

作者	生姜 DrGinger
脚本	生姜 DrGinger
视频	崔崔 CuiCui
开源学习资源	<a href="https://github.com/Visualize-ML">https://github.com/Visualize-ML</a>
平台	<a href="https://www.youtube.com/@DrGinger_Jiang">https://www.youtube.com/@DrGinger_Jiang</a> <a href="https://space.bilibili.com/3546865719052873">https://space.bilibili.com/3546865719052873</a> <a href="https://space.bilibili.com/513194466">https://space.bilibili.com/513194466</a>

## 7.3 初等行变换



### 本节你将掌握的核心技能：

- ▶ 初等行变换：行交换、行数乘、行倍加三类基本操作。
- ▶ 增广矩阵：将线性方程组写成  $[A | b]$  形式。
- ▶ 求解线性方程组：通过初等行变换将增广矩阵化为简化形式。
- ▶ 初等行变换求逆矩阵。
- ▶ Python 编程实现高斯消元法、逆矩阵求解。

本节介绍了初等行变换及其在求解线性方程组和求逆矩阵中的应用。首先，通过“鸡兔同笼”问题，引入增广矩阵的概念，并逐步进行初等行变换，包括行数乘、行交换和行倍加，以将方程组化为简化形式。接着，利用相同的行变换方法，将矩阵  $A$  转化为单位矩阵，从而求得其逆矩阵。整个过程中，我们观察了方程组、矩阵、增广矩阵、线性组合的变化，直观展示了初等行变换在数学计算中的作用。

### 初等行变换

本章第一节提过**初等行变换** (elementary row operations)。**初等行变换**的作用是通过行简化矩阵，特别是在求解线性方程组、计算矩阵的秩和求逆矩阵时常用。

**初等行变换**是对矩阵的行完成的基本操作，包括：

- ▶ **行交换** (row swapping)，交换选定两行；
- ▶ **行数乘** (row scaling)，某一行所有元素数乘**非零**标量；
- ▶ **行倍加** (row sum)，一行的**非零**倍数加到另一行。

### 回看鸡兔同笼问题

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>

本书配套微课视频均发布在 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](mailto:jiang.visualize.ml@gmail.com)

回看简化版的“鸡兔同笼”问题。

笼子里面有  $x_1$  只鸡， $x_2$  只兔；共有 3 个头，8 只脚，写成线性方程组

$$\begin{cases} x_1 + x_2 = 3 \\ 2x_1 + 4x_2 = 8 \end{cases} \quad (1)$$

图 1 (a) 所示为以上线性方程组对应的直线；由于两个方程未知数的系数均非零，两条直线均为斜线。

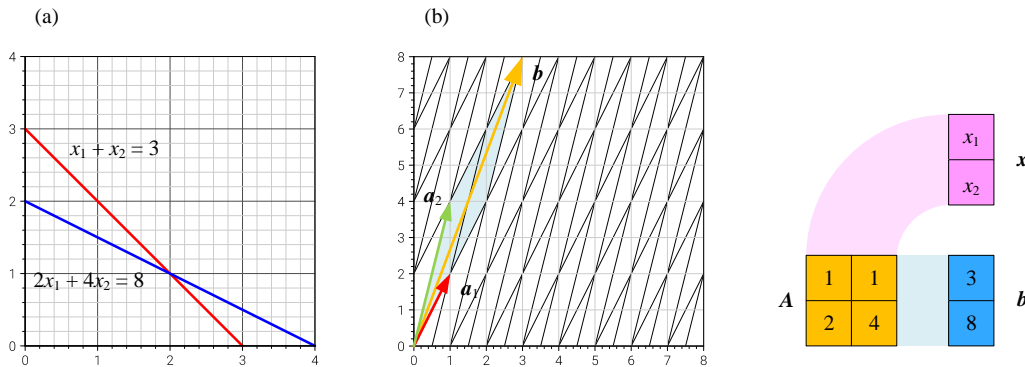


图 1. 原始方程组

(1) 写成矩阵乘法形式

$$\underbrace{\begin{bmatrix} 1 & 1 \\ 2 & 4 \end{bmatrix}}_A \underbrace{\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}}_x = \underbrace{\begin{bmatrix} 3 \\ 8 \end{bmatrix}}_b \quad (2)$$

