等式成立的向量  $x$ 。

展开得到：

$$\begin{cases} x_1 + x_2 = 0 \\ x_1 + x_3 = 0 \end{cases}$$

对等式简单进行处理后发现：在原始空间  $R^3$  中，各成分满足  $x_2 = x_3 = -x_1$  的向量（记作  $c \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$ ，

$c$  取任意实数），经过矩阵  $A$  的映射作用，都能映射到目标空间中的零向量  $\begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$  上。从通式  $c \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$  中

可以显而易见的是，原始空间中满足这个条件的向量有无穷多个。

如图 2.6 所示，所有满足条件的向量都分布在一条直线上。

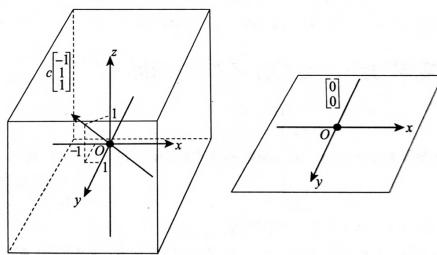


图 2.6 分布在一条直线上的向量被压缩为一个点

在图 2.6 中揭示了一个现象：已知映射后在二维目标空间中的向量  $y$ ，想要寻找原始空间中的向量  $x$  在哪里是无法判断出来的。

在图 2.6 中可以看出，目标空间中的零向量，其对应在原始空间中的向量  $x$  可以是直线上的任

## 2.2.5 “高瘦”矩阵不存在逆映射：目标空间无法全覆盖

“矮胖”矩阵在映射的过程中，压缩了空间维度，丢失了信息，因此这个映射可谓是“再也回不去了”。那么，如果映射的结果向量  $y$  的维度大于原始空间向量  $x$  的维度，如在矩阵  $\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$  的作用下，将一个二维向量映射成了一个三维向量，又是一个什么样的情况呢？

从表面上来看，似乎没有压缩空间，反而是把一个二维向量扩充成了三维向量，看上去信息量应该是更大了。从直观上来看，目标空间中的向量肯定能够逆映射回原始空间，并找到出发点。实际上这一定能做到吗？并不能，因为我们将二维空间映射到了一个三维空间中，而仅凭二维空间所携带的信息量就想把三维空间全部覆盖，那是不可能的，最终的映射效果如图 2.7 所示。

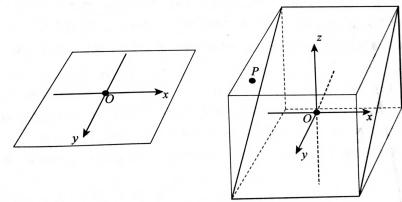


图 2.7 经过“高瘦”矩阵的映射，无法覆盖整个目标空间

从表面的维度数字上看，之前的一个二维空间被映射到了三维空间中，但实际上我们发现，映射后的最终结果实质上是一个“穿过滤点”并且“倾斜”的面搭在三维空间中的一个面上，它由三维向量构成，但是它是二维的平面。那么，可以很明显看出，位于这个二维平面外的任意一点，都无法找到原始空间中对应的出发点。因此“高瘦”矩阵的逆映射自然也是不存在的。

# linear algebra $ax = b$

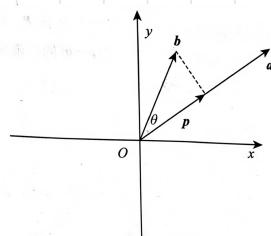


图 3.1 寻找距离直线最近的距离

从图 3.1 中可以发现，向量  $b$  和向量  $a$  的夹角是  $\theta$ ，因此，通过点  $b$  到直线的最近距离可以表示为  $|b| \sin \theta$ 。还需要注意到向量  $p$ ，它是从原点出发到垂足的向量，是向量  $b$  在向量  $a$  上的投影。而对于向量  $e = b - p$ ，我们称为误差向量，它的长度就是我们要寻找的最近距离。

当然，也可以用向量点积的方式得到投影向量  $p$  的长度，然后再进一步通过代数运算的方法得到向量  $p$  的坐标表示。但是，这里谈的是直线的情况，一维空间的计算是非常简单的，如果进一步对问题进行拓展，去讨论向量在二维平面、三维空间甚至是更高维空间中的投影问题，那么依靠这种方法就不太合适了。所以，在这里需要借助矩阵工具来描述这个投影的过程，即需要一种通用的计算方法。

首先需要分析如何利用矩阵描述向量  $b$  向一条直线上进行投影的过程。如果将向量  $a$  作为这条直线的基向量，那么向量  $p$  就可以用向量  $a$  来进行表示，记作  $p = \hat{x}a$ （ $\hat{x}$  是一个标量），接下来的目标就是去求取标量系数  $\hat{x}$ 、投影向量  $p$  和投影矩阵  $P$ 。

在这里仍然需要牢牢地把握住一个核心要点，那就是误差向量  $e$  和基向量  $a$  之间的垂直关系，

因此就有等式  $a \cdot e = 0$  成立，再进一步对其进行展开，得到下面的等式关系。

$$a \cdot e = 0 \Rightarrow a \cdot (b - p) = 0 \Rightarrow a \cdot (b - \hat{x}a) = 0$$

