

02-单位矩阵和逆矩阵

注意，本小节公式较多，且很多公式没法直接在 GitHub 上预览。建议 Clone 下来使用 Jupyter 阅读。

上一节讲述了一些线性代数的基本概念和矩阵的基本运算，这一节，带大家认识一下另外一个强大工具——矩阵的逆。

本节主要内容：

- 方阵
- 单位矩阵
- 逆矩阵
 - 使用行列式求 2×2 矩阵的逆矩阵
 - 使用初等行运算求更大矩阵的逆矩阵

2.1 方阵

什么是方阵？

顾名思义，方形的矩阵，即行数和列数相等的矩阵 ($n \times n$)。

比如：

$$M = \begin{bmatrix} 0 & 1 \\ 3 & 4 \end{bmatrix}$$

再比如：

$$M = \begin{bmatrix} 0 & 1 & 2 \\ 3 & 4 & 5 \\ 6 & 7 & 8 \end{bmatrix}$$

都是方阵。

方阵中存在主对角线的概念。什么是主对角线？

从方阵的左上角到右下角的所有元素的连线被称为方阵的主对角线。

比如上面第一个矩阵的 $0 \rightarrow 4$ ，第二个矩阵的 $0 \rightarrow 4 \rightarrow 8$ 都是主对角线。

2.2 单位矩阵

那什么是单位矩阵？

单位矩阵是一种特殊的方阵。单位矩阵是“主对角线上的数字全是1，其余位置的数字全是0”的方阵。

单位矩阵一般使用字母 I 表示。

比如2阶矩阵的单位矩阵为：

$$I = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

3阶矩阵的单位矩阵为：

$$I = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

更高阶矩阵以此类推。

尝试使用 Numpy 生成单位矩阵(以4阶单位矩阵为例)：

In [1]:

```
import numpy as np
np.identity(4)
```

Out[1]:

```
array([[1., 0., 0., 0.],
       [0., 1., 0., 0.],
       [0., 0., 1., 0.],
       [0., 0., 0., 1.]])
```

2.3 逆矩阵

在讨论逆矩阵之前，我们先回想一下倒数。

任何数和它的倒数的乘积都为1。

比如数字4，它的倒数是 $\frac{1}{4}$ ，也可以表示成 4^{-1} ， $4 * 4^{-1} = 1$ 。

而对于矩阵而言，单位矩阵就是标量运算里面的1，逆矩阵就是矩阵的倒数，所以我们也使用 A^{-1} 来表示矩阵 A 的逆矩阵。

注意，我们不能把逆矩阵写成 $\frac{1}{A}$ 的形式，因为矩阵不支持除法运算。

只有方阵才可能存在逆矩阵（但非方矩阵可以求伪逆矩阵，后面介绍）。

类比于倒数，在矩阵运算里，任何矩阵和它的逆矩阵的乘积都为同阶的单位矩阵。

即：

$$AA^{-1} = I \text{ 或 } A^{-1}A = I$$

2.4 使用行列式计算2阶矩阵的逆矩阵

如何计算矩阵的逆？

对于2阶矩阵，可以直接使用行列式进行计算。

即，假设存在2阶矩阵：

$$A = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$$

如果它的逆矩阵存在，则它的逆矩阵：

$$A^{-1} = \frac{1}{ad - bc} \begin{bmatrix} d & -b \\ -c & a \end{bmatrix}$$

关于行列式的概念我就不再展开了，对于高阶矩阵而言，行列式使用起来非常复杂。后面会介绍使用初等行运算的方法求2阶以上的矩阵的逆矩阵的方法。

注意，上面提到了“如果它的逆矩阵存在”，为什么这么说呢？因为有些矩阵是没有逆矩阵的。拿2阶矩阵举例，看上面的公式，当 $ad = bc$ 时，除数为0，运算无意义。

那么求矩阵的逆矩阵有什么意义呢？来看一下经典的 鸡兔同笼 问题。

已知在一个笼子里，有公鸡和兔子加起来共有15只，它们的脚加起来共50只。请问，笼中兔子和鸡各有多少只？

比较经典的方法是列方程组，我们现在使用矩阵来求解，你会发现它和方程组求解有异曲同工之妙。

首先，我们设笼中有兔子 x_1 只，鸡 x_2 只，则我们可以写出如下矩阵：

$$\begin{bmatrix} x_1 & x_2 \end{bmatrix} \begin{bmatrix} 1 & 4 \\ 1 & 2 \end{bmatrix} = \begin{bmatrix} 15 & 50 \end{bmatrix}$$

有点懵？没关系，还记得上一节说的矩阵乘法怎么算吗？我们把它展开：

$$\begin{aligned} x_1 + x_2 &= 15 \\ x_1 * 4 + x_2 * 2 &= 50 \end{aligned}$$

咦？这不就是方程组吗？没错，这就是我说的异曲同工之妙咯~它们本质上表达的内容是一致的。

但是这里我们不用方程组的方法求解，咱们来求矩阵的逆。设：

$$\begin{aligned} X &= \begin{bmatrix} x_1 & x_2 \end{bmatrix} \\ A &= \begin{bmatrix} 1 & 4 \\ 1 & 2 \end{bmatrix} \\ B &= \begin{bmatrix} 15 & 50 \end{bmatrix} \end{aligned}$$