把这个问题写成**增广矩阵** (augmented matrix)。

增广矩阵是在系数矩阵  $A$  右侧附上常数向量  $b$  形成的新矩阵，即

$$[A \mid b] = \left[ \begin{array}{cc|c} 1 & 1 & 3 \\ 2 & 4 & 8 \end{array} \right] \quad (3)$$

**⚠ 注意**，有些时候省略上式中  $A$ 、 $b$  之间的竖线。

大家很快就会看到对增广矩阵进行初等行变换可以求解线性方程组  $Ax = b$ 。

根据上一节内容，我们也可以把系数矩阵  $A$  写成列向量  $[a_1, a_2]$ ，这样“鸡兔同笼”问题写成线性组合形式

$$x_1 \underbrace{\begin{bmatrix} 1 \\ 2 \end{bmatrix}}_{a_1} + x_2 \underbrace{\begin{bmatrix} 1 \\ 4 \end{bmatrix}}_{a_2} = \underbrace{\begin{bmatrix} 3 \\ 8 \end{bmatrix}}_b \quad (4)$$

图 1 (b) 对应上式的线性组合。

图 1 (b) 中， $a_1$ 、 $a_2$  夹角小于 90 度，但是  $a_1$ 、 $a_2$  不共线 (线性无关)，形成的网格为平行四边形。方阵  $A$  的行列式为  $\det(A) = 2$  (单个平行四边形的面积)，方阵  $A$  可逆，这也意味着  $2 \times 2$  方阵  $A$  的秩为 2。

下面让我们用初等行变换一步步求解这个问题，并观察求解过程直线、系数矩阵、增广矩阵、线性组合变化。

$$R_2/2 \rightarrow R_2$$

对第二行完成**行数乘**  $R_2/2 \rightarrow R_2$ ，线性方程组变为

$$\begin{cases} x_1 + x_2 = 3 \\ x_1 + 2x_2 = 4 \end{cases} \quad (5)$$

**行数乘**  $R_2/2 \rightarrow R_2$  表示将第 2 行的每个元素都除以 2，并替换原来的第 2 行，目的是使主元为 1，便于后续运算。**主元** (pivot, pivot element) 就是在消去过程中，每一行中最左边的非零系数。