此时，再做一点简单的运算：

$$\begin{bmatrix} 1 & 1 & 1 \end{bmatrix} \vec{b} \\ a \cdot b - \hat{x}a \cdot a = 0 \Rightarrow \hat{x} = \frac{a \cdot b}{a \cdot a}$$

我们知道  $\vec{a} \cdot \vec{b} = \vec{a}^T \vec{b}$ ，因此最终就求出了标量系数  $\hat{x}$  的表达式为  $\hat{x} = \frac{\vec{a}^T \vec{b}}{\vec{a} \cdot \vec{a}}$ 。

投影向量也能顺势得出： $p = \hat{x}a = \frac{\vec{a}^T \vec{b}}{\vec{a} \cdot \vec{a}} a$ 。

最后，求将向量  $b$  变换到其投影向量  $p$  的投影变换矩阵  $P$ 。这里有一个小技巧，由于  $p = \hat{x}a$  是标量与向量的乘法，因此交换一下位置后的  $p = a\hat{x}$  同样能够成立，于是就有了等式：

$$p = a\hat{x} = a \frac{\vec{a}^T \vec{b}}{\vec{a} \cdot \vec{a}} = \frac{\vec{a}\vec{a}^T}{\vec{a} \cdot \vec{a}} \vec{b}$$

由此得到了投影矩阵  $P = \frac{\vec{a}\vec{a}^T}{\vec{a} \cdot \vec{a}}$ 。

最终求得了这 3 个值：

$$\begin{cases} \hat{x} = \frac{\vec{a}^T \vec{b}}{\vec{a} \cdot \vec{a}} \\ p = \frac{\vec{a}\vec{a}^T}{\vec{a} \cdot \vec{a}} a \\ P = \frac{\vec{a}\vec{a}^T}{\vec{a} \cdot \vec{a}} \end{cases}$$

$\vec{a}x = \vec{b}$  P projection

$$\vec{x} = \frac{\vec{a}^T \vec{b}}{\vec{a} \cdot \vec{a}} a$$

$$P = \frac{\vec{a}\vec{a}^T}{\vec{a} \cdot \vec{a}}$$

projection matrix of matrix  $a$

$P \vec{b} \rightarrow$  given a vector  $\vec{b}$ , find its projection in matrix  $a$ 's space

$\vec{x}a = \vec{a}\vec{x}$   
scalar, here

## 举个例子

下面举一个简单的一维直线投影例子来实际运用一下上面的公式。已知向量  $a = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}$ ，利用公式来寻找向量  $b = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$  在该向量上的投影向量  $p$ ，以及任意向量在向量  $a$  上的投影矩阵  $P$ 。

首先，求一下投影向量  $p$ ，按照公式，整个计算过程非常容易：

$$\vec{x} = \frac{\vec{a}^T \vec{b}}{\vec{a} \cdot \vec{a}} = \frac{\begin{bmatrix} 1 & 2 & 3 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}}{\begin{bmatrix} 1 & 2 & 3 \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}} = \frac{1}{14} \begin{bmatrix} 1 & 2 & 3 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} = \frac{1}{14} \begin{bmatrix} 6 \\ 12 \\ 9 \end{bmatrix} = \begin{bmatrix} \frac{3}{7} \\ \frac{6}{7} \\ \frac{3}{7} \end{bmatrix}$$

$\vec{a} \cdot \vec{b} = |a||b|\cos\theta$   
 $\vec{a} \cdot \vec{a} = |a|^2$   
 $b$  in  $a$  projection

因此求取了向量  $p$ ：

$$p = \vec{x}a = \frac{3}{7} \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix} \rightarrow \text{坐标 base} \rightarrow 0$$

然后，求在向量  $a$  上投影的矩阵  $P$ ：

直接代入公式可得：

$$P = \frac{\vec{a}\vec{a}^T}{\vec{a} \cdot \vec{a}} = \frac{\begin{bmatrix} 1 & 2 & 3 \end{bmatrix} \begin{bmatrix} 1 & 2 & 3 \end{bmatrix}}{\begin{bmatrix} 1 & 2 & 3 \end{bmatrix} \begin{bmatrix} 1 & 2 & 3 \end{bmatrix}} = \frac{1}{14} \begin{bmatrix} 1 & 2 & 3 \end{bmatrix} \begin{bmatrix} 1 & 2 & 3 \end{bmatrix}$$

最后，对前后两次运算的结果进行检验：

$$p = Pb = \frac{1}{14} \begin{bmatrix} 1 & 2 & 3 \end{bmatrix} \begin{bmatrix} 1 & 4 & 6 \\ 3 & 6 & 9 \end{bmatrix} = \frac{1}{14} \begin{bmatrix} 6 \\ 12 \\ 9 \end{bmatrix} = \begin{bmatrix} \frac{3}{7} \\ \frac{6}{7} \\ \frac{3}{7} \end{bmatrix}$$

经过核算可以发现，前后两次计算得到的向量  $p$  结果一致，我们的计算过程是准确无误的。



$$a \vec{x} = \vec{b}$$

$$\begin{bmatrix} | & | & | & | \\ m \times n & & & \end{bmatrix} \begin{bmatrix} ] \\ n \times 1 \end{bmatrix} = \begin{bmatrix} ] \\ m \times 1 \end{bmatrix}$$