则，上面的矩阵运算可以写成：

$$XA = B$$

两边同时乘以 A 的逆矩阵，则：

$$XAA^{-1} = BA^{-1}$$

因为任何矩阵和它的逆矩阵的乘积都是单位矩阵，所以：

$$XI = BA^{-1}$$

单位矩阵相当于矩阵里面的数字1，任何矩阵和单位矩阵相乘都等于它本身，所以：

$$X = BA^{-1}$$

你看，我们已经使用逆矩阵表示出 X 的值了，接下来计算具体的值：

$$\begin{aligned} A^{-1} &= \frac{1}{1 * 2 - 4 * 1} \begin{bmatrix} 2 & -4 \\ -1 & 1 \end{bmatrix} = \begin{bmatrix} -1 & 2 \\ 0.5 & -0.5 \end{bmatrix} \\ X &= \begin{bmatrix} 15 & 50 \end{bmatrix} \begin{bmatrix} -1 & 2 \\ 0.5 & -0.5 \end{bmatrix} = \begin{bmatrix} 10 & 5 \end{bmatrix} \end{aligned}$$

结果出来了，共有兔子10只，鸡5只。

上面是我们手算的结果，在 Numpy 中如何实现呢？

先求 A 矩阵的逆：

In [2]:

```
a = np.matrix([[1, 4], [1, 2]])
ai = a.I
ai
```

Out[2]:

```
matrix([[ -1. ,  2. ],
        [ 0.5, -0.5]])
```

结果和我们算的一样~

再求矩阵 X :

In [3]:

```
np.dot(np.array([15, 50]), ai)
```

Out[3]:

```
matrix([[10.,  5.]])
```

大功告成!

2.5 使用初等行运算求解逆矩阵

对于2阶以上的矩阵，使用初等行运算会更加方便。

对于一个矩阵 A ，我们在 A 的右侧加上一个同阶的单位矩阵，这样的矩阵我们称为 增广矩阵 。

比如，对于矩阵：

$$A = \begin{bmatrix} -1 & -2 & 1 \\ 1 & 1 & 0 \\ -2 & 0 & -1 \end{bmatrix}$$

它的增广矩阵可以表示如下：

$$\left[\begin{array}{ccc|ccc} -1 & -2 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 \\ -2 & 0 & -1 & 0 & 0 & 1 \end{array} \right]$$

为什么可以使用增广矩阵和初等行变换求解逆矩阵？请看，这是一个增广矩阵：

$$\left[\begin{array}{c|c} A & I \end{array} \right]$$

给两边同时乘上 A^{-1} ：

$$\left[\begin{array}{c|c} AA^{-1} & IA^{-1} \end{array} \right]$$

即：

$$\left[\begin{array}{c|c} I & A^{-1} \end{array} \right]$$

是不是很神奇？我们只需要把左边的 A 矩阵通过初等行运算转变为单位矩阵，右边增广部分的矩阵则自动变成了矩阵的逆。

我们以这个矩阵为例：

$$A = \begin{bmatrix} -1 & -2 & 1 \\ 1 & 1 & 0 \\ -2 & 0 & -1 \end{bmatrix}$$

其增广矩阵：

$$\left[\begin{array}{ccc|ccc} -1 & -2 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 \\ -2 & 0 & -1 & 0 & 0 & 1 \end{array} \right]$$

将第3行加在第1行上：

$$\left[\begin{array}{ccc|ccc} -3 & -2 & 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 & 1 & 0 \\ -2 & 0 & -1 & 0 & 0 & 1 \end{array} \right]$$

将第2行乘以2，加在第1行上：

$$\left[\begin{array}{ccc|ccc} -1 & 0 & 0 & 1 & 2 & 1 \\ 1 & 1 & 0 & 0 & 1 & 0 \\ -2 & 0 & -1 & 0 & 0 & 1 \end{array} \right]$$

使第1行乘以 -1 ：

$$\left[\begin{array}{ccc|ccc} 1 & 0 & 0 & -1 & -2 & -1 \\ 1 & 1 & 0 & 0 & 1 & 0 \\ -2 & 0 & -1 & 0 & 0 & 1 \end{array} \right]$$

使第2行减去第1行：

$$\left[\begin{array}{ccc|ccc} 1 & 0 & 0 & -1 & -2 & -1 \\ 0 & 1 & 0 & 1 & 3 & 1 \\ -2 & 0 & -1 & 0 & 0 & 1 \end{array} \right]$$

将第1行乘以2，加在第3行上：

$$\left[\begin{array}{ccc|ccc} 1 & 0 & 0 & -1 & -2 & -1 \\ 0 & 1 & 0 & 1 & 3 & 1 \\ 0 & 0 & -1 & -2 & -4 & -1 \end{array} \right]$$

使第3行乘以-1：

$$\left[\begin{array}{ccc|ccc} 1 & 0 & 0 & -1 & -2 & -1 \\ 0 & 1 & 0 & 1 & 3 & 1 \\ 0 & 0 & 1 & 2 & 4 & 1 \end{array} \right]$$

左边已经变成单位矩阵了，右边的结果即为矩阵A的逆矩阵。

算的对不对？使用 Numpy 验证一下：

In [4]:

```
np.matrix([[ -1, -2, 1], [1, 1, 0], [ -2, 0, -1]]).I
```

Out[4]:

```
matrix([[ -1., -2., -1.],
        [ 1.,  3.,  1.],
        [ 2.,  4.,  1.]])
```

你现在掌握了逆矩阵的求解方法！恭喜！