⚠ 注意，主元不同于主对角线元素，因为主元会随着行变换改变。

图 2 (a) 所示上式线性方程组对应的两条直线；比较图 1 (a)、图 2 (a)，显然，**行数乘** 不改变直线位置、形态。

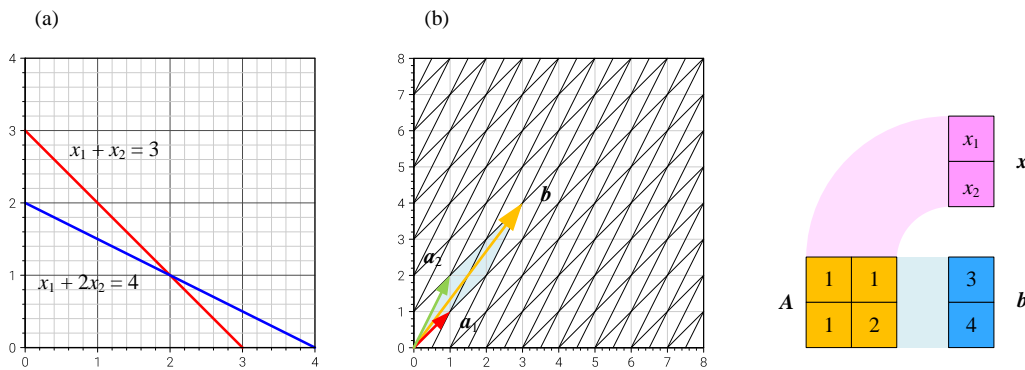


图 2. 行数乘,  $R_2/2 \rightarrow R_2$

(5) 对应的矩阵乘法变为

$$\underbrace{\begin{bmatrix} 1 & 1 \\ 1 & 2 \end{bmatrix}}_A \underbrace{\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}}_x = \underbrace{\begin{bmatrix} 3 \\ 4 \end{bmatrix}}_b \quad (6)$$

增广矩阵变为

$$[A \mid b] = \left[ \begin{array}{cc|c} 1 & 1 & 3 \\ 1 & 2 & 4 \end{array} \right] \quad (7)$$

线性组合变为

$$x_1 \underbrace{\begin{bmatrix} 1 \\ 1 \end{bmatrix}}_{a_1} + x_2 \underbrace{\begin{bmatrix} 1 \\ 2 \end{bmatrix}}_{a_2} = \underbrace{\begin{bmatrix} 3 \\ 4 \end{bmatrix}}_b \quad (8)$$

图 2 (b) 所示为线性组合，**行数乘**改变了矩阵  $A$  的列向量，行列式变为 1。这时网格还是平行四边形。

$$R_2 - R_1 \rightarrow R_2$$

通过**行倍加**  $R_2 - R_1 \rightarrow R_2$ ，即用 (5) 第 2 行减去第 1 行，然后用结果替换第 2 行，这样消去了第 2 行的  $x_1$ 。线性方程组变为

$$\begin{cases} x_1 + x_2 = 3 \\ x_2 = 1 \end{cases} \quad (9)$$

如图 3 (a) 所示， $x_2 = 1$  对应一条水平线，这意味着  $x_2$  的解已经出现！

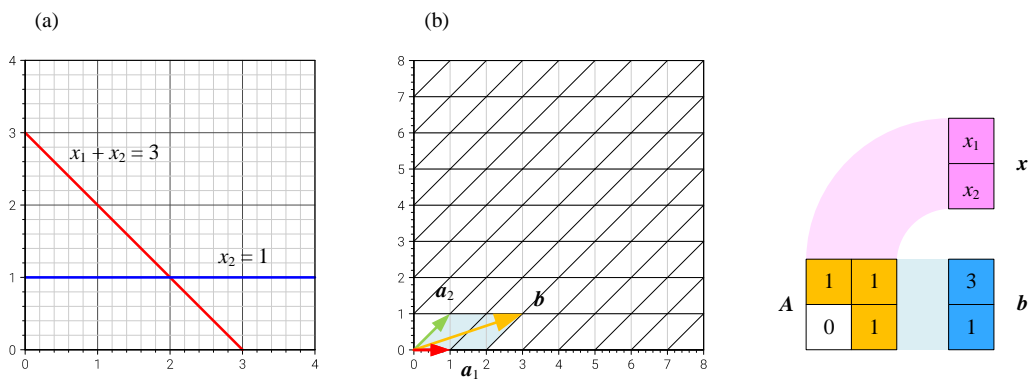


图 3. 行倍加， $R_2 - R_1 \rightarrow R_2$

(9) 对应的矩阵乘法变为

$$\underbrace{\begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}}_A \underbrace{\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}}_x = \underbrace{\begin{bmatrix} 3 \\ 1 \end{bmatrix}}_b \quad (10)$$

我们发现矩阵  $A$  已经变成上三角矩阵！矩阵的行列式为主对角线元素乘积，结果为 1。

增广矩阵变为

$$[A \mid b] = \left[ \begin{array}{cc|c} 1 & 1 & 3 \\ 0 & 1 & 1 \end{array} \right] \quad (11)$$