The coordinates of  $\vec{x}$  under base  $a$  new vector  $\vec{b}$  in  $\mathbb{R}^m$   $\rightarrow \vec{x}$  看作坐标 or vector

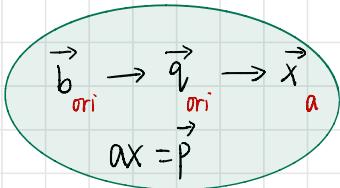
$$a^{m \times n} \quad a^T b = \begin{bmatrix} a^T b \\ \vdots \\ a^T b \end{bmatrix} \quad \vec{b} \text{ 在 } a \text{ 列 space 的坐标 projection}$$

$$a^T a \quad \vec{b} \text{ 列 vector 的长度}$$

$$a a^T \quad \vec{b} \text{ 行 vector 的长度}$$

Find  $\vec{b}$  在 basis  $a$  的坐标  $x$

- ① Projection matrix of  $a$   $P$
- ② Project  $\vec{b}$  to basis  $a$  (column space)



$$\vec{P}_{ori} = P \vec{b} = \vec{a}\vec{x}$$

$$\vec{x}_a = \frac{\vec{a}^T \vec{b}}{\vec{a}^T \vec{a}} \vec{a} \cdot \vec{b}$$

$$\vec{P}_{ori} = \vec{a}\vec{x}$$

### 4.1.1 重要回顾：坐标值取决于基底

在本节的开头，需要再次提及一个之前反复强调的核心概念：对于一个指定的向量而言，它在空间中的位置是绝对的，而它的坐标值却是相对的。向量坐标的取值依托于空间中所选取的基底。

更直白地说，对于同一个向量，如果选取的基底不同，其所对应的坐标值就不同。

下面从图 4.1 中回顾一下上述概念。

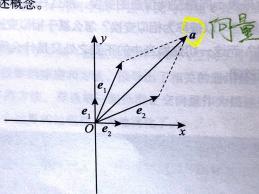


图 4.1 用不同的基描述同一个向量

从图 4.1 中可以看到，向量  $a$  在空间中的位置是固定的，如果使用一组默认基底  $(e_1, e_2)$ ，即用  $\begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix}$  来对它进行表示，则向量  $a$  可以表示为  $a = 3\begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix}$ ，那么意味着在基底  $(e_1, e_2)$  下，向量  $a$  的坐标为  $\begin{bmatrix} 3 \\ 0 \\ 3 \end{bmatrix}$ 。

#### 机器学习线性代数基础 Python 语言描述

线性变换 左乘矩阵

但是，如果采用新的一组基底，即在基底  $\begin{bmatrix} 1 & 1 \\ 0 & 2 \end{bmatrix}$  对上述过程进行描述，则向量坐标的转换就

变成了  $\begin{bmatrix} 1 \\ 1 \end{bmatrix} \rightarrow \begin{bmatrix} -4 \\ 5 \end{bmatrix}$ （计算过程可以参考前面的讲解介绍）。那么不难发现，在默认基底  $\begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix}$  下的变

换矩阵  $\begin{bmatrix} 1 & -2 \\ 1 & 1 \end{bmatrix}$  显然就无法表示新基底下的坐标转换了，因为通过计算可以观察出： $\begin{bmatrix} 1 & -2 \\ 1 & 1 \end{bmatrix} \neq \begin{bmatrix} -4 \\ 5 \end{bmatrix}$ 。

通过这个实例可以发现，对于同一个向量的空间位置改变，由于我们所选取的基底不同，因此表征其线性变换的矩阵就不同。

下面再举一个生活中的小例子。假如，一辆车从点  $A$  向点  $B$  行驶，如果此时我在点  $A$  面向车尾站立，在我看车，车是离我远去的，越来越远；而如果你在这时是在点  $B$  面向车头站立，在你看车，车是越来越近的。其实，车还是那辆车，也还是那样行驶，只是你和我所站的位置不同、视角不同，因此感受到的运动状态就是不同的。

### 4.1.4 利用基底变换推导相似矩阵间的关系式

在基底  $(e_1, e_2)$  下，坐标为  $x$  的向量通过矩阵  $A$  完成了线性变换的过程，线性变换后的向量坐标为  $x'$ ，也可以通过矩阵  $P$ ，将向量变换到新基底  $(e'_1, e'_2)$  下的坐标表示，即用新的基底下的坐标来表示向量，记作  $Px$ 。

这时在新的基底下，用来表示上面同一个空间变换的是另一个矩阵  $B$ ，即在新基底  $(e'_1, e'_2)$  下变换后的目标坐标为  $BPx$ 。最终我们还是需要在原始基底的坐标系下讨论和比较问题，因此需要再次把坐标从新基底  $(e'_1, e'_2)$  变回到原始基底  $(e_1, e_2)$  下。这显然是一个逆变换过程，即通过左乘一个逆矩阵  $P^{-1}$  来完成，因此和最初直接用矩阵  $A$  进行变换可以说是殊途同归。

描述上述变换过程的矩阵是： $A = P^{-1}B P$ ，其中，矩阵  $A$  和矩阵  $B$  就是我们所说的相似矩阵，它们分别表示了同一个向量在两个不同基底  $(e_1, e_2)$  和  $(e'_1, e'_2)$  下的相似变换过程。