线性组合变为

$$x_1 \underbrace{\begin{bmatrix} 1 \\ 0 \end{bmatrix}}_{a_1} + x_2 \underbrace{\begin{bmatrix} 1 \\ 1 \end{bmatrix}}_{a_2} = \underbrace{\begin{bmatrix} 3 \\ 1 \end{bmatrix}}_b \quad (12)$$

如图 3 (b) 所示，平行四边形网格变为沿横轴剪切的网格。

$$R_1 - R_2 \rightarrow R_1$$

再利用**行倍加**  $R_1 - R_2 \rightarrow R_1$ ，即用第 1 行减去第 2 行的对应元素，结果替换原来的第 1 行。线性方程组变为

$$\begin{cases} x_1 = 2 \\ x_2 = 1 \end{cases} \quad (13)$$

观察图 4 (a)，两条直线已经变成竖直、水平， $x_1$ 、 $x_2$  的解都出现了！

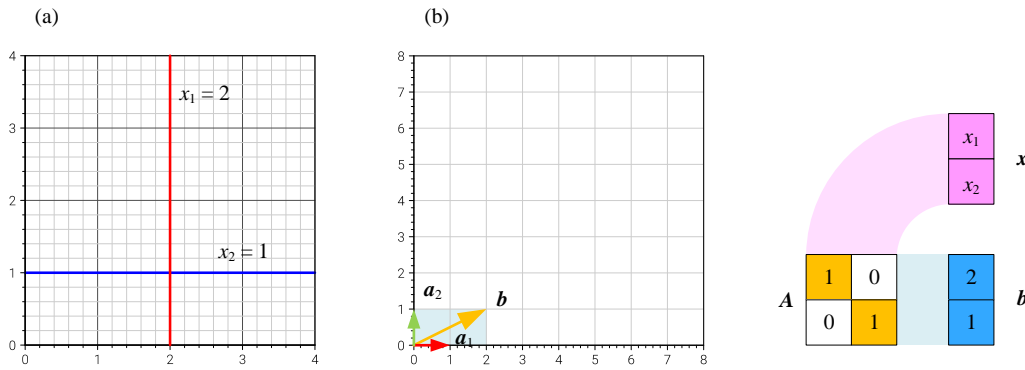


图 4. 行倍加， $R_1 - R_2 \rightarrow R_1$

矩阵乘法变为

$$\underbrace{\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}}_A \underbrace{\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}}_x = \underbrace{\begin{bmatrix} 2 \\ 1 \end{bmatrix}}_b \quad (14)$$

矩阵  $A$  变成了单位矩阵。根据单位矩阵的矩阵乘法性质，上式可以化简为

$$\underbrace{\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}}_x = \underbrace{\begin{bmatrix} 2 \\ 1 \end{bmatrix}}_b \quad (15)$$

增广矩阵变为

$$[A \mid b] = \left[ \begin{array}{cc|c} 1 & 0 & 2 \\ 0 & 1 & 1 \end{array} \right] \quad (16)$$

线性组合变为

$$x_1 \underbrace{\begin{bmatrix} 1 \\ 0 \end{bmatrix}}_{a_1} + x_2 \underbrace{\begin{bmatrix} 0 \\ 1 \end{bmatrix}}_{a_2} = \underbrace{\begin{bmatrix} 2 \\ 1 \end{bmatrix}}_b \quad (17)$$