那么具体这个矩阵  $P$  该如何进行表示，或者说它是如何得到的？下面来分析一下这个变换过程，即向量在空间中发生一次线性变换，由原来的空间位置  $M$  变换到目标空间位置  $N$ 。本质上非常简单，如图 4.3 所示。

→ 用一个向量  $\vec{a}$

换基 ①

$$\vec{a} = [e_1, e_2] \begin{bmatrix} x \\ y \end{bmatrix}$$

$$\vec{a} = [e'_1, e'_2] \begin{bmatrix} x' \\ y' \end{bmatrix}$$

$P$ [ $\vec{a}$ ]

左乘矩阵

基不同，坐标不同

→ 对向量进行线性变换

线性变换 ②

$$A[x]_{[e_1, e_2]} = \vec{a}'_{[e'_1, e'_2]}$$

$$B[x']_{[e'_1, e'_2]} = \vec{a}'_{[e'_1, e'_2]}$$

基不同，同一变换的变换矩阵不同 (A, B)

$$\rightarrow A = P^{-1} B P$$

combination ③

A 是一个一般矩阵 (线性变换)

B 是 A 的相似矩阵  $B = P^{-1} A P$

在 4.1 节中，介绍了对角矩阵的优良性质。因此，在向量的线性变换中，如果能够利用矩阵的相似变换，将表示线性变换的矩阵转换为一个相似对角矩阵，则在新的基底表示下，线性变换的过程就能够大大简化，由原本的长度和方向均会发生变化变成仅仅在基向量方向上做长度的伸缩变换。

在 4.1 节末文留下了一个问题，即如何将一般的方阵  $A$  经过相似变换转换成一个对角矩阵  $A'$ 。我们在本节中着重解决这个问题。我们将首先介绍矩阵的特征值和特征向量的有关特性，并说明可以利用矩阵的特征向量来构造转换矩阵  $P$ ，由此将一个一般方阵成功地转换为对角矩阵。通过对本节内容的学习，读者能够明确这个问题中所蕴含的几何意义，并在此基础上深刻理解变换的思路和方法。

Find 对角矩阵  $B$ ，在  $B$  对应的基底表示下  $\begin{bmatrix} x' \\ y' \end{bmatrix}$

做变换

换回 original 基底 → 线性变换后的向量

$$AX \text{ 等同于 } B(PX) = \begin{bmatrix} \lambda & 0 \\ 0 & \lambda \end{bmatrix} \begin{bmatrix} x' \\ y' \end{bmatrix}$$

# 转换矩阵 P 与 projection matrix

新坐标  $x' = Px$

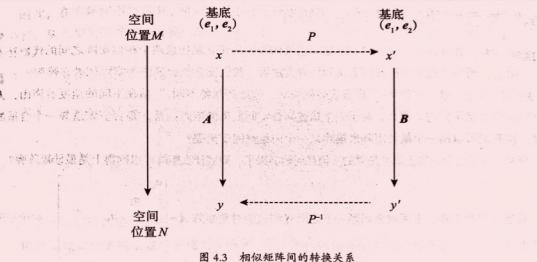


图 4.3 相似矩阵间的转换关系

假设讨论的前提是：在默认基底  $(e_1, e_2)$  的表示下，向量在矩阵  $A$  的作用下由坐标  $x$  变为坐标  $y$ ；而在新的基底  $(e'_1, e'_2)$  的表示下，向量在矩阵  $B$  的作用下由坐标  $x'$  变为坐标  $y'$ ，同时坐标之间的关系满足等式： $x' = Px$ 。投影？

在原始默认基底  $(e_1, e_2)$  的表示下，向量被表示为  $ae_1 + be_2$  的形式，其坐标为  $x = \begin{bmatrix} a \\ b \end{bmatrix}$ ；而在新基底  $(e'_1, e'_2)$  下坐标该如何表示呢？

在原始默认基底  $(e_1, e_2)$  的表示下，向量被表示为  $ae_1 + be_2$  的形式，其坐标为  $x = \begin{bmatrix} a \\ b \end{bmatrix}$ ；而在新基底  $(e'_1, e'_2)$  下坐标该如何表示呢？

假设两组基底的线性关系为  $\begin{cases} e'_1 = ce_1 + de'_2 \\ e'_2 = fe'_1 + ge'_2 \end{cases}$  明确了这层关系，就可以很容易的做一个基底变换，

即在新的基底  $(e'_1, e'_2)$  的表示下，向量被表示为  $ae_1 + be_2 = a(ce'_1 + de'_2) + b(fe'_1 + ge'_2) = (ac + bf)e'_1 + (ad + bg)e'_2$ ，其坐标为  $x' = \begin{bmatrix} ac + bf \\ ad + bg \end{bmatrix}$ 。

仔细观察一下这个向量的表达式，不难发现坐标可以分解成在新基底下的坐标。

一个矩阵和一个向量相乘的形式： $x' = \begin{bmatrix} c & f \\ d & g \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix}$ ，而恰好这个被分解出来的向量

下的坐标  $x$ 。在此基础上就把式子化简为  $x' = \begin{bmatrix} c & f \\ d & g \end{bmatrix} x$ 。此时，转换矩阵  $P$  就为  $P = \begin{bmatrix} c & f \\ d & g \end{bmatrix}$ 。矩阵  $P$  中各个元素的取值与两组基底之间的线性关系系数是完全一一对应的。