如图 4 (b) 所示，网格已经变成单位正方形；两个列向量已经变成标准正交基  $[e_1, e_2]$ 。

通过初等行变换求解线性方程组，我们观察到了方程、矩阵、增广矩阵、线性组合（平行四边形网格）的变换。

## 编程：初等行变换求解线性方程组

代码 1 为用初等行变换求解线性方程组的 Python 代码。下面聊聊其中的关键语句。

- a 定义一个名为 `gauss_elimination` 的函数，该函数接收两个参数： $A$  为系数矩阵，一个二维 NumPy 数组； $b$  为常数向量，一个一维 NumPy 数组。
- b 将  $A$  和  $b$  转换为 `float` 类型，确保计算过程中不会因为整数运算导致错误。
- c 构造增广矩阵  $[A|b]$ ；用 `b.reshape(-1, 1)` 将  $b$  转换为列向量，然后用 `np.hstack()` 把  $A$  和  $b$  水平拼接在一起，形成新的  $n \times (n+1)$  形状的矩阵  $[A|b]$ ，变量名为 `Ab`。
- d 用 `for` 循环将  $A$  转换为上三角矩阵。使用 `for` 循环遍历矩阵  $A$  的列索引  $i$ ，控制消元过程。
- e 在当前列  $i$  及其下面的所有行中，找到绝对值最大的元素所在的行（称为主元行）。`np.abs(Ab[i:, i])` 取  $i$  列的绝对值，`np.argmax` 找出其中最大值的索引，加上  $i$  计算其在 `Ab` 中的真实行号。
- f 交换当前行  $i$  和 `max_row`，确保对角线元素（主元）绝对值尽可能大，以减少计算误差。
- g 用行数乘，将主元系数变为 1。
- h 使用嵌套循环，将当前列  $i$  下面所有行的  $i$  列元素消去，使其变为 0。
- i 用行倍加法消去  $i$  列的元素。具体做法是：`Ab[j, i]` 是第  $j$  行当前列的系数。`Ab[j, i] * Ab[i]` 是要消去  $j$  行  $i$  列元素所需要的倍数。`Ab[j] - Ab[j, i] * Ab[i]` 是行变换的新值。
- j 初始化解向量  $x$ ，用零填充长度为  $n$  的数组。
- k 从最后一行开始，逐行向上求解 `x[i]`。`Ab[i, -1]` 是  $b$  的值，即增广矩阵的最后一列。`np.dot(Ab[i, i+1:n], x[i+1:n])` 计算已知解的加权和，`Ab[i, -1] - np.dot(Ab[i, i+1:n], x[i+1:n])` 就是当前行的解。
- 1 定义  $A$ 、 $b$ ，并调用自定义函数求解线性方程组。

代码 1. 初等行变换求解线性方程组 |  LA\_Ch07\_03\_01.ipynb

```

## 初始化
import numpy as np

## 自定义函数
a def gauss_elimination(A, b):
    # 确保使用浮点数计算
    b A = A.astype(float)
    b b = b.astype(float)
    b n = len(b)

    # 构造增广矩阵 [A|b]
    c Ab = np.hstack([A, b.reshape(-1, 1)])

    # 原始矩阵 A > 上三角
    d for i in range(n):
        # 1) 行交换, 选择绝对值最大为主元
        # 主元: 主对角线元素
        e max_row = np.argmax(np.abs(Ab[i:, i])) + i
        f Ab[[i, max_row]] = Ab[[max_row, i]]
        # 交换行, 行 i <> 行 max_row

        # 2) 行数乘, 归一化主元, 主元变为 1
        g Ab[i] = Ab[i] / Ab[i, i]

        # 3) 行倍加, 消元, 使主元下面所有行的 i 列变为 0
        h for j in range(i+1, n):
            # Ab[j] -= Ab[j, i] * Ab[i]
            i Ab[j] = Ab[j] - Ab[j, i] * Ab[i]

    # 回代求解, 上三角矩阵 > 单位矩阵
    j x = np.zeros(n)
    k for i in range(n-1, -1, -1):
        x[i] = Ab[i, -1] - np.dot(Ab[i, i+1:n], x[i+1:n])

    return x

## 定义矩阵 A 和向量 b
l A = np.array([[1, 3, 6],
               [2, 7, 14],
               [0, 2, 5]])

b = np.array([25, 58, 19])

## 求解 Ax = b
x = gauss_elimination(A, b)