综上所述，对于空间位置里的同一个向量，选取不同的基底进行表示，其坐标值就是不同的。对于空间中的同一个位置变换，在不同的基底下，用于描述的矩阵也是不相同的，而这些不同矩阵所描述的线性变换是相似的，它们也被称为相似矩阵，这些矩阵之间的代数关系和其对应基底之间的数量转换关系经过计算发现是完全对应的。

## 4.2.4 用基变换的方法再次推导对角化过程

利用基底变换的方法，再来分析一下对角化的具体过程，先来看一个三维空间的例子。

在这个例子中，假设矩阵  $A$  的 3 个特征向量依次表示为  $p_1 = \begin{bmatrix} p_{11} \\ p_{21} \\ p_{31} \end{bmatrix}$ ,  $p_2 = \begin{bmatrix} p_{12} \\ p_{22} \\ p_{32} \end{bmatrix}$ ,  $p_3 = \begin{bmatrix} p_{13} \\ p_{23} \\ p_{33} \end{bmatrix}$ ，使用默

认的基底  $(e_1, e_2, e_3)$  和由特征向量  $(p_1, p_2, p_3)$  构成的基底分别对同一向量进行表示，即有  $x_1 e_1 + x_2 e_2 + x_3 e_3 = y_1 p_1 + y_2 p_2 + y_3 p_3$ 。

利用基向量将这个等式做进一步的代入操作，即

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \begin{bmatrix} p_{11} & p_{12} & p_{13} \\ p_{21} & p_{22} & p_{23} \\ p_{31} & p_{32} & p_{33} \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix}$$

再对式子进行展开：

$$\begin{aligned} x_1 &= p_{11}y_1 + p_{12}y_2 + p_{13}y_3 \\ x_2 &= p_{21}y_1 + p_{22}y_2 + p_{23}y_3 \\ x_3 &= p_{31}y_1 + p_{32}y_2 + p_{33}y_3 \end{aligned}$$

很明显，这一组等式可以用矩阵乘法对其进行表示：

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} p_{11} & p_{12} & p_{13} \\ p_{21} & p_{22} & p_{23} \\ p_{31} & p_{32} & p_{33} \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} \Rightarrow x = Py$$

A 的特征向量

note that the  
坐标 is new ones

取逆，则有  $y = P^{-1}x$ 。

有了  $y = P^{-1}x$  这个转换等式，就可以继续往下分析：

因此，对于向量在空间中的位置改变，在以  $(e_1, e_2, e_3, \dots, e_r)$  为基底进行的坐标表示下，我们的变换矩阵为  $A$ ，而在基底  $(p_1, p_2, p_3, \dots, p_r)$  下，变换矩阵则为  $A'$ 。

那么，首先从基底  $(e_1, e_2, e_3, \dots, e_r)$  变换到基底  $(p_1, p_2, p_3, \dots, p_r)$  下，向量的坐标值就由  $x$  变为  $P^{-1}x$ ，接着就能利用左乘矩阵  $A$  进行线性变换过程，即  $A(P^{-1}x)$ 。线性变换结束后，需要重新回到原始基底  $(e_1, e_2, \dots, e_r)$  下进行坐标表示，所以我们再做一次逆变换，即  $P(A(P^{-1}x))$ 。按照这种方式所进行的整个变换过程其实和  $Ax$  是殊途同归的，因此才有了  $A = PAP^{-1}$ 。

说到这里，对于对角化里所涉及的正反等式： $A = PAP^{-1}$ ,  $A = P^{-1}AP$ ，其本质意义就十分清晰了。

一个矩阵可以用  $AP = \lambda P$  表示  $A = P^{-1}\lambda P$  表示

用新 base 表示 original

vector

$$\textcircled{1} \quad \begin{bmatrix} \lambda_1 & \lambda_{2...} \end{bmatrix}$$

$$A\vec{v} = APy = \textcircled{2} \quad \textcircled{3} \quad \vec{y} = \lambda y$$

$\vec{v} = Py$   
新坐标  $[P]$

Thus  $\lambda y$   $\textcircled{4}$

- ① 向量  $x$  在特征向量上的投影
- ② 在  $P$  基底下的坐标

在新基下的坐标值  $\lambda P^{-1}x$

乘以新基  $P$   $P\lambda P^{-1}x$

# 新基 · 新坐标 → 向量 → 向量

## 4.2.3 几何意义

对于一个指定的向量  $v$ , 如果使用默认的基底  $(e_1, e_2, e_3, \dots, e_n)$  对其进行表示, 向量  $v$  即被表示为  $v = x_1e_1 + x_2e_2 + x_3e_3 + \dots + x_ne_n$ 。使用方阵  $A$  对其进行线性变换, 那么正如我们在前面的内容中所介绍的, 这一组默认的基底就会变成一组新的目标向量。一般情况下, 原始的基向量和转换后的目标向量是不共线的。

此时, 对该向量更换一组基底进行重新表示, 采用方阵  $A$  的一组特征向量  $(p_1, p_2, p_3, \dots, p_n)$  作为其新的基底, 则该向量被表示为  $v = y_1p_1 + y_2p_2 + y_3p_3 + \dots + y_np_n$ , 在新的基底下, 其新的坐标即为