```

为了方便大家理解这段代码，下面我们代码 1 给出的例子，按照上述逻辑逐步计算。

## 构造增广矩阵

根据给出的  $A$  和  $b$ ，构造增广矩阵

$$\begin{bmatrix} 1 & 3 & 6 & 25 \\ 2 & 7 & 14 & 58 \\ 0 & 2 & 5 & 19 \end{bmatrix} \quad (18)$$

对应代码 。

## 第一列

**行交换：**看 (18) 第 1 列，第 1 列、第 2 行的绝对值最大；交换第 1 行、第 2 行

$$\begin{bmatrix} 2 & 7 & 14 & 58 \\ 1 & 3 & 6 & 25 \\ 0 & 2 & 5 & 19 \end{bmatrix} \quad (19)$$

**行数乘：**(19) 第 1 列主元 (第 1 行、第 1 列) 归一化

$$\begin{bmatrix} 1 & 3.5 & 7 & 29 \\ 1 & 3 & 6 & 25 \\ 0 & 2 & 5 & 19 \end{bmatrix} \quad (20)$$

**行加倍：**消元，使 (20) 第 1 列主元以下的元素为 0

$$\begin{bmatrix} 1 & 3.5 & 7 & 29 \\ 0 & -0.5 & -1 & -4 \\ 0 & 2 & 5 & 19 \end{bmatrix} \quad (21)$$

## 第二列

**行交换：**看 (21) 第 2 列 (主元以下元素)，第 2 列、第 3 行元素的绝对值最大；交换第 2 行、第 3 行

$$\begin{bmatrix} 1 & 3.5 & 7 & 29 \\ 0 & 2 & 5 & 19 \\ 0 & -0.5 & -1 & -4 \end{bmatrix} \quad (22)$$

**行数乘：**(22) 第 2 列主元 (第 2 行、第 2 列) 归一化

$$\begin{bmatrix} 1 & 3.5 & 7 & 29 \\ 0 & 1 & 2.5 & 9.5 \\ 0 & -0.5 & -1 & -4 \end{bmatrix} \quad (23)$$

**行加倍：**消元，使 (23) 第 2 列主元以下的元素为 0

$$\begin{bmatrix} 1 & 3.5 & 7 & 29 \\ 0 & 1 & 2.5 & 9.5 \\ 0 & 0 & 0.25 & 0.75 \end{bmatrix} \quad (24)$$



### 第三列

由于方阵  $A$  只有 3 列，这一次只需要完成**行数乘**：(24) 第 3 列主元 (第 3 行、第 3 列) 归一化

$$\begin{bmatrix} 1 & 3.5 & 7 & 29 \\ 0 & 1 & 2.5 & 9.5 \\ 0 & 0 & 1 & 3 \end{bmatrix} \quad (25)$$

这时，<sup>d</sup> 的 for 循环结束，矩阵  $A$  已经被转化成下三角矩阵。

### 反向代回

带回求解则特别简单

$$\begin{aligned} x_3 &= 3 \\ x_2 &= 9.5 - 2.5 \times 3 = 2 \\ x_1 &= 29 - 3.5 \times 2 - 7 \times 3 = 1 \end{aligned} \quad (26)$$

以上计算对应<sup>k</sup>。上述过程可以用图 5 来总结。

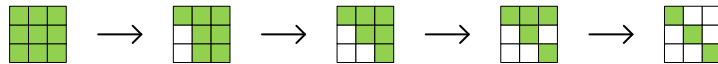


图 5. 初等行变换的流程

### 手解逆矩阵

本书第 5 章介绍了如何用伴随矩阵法手解逆矩阵，当时提到过初等行变换也可以用来手解逆矩阵。

下面让我们用初等行变换求解 (2) 中矩阵  $A$  的逆矩阵。

首先把  $A$  与单位矩阵  $I$  拼接形成增广矩阵

$$[A \mid I] = \left[ \begin{array}{cc|cc} 1 & 1 & 1 & 0 \\ 2 & 4 & 0 & 1 \end{array} \right] \quad (27)$$