$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$$

在此基础上利用方阵  $A$  对其进行线性变换, 则有  $Av = A(y_1p_1 + y_2p_2 + y_3p_3 + \dots + y_np_n) = Ay_1p_1 + Ay_2p_2 + Ay_3p_3 + \dots + Ay_np_n$ 。

基于等式  $A(y_1p_1 + y_2p_2 + y_3p_3 + \dots + y_np_n) = Ay_1p_1 + Ay_2p_2 + Ay_3p_3 + \dots + Ay_np_n$ , 再用上面讲过的特征定义式:  $Ap = \lambda p$ , 对其进行替换则有

$$Ay_1p_1 + Ay_2p_2 + Ay_3p_3 + \dots + Ay_np_n = \lambda_1y_1p_1 + \lambda_2y_2p_2 + \lambda_3y_3p_3 + \dots + \lambda_ny_np_n \rightarrow$$

这个等式的推导结果可以说是非常漂亮的, 利用基底  $(p_1, p_2, p_3, \dots, p_n)$  所表示的向量  $v$ , 经过

矩阵  $A$  的线性转换后, 其基底在保持不变的前提下, 向量的坐标由  $\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$  变为  $\begin{bmatrix} \lambda_1y_1 \\ \lambda_2y_2 \\ \vdots \\ \lambda_ny_n \end{bmatrix}$  的形式。

在新基底下  $[p_1 \dots p_n]$

最后再来概况一下其几何含义。对一个特定向量施加矩阵  $A$  所描述的线性变换, 使用矩阵

阵  $A$  的特征向量  $(p_1, p_2, p_3, \dots, p_n)$  作为空间的基底来对该向量进行坐标表示, 则该空间变换即可简化为各个维度的坐标值在其基向量的方向上对应伸缩  $\lambda_i$  倍。

## Understanding under PCA

### Change of Basis !!!

- Let  $X$  and  $Y$  be  $m \times n$  matrices related by a linear transformation  $P$ .
- $X$  is the original recorded data set and  $Y$  is a re-representation of that data set.

$$PX = Y$$

Let's define;

- $p_i$  are the rows of  $P$ .
- $x_i$  are the columns of  $X$ .
- $y_i$  are the columns of  $Y$ .
- What does this mean?
- $P$  is a matrix that transforms  $X$  into  $Y$ .
- Geometrically,  $P$  is a rotation and a stretch (scaling) which again transforms  $X$  into  $Y$ .
- The rows of  $P$ ,  $\{p_1, p_2, \dots, p_m\}$  are a set of new basis vectors for expressing the columns of  $X$ .

Which parameters do we want to keep?

- Parameter that doesn't depend on others (e.g. eye color), i.e. uncorrelated  $\Rightarrow$  low covariance.
- Parameter that changes a lot (grades)
- The opposite of noise
- High variance

① 新基底是变换矩阵的特征向量

② 在新基底下

$$A\vec{v} = APy = \lambda P\vec{y} = \lambda Py$$

$$\lambda = [\lambda_1 \lambda_2 \dots]$$

$$y = P^{-1}x$$

① 向量  $x$  在特征向量上的投影

② 在  $P$  基底下的坐标

### Change of Basis !!!

- Lets write out the explicit dot products of  $px$ .
- We can note the form of each column of  $Y$ .
- We can see that each coefficient of  $y_i$  is a dot-product of  $x_i$  with the corresponding row in  $P$ .

In other words, the  $j^{th}$  coefficient of  $y_i$  is a projection onto the  $j^{th}$  row of  $P$ .

Therefore, the rows of  $P$  are a new set of basis vectors for representing the columns of  $X$ .

- The row vectors  $\{p_1, p_2, \dots, p_m\}$  in this transformation will become the principal components of  $X$ .
- Changing the basis doesn't change the data – only its representation.
- Changing the basis is actually projecting the data vectors on the basis vectors.
- Geometrically,  $P$  is a rotation and a stretch of  $X$ .
- If  $P$  basis is orthonormal (length = 1) then the transformation  $P$  is only a rotation

## Covariance vector

$X X^T$

WISCONSIN  
UNIVERSITY

## PCA Derivation

- We want to find new points  $Z = Xw$  Project data vector
- Given  $w$  as a  $D \times 1$  column vector, the projection onto this axis is:  $Z = Xw$
- We want to maximize the variance of  $Z$  Find  $w$  that maximize variance of  $Z$
- $w^* = \operatorname{argmax}_w (\operatorname{Var}(Z))$   
 $= \operatorname{argmax}_w (\operatorname{Var}(Xw))$
- Assuming our data's columns are zero mean (which they should be if we standardized our data)

$$\operatorname{Var}(Xw) = \frac{(Xw)^T (Xw)}{N-1} = \frac{w^T X^T X w}{N-1} = w^T \Sigma w$$

- Where  $\Sigma$  is the covariance matrix of  $X$

projection  
? projection matrix

## PCA Derivation

$$w^* = \operatorname{argmax}_w (w^T \Sigma w)$$

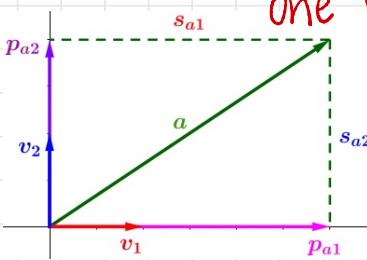
- To enforce that  $w$  is unit length (otherwise, we would maximize this by just setting  $w = [\infty, \dots, \infty]^T$ ) we can add the penalty term  $\lambda(w^T w - 1)$
- All put together we get the cost function:  
 $J(w) = w^T \Sigma w - \lambda(w^T w - 1)$
- Maximizing this, we get the **Lagrange** problem:  
 $w^* = \operatorname{argmax}_w (w^T \Sigma w - \lambda(w^T w - 1))$
- Taking the derivative and setting this equal to zero we get:  
 $2\Sigma w - 2\lambda w = 0$
- Which becomes  $(\Sigma - \lambda I)w = 0$
- We could also write this as  $\Sigma w = \lambda w$
- Since  $\Sigma$  is a matrix,  $\lambda$  is a scalar, and the thing that we're solving for,  $w$ , is a vector, this can be solved using **eigen-decomposition!**

19

# SVD

<https://towardsdatascience.com/svd-8c2f72e264f>

SVD is nothing more than decomposing vectors onto orthogonal axes



One Vector

Any vector can be expressed in terms of:

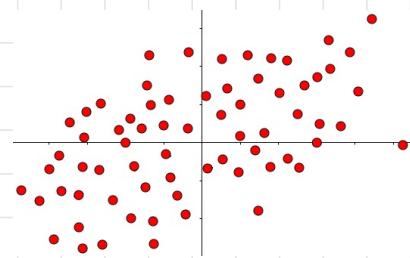
1. Projection directions unit vectors ( $v_1, v_2, \dots$ ).

2. The lengths of projections onto them ( $s_{a1}, s_{a2}, \dots$ ).

$$p_{a1} = s_{a1}^* v_1 \text{ and } p_{a2} = s_{a2}^* v_2$$

The vectors of projection ( $p_{a1}$  and  $p_{a2}$ )

- which are used to reconstruct the original vector
- deduced from the former 2 pieces.

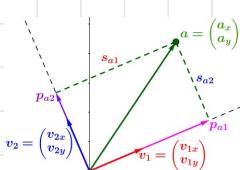


All what SVD does is extend this conclusion to more than one vector (or point) and to all dimensions :

$$S = \begin{pmatrix} s_{a1} & s_{a2} \\ s_{b1} & s_{b2} \end{pmatrix}$$

$$\text{Magnitude of 1st column} = \sigma_1 = \sqrt{(s_{a1})^2 + (s_{b1})^2}$$

$$\text{Magnitude of 2nd column} = \sigma_2 = \sqrt{(s_{a2})^2 + (s_{b2})^2}$$



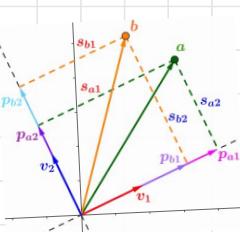
$$a^T \cdot v_1 = (a_x \ a_y) \cdot \begin{pmatrix} v_{1x} \\ v_{1y} \end{pmatrix} = s_{a1} \quad A \cdot V = \begin{pmatrix} a_x & a_y \end{pmatrix} \cdot \begin{pmatrix} v_{1x} & v_{2x} & \dots \\ v_{1y} & v_{2y} & \dots \end{pmatrix} = \begin{pmatrix} s_{a1} & s_{a2} & \dots \\ s_{b1} & s_{b2} & \dots \end{pmatrix} = S$$

$$a^T \cdot v_2 = (a_x \ a_y) \cdot \begin{pmatrix} v_{2x} \\ v_{2y} \end{pmatrix} = s_{a2}$$

$$a^T \cdot V = (a_x \ a_y) \cdot \begin{pmatrix} v_{1x} & v_{2x} \\ v_{1y} & v_{2y} \end{pmatrix} = (s_{a1} \ s_{a2})$$

$$A \cdot V = S$$

Matrix of points      The dot product performs the projection      Matrix of decomposition axes      Matrix of the lengths of projections



$$A = S V^{-1} = S V^T \quad A = U \Sigma V^T$$

$$S = \begin{pmatrix} s_{a1} & s_{a2} \\ s_{b1} & s_{b2} \end{pmatrix} = \begin{pmatrix} u_{a1} & u_{a2} \\ u_{b1} & u_{b2} \end{pmatrix} \begin{pmatrix} \sigma_1 & 0 \\ 0 & \sigma_2 \end{pmatrix} = U \Sigma$$

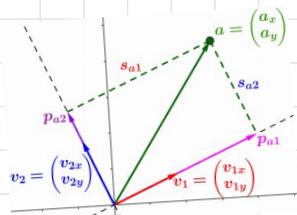
Lengths of projections on  $v_1$

Lengths of projections on  $v_2$

Lengths of projections on  $v_1$ , but divided by  $\sigma_1$  to become a unit vector

Lengths of projections on  $v_2$ , but divided by  $\sigma_2$  to become a unit vector

Hmm?