我们的目标是通过初等变换，把增广矩阵左侧的方阵  $A$  变为  $I$ ，右侧的矩阵就是逆矩阵  $A^{-1}$ ；就是通过初等行变换把增广矩阵变为

$$[I \mid A^{-1}] \quad (28)$$

下面，让我们重复之前的初等行变换。

对第二行完成**行数乘**  $R_2/2 \rightarrow R_2$ ，(27) 增广矩阵变为

$$\left[ \begin{array}{cc|cc} 1 & 1 & 1 & 0 \\ 1 & 2 & 0 & 1/2 \end{array} \right] \quad (29)$$

通过**行倍加**  $R_2 - R_1 \rightarrow R_2$ , (29) 增广矩阵变为

$$\left[ \begin{array}{cc|cc} 1 & 1 & 1 & 0 \\ 0 & 1 & -1 & 1/2 \end{array} \right] \quad (30)$$

最后利用**行倍加**  $R_1 - R_2 \rightarrow R_1$ , (30) 增广矩阵变为

$$\left[ \begin{array}{cc|cc} 1 & 0 & 2 & -1/2 \\ 0 & 1 & -1 & 1/2 \end{array} \right] \quad (31)$$

至此，增广矩阵左侧方阵变为  $I$ ，右侧方阵就是逆矩阵  $A^{-1}$ 。

验证一下，矩阵  $A$  乘  $A^{-1}$

$$\begin{bmatrix} 1 & 1 \\ 2 & 4 \end{bmatrix} @ \begin{bmatrix} 2 & -0.5 \\ -1 & 0.5 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \quad (32)$$

逆矩阵  $A^{-1}$  乘  $A$

$$\begin{bmatrix} 2 & -0.5 \\ -1 & 0.5 \end{bmatrix} @ \begin{bmatrix} 1 & 1 \\ 2 & 4 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \quad (33)$$

## 编程：初等行变换求解逆矩阵

代码 2 用初等行变换求解逆矩阵。下面聊聊其中关键语句。

- a** 使用 `if n != m`: 这一行代码检查  $A$  是否是方阵。方阵的定义是行数等于列数，即  $n == m$ ，如果  $n$  不等于  $m$ ，说明  $A$  不是方阵，我们就打印错误信息。并返回 `None`，表示这个函数无法继续执行。
- b** 使用 `np.linalg.det(A)` 计算  $A$  的行列式，并用 `if np.linalg.det(A) == 0`: 这行代码检查它是否为 0。如果行列式等于 0，则  $A$  是一个奇异矩阵，没有逆矩阵，所以我们打印错误信息，并返回 `None`。
- c** 创建一个单位矩阵  $I$ ，它的大小与  $A$  相同。单位矩阵是主对角线上全是 1，其他地方全是 0 的矩阵。
- d** 使用 `A_I = np.hstack((A, I))` 这行代码将  $A$  和  $I$  水平拼接在一起，形成一个增广矩阵  $[A | I]$ 。在数学上，这样的矩阵表示我们要同时对  $A$  和  $I$  进行相同的行变换，最终把  $A$  变成单位矩阵， $I$  就变成  $A$  的逆矩阵。
- e** 进入 `for i in range(n)`: 这一循环，它的目的是对  $A_I$  进行高斯-约当消元法，把  $A$  变成单位矩阵。循环的  $i$  变量表示当前正在处理的列索引。
- f** 找到当前列中绝对值最大的元素所在的行 `max_row`，然后用 `A_I[[i, max_row]] = A_I[[max_row, i]]` 交换  $i$  行和 `max_row` 行。
- g** 执行 `A_I[i] = A_I[i] / A_I[i, i]` 这行代码，它的作用是把当前行的主对角线元素变成 1。我们让  $i$  行的所有元素都除以 `A_I[i, i]`，这样主元就变成 1，而其他元素按比例缩放。