$$a^T \cdot v_1 = (a_x \ a_y) \cdot \begin{pmatrix} v_{1x} \\ v_{1y} \end{pmatrix} = s_{a1}$$

$$a^T \cdot v_2 = (a_x \ a_y) \cdot \begin{pmatrix} v_{2x} \\ v_{2y} \end{pmatrix} = s_{a2}$$

$$a^T \cdot V = (a_x \ a_y) \cdot \begin{pmatrix} v_{1x} & v_{2x} \\ v_{1y} & v_{2y} \end{pmatrix} = (s_{a1} \ s_{a2})$$

$$A \cdot V = \begin{pmatrix} a_x & a_y & \dots \\ b_x & b_y & \dots \\ \vdots & \vdots & \ddots \end{pmatrix} \cdot \begin{pmatrix} v_{1x} & v_{2x} & \dots \\ v_{1y} & v_{2y} & \dots \\ \vdots & \vdots & \ddots \end{pmatrix} = \begin{pmatrix} s_{a1} & s_{a2} & \dots \\ s_{b1} & s_{b2} & \dots \\ \vdots & \vdots & \ddots \end{pmatrix} = S$$

$n \times d$        $d \times d$        $n \times d$

$A \cdot V = S$  projection

Matrix of points  $\downarrow$   $A$   $\downarrow$  Matrix of the lengths of projections

Matrix of decompositon axes  $\downarrow$   $V$   $\downarrow$  base

The dot product performs the projection

$$A = S V^{-1} = S V^T$$

$v_1 \quad v_2$

$$S = \begin{pmatrix} (s_{a1}) & (s_{a2}) \\ (s_{b1}) & (s_{b2}) \end{pmatrix}$$

A column vector containing the lengths of projections of each point on the 1st axis  $v_1$

A column vector containing the lengths of projections of each point on the 2nd axis  $v_2$

$$A = U \Sigma V^T$$

$$S = U \Sigma$$

$$S = \begin{pmatrix} \frac{s_{a1}}{\sigma_1} & \frac{s_{a2}}{\sigma_2} \\ \frac{s_{b1}}{\sigma_1} & \frac{s_{b2}}{\sigma_2} \end{pmatrix} \begin{pmatrix} \sigma_1 & 0 \\ 0 & \sigma_2 \end{pmatrix} = \begin{pmatrix} u_{a1} & u_{a2} \\ u_{b1} & u_{b2} \end{pmatrix} \begin{pmatrix} \sigma_1 & 0 \\ 0 & \sigma_2 \end{pmatrix}$$

長度

$$S = \begin{pmatrix} s_{a1} & s_{a2} \\ s_{b1} & s_{b2} \end{pmatrix}$$

$$\text{Magnitude of 1st column} = \sigma_1 = \sqrt{(s_{a1})^2 + (s_{b1})^2}$$

$$\text{Magnitude of 2nd column} = \sigma_2 = \sqrt{(s_{a2})^2 + (s_{b2})^2}$$

$$S = \begin{pmatrix} s_{a1} & s_{a2} \\ s_{b1} & s_{b2} \end{pmatrix} = \begin{pmatrix} u_{a1} & u_{a2} \\ u_{b1} & u_{b2} \end{pmatrix} \begin{pmatrix} \sigma_1 & 0 \\ 0 & \sigma_2 \end{pmatrix} = U \Sigma$$

Lengths of projections on  $v_1$

Lengths of projections on  $v_2$

Lengths of projections on  $v_1$ , but divided by  $\sigma_1$  to become a unit vector

Lengths of projections on  $v_2$ , but divided by  $\sigma_2$  to become a unit vector

Hmm?

# 总结

eigenvectors 和 SVD 都是找到另一种方式表达同一个 matrix A

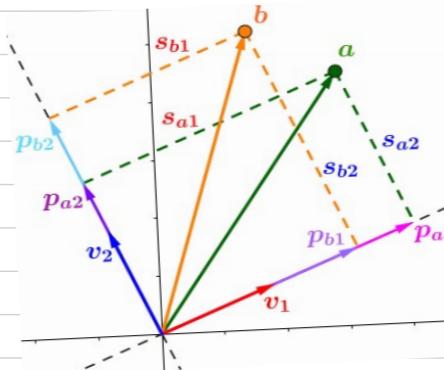
$$\textcircled{1} \quad AP = \lambda P \Rightarrow A = P^{-1} \lambda P \quad \text{用 eigenvector 表示}$$

$$\textcircled{2} \quad A = \underset{m \times n}{\cancel{S}} V^T = U \sum V^T \quad \text{decompose/project A to V}$$

$Av = \lambda v$     unit vector    length

$U : m \times n$   
 m 个行向量 in A  
 n 维列向量空间

$\Sigma : n \times n$



Equivalent form:  
 $AV = US$

$$A[v_1 \ v_2 \ \dots \ v_n] = [u_1 \ u_2 \ \dots \ u_m] \begin{bmatrix} \sigma_1 & & \\ & \ddots & \\ & & \sigma_n \\ & & 0 \end{bmatrix}$$

$Av_j = \sigma_j u_j, \quad j = 1, \dots, n$

## Geometry:

A is a linear transformation of vectors from  $R^n$  to  $R^m$ .

$u_1, \dots, u_m$  are an orthonormal basis of  $R^m$  (the output space).

$v_1, \dots, v_n$  are an orthonormal basis of  $R^n$  (the input space).

Each  $v_j$  is mapped to  $\sigma_j u_j$ .