h 进入 `for j in range(n):` 这个嵌套循环，它的目的是将  $i$  列的（主元之外）其他行的元素都变成 0。我们用 `if i != j:` 确保我们不修改当前的主元行，然后执行  $A\_I[j] -= A\_I[j, i] * A\_I[i]$ ，它的作用是用当前的主元行对  $j$  行进行消元，使  $j$  行  $i$  列的元素变成 0。

i 使用 `A_inv = A_I[:, n:]` 提取右侧矩阵  $A$  的逆。

代码 2. 初等行变换求解逆矩阵 |  LA\_Ch07\_03\_02.ipynb

```

## 初始化
import numpy as np

## 自定义函数
def inverse_matrix(A):

    # 确保 A 是 NumPy 数组并转换为浮点数类型
    A = np.array(A, dtype=float)

    # 获取矩阵的行数和列数
    n, m = A.shape

    # 判断是否为方阵
    if n != m:
        print("错误：矩阵不是方阵，无法求逆。")
        return None

    # 计算行列式，判断是否可逆
    if np.linalg.det(A) == 0:
        print("错误：矩阵的行列式为 0，无法求逆。")
        return None

    # 创建单位矩阵 I
    I = np.eye(n)

    # 构造增广矩阵 [A | I]
    A_I = np.hstack((A, I))

    # 进行高斯-约当消元法
    for i in range(n):
        # 1) 行交换，选取主元
        max_row = np.argmax(np.abs(A_I[i:, i])) + i
        A_I[[i, max_row]] = A_I[[max_row, i]]

        # 2) 行数乘，归一化主元（主对角线元素变为 1）
        A_I[i] = A_I[i] / A_I[i, i]

        # 3) 行加倍，消元（使主元列的其他元素变为 0）
        for j in range(n):
            if i != j:
                A_I[j] -= A_I[j, i] * A_I[i]

    # 提取逆矩阵
    A_inv = A_I[:, n:]

    return A_inv

## 定义矩阵 A
A = np.array([[1, 3, 6],
               [2, 7, 14],
               [0, 2, 5]])

## 计算逆矩阵
A_inv = inverse_matrix(A)

```





请大家用 DeepSeek/ChatGPT 等工具完成本节如下习题。

**Q1.** 用初等行变换求解如下线性方程组，并绘制变换过程直线、线性组合变换。

► 
$$\begin{cases} x_1 + x_2 = 4 \\ 2x_1 + 4x_2 = 12 \end{cases}$$

► 
$$\begin{cases} x_1 + 2x_2 + 3x_3 = 1 \\ 3x_1 + x_3 = 2 \\ x_1 + 2x_2 + x_3 = 3 \end{cases}$$

► 
$$\begin{cases} x_1 + x_2 = 2 \\ x_1 - x_3 = 1 \\ x_2 - x_3 = 0 \end{cases}$$

**Q2.** 用初等行变换求解如下矩阵的逆。

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

► 
$$A = \begin{bmatrix} 1 & 2 & 3 \\ 3 & 0 & 1 \\ 1 & 2 & 1 \end{bmatrix}$$

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

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

**Q3.** 用初等行变换计算上一题矩阵的秩？

**Q4.** 什么是初等矩阵？

**Q5.** 什么是 LU 分解？

**Q6.** 自学用 SciPy 库函数完成 LU 分解？

<https://docs.scipy.org/doc/scipy/reference/generated/scipy.linalg.lu.html>

**Q7.** 什么是 LDL 分解？

**Q8.** 自学使用 Scipy 库函数完成 LDL 分解？

<https://docs.scipy.org/doc/scipy/reference/generated/scipy.linalg.ldl.html>