

23

Fundamentals of Linear Algebra

鸡兔同笼 1

之从《孙子算经》到线性代数



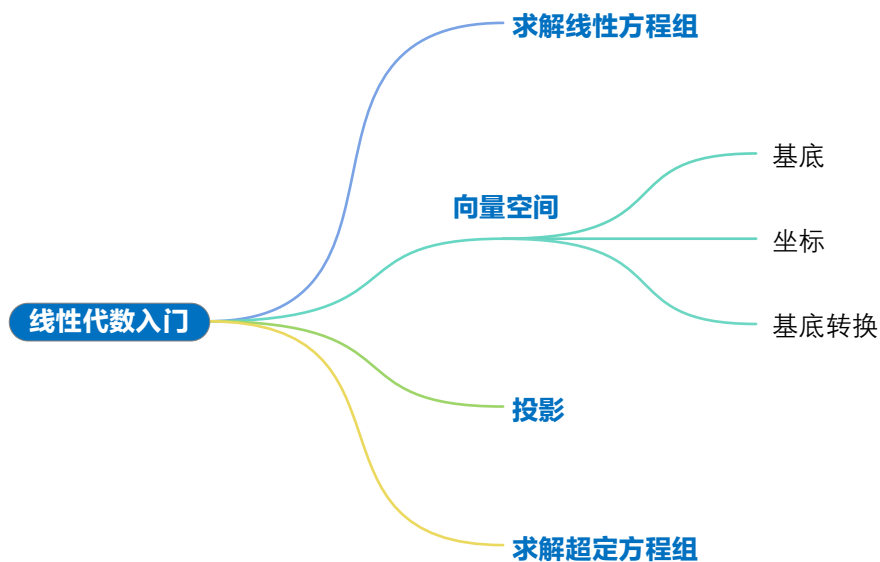
这就是数学。她提醒你无形灵魂的存在，她赋予数学发现以生命；她唤醒沉睡的心灵，她净化心智；她给思想以光辉。她涤荡与生俱来的蒙昧与无知。

This, therefore, is mathematics: she reminds you of the invisible form of the soul; she gives life to her own discoveries; she awakens the mind and purifies the intellect; she brings light to our intrinsic ideas; she abolishes oblivion and ignorance which are ours by birth.

—— 普罗克洛 (Proclus) | 古希腊哲学家 | 412 ~ 485



```
matplotlib.pyplot.quiver() 绘制箭头图
numpy.column_stack() 将两个矩阵按列合并
numpy.linalg.inv() 矩阵求逆
numpy.linalg.solve() 求解线性方程组
numpy.matrix() 创建矩阵
sympy.solve() 求解符号方程组
sympy.solvers.solveset.linsolve() 求解符号线性方程组
```



23.1 从鸡兔同笼说起

云山青青，凤泉冷冷，山色可爱，泉声可听。土地平旷，屋舍俨然，阡陌交通，鸡犬相闻。

崇山峻岭之中，茂林修竹深处，有五十余户人家。小村村民甘其食，美其服，安其居，乐其俗。黄发垂髫，怡然自乐。

村民善养鸡兔，又善筹算。在这个与世隔绝的小村庄，鸡兔同笼这样的经典数学问题，代代流传，深入人心。

本书最后三章给大家说说村民在养鸡养兔遇到的数学问题，讲讲如何用线性代数帮助大伙儿解决这些问题。

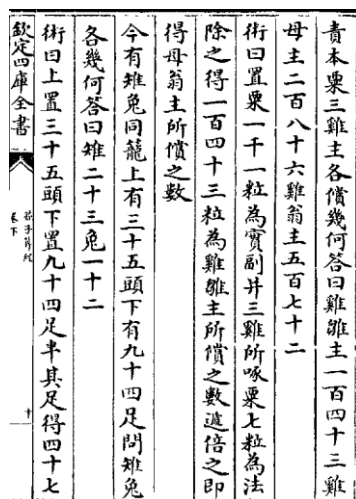


图 1. 《孙子算经》中的鸡兔同笼问题 (来源: <https://cnkgraph.com/>)

鸡兔同笼原题

《孙子算经》中鸡兔同笼问题这样说，“今有雉兔同笼，上有三十五头，下有九十四足，问雉兔各几何？”本书前文构造二元一次方程组，用代数方法解决鸡兔同笼问题：

$$\begin{cases} x_1 + x_2 = 35 \\ 2x_1 + 4x_2 = 94 \end{cases} \quad (1)$$

其中， x_1 为鸡数量， x_2 为兔数量。

求得笼子里有 23 只鸡，12 只兔：

$$\begin{cases} x_1 = 23 \\ x_2 = 12 \end{cases} \quad (2)$$

此外，本书之前也介绍过利用坐标系图解鸡兔同笼问题。

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

线性方程组

下面，我们看一下如何用本书前文学过的线性代数知识解决这个数学问题。

(1) 中第一个等式写成矩阵运算形式，得到：

$$1 \cdot x_1 + 1 \cdot x_2 = 35 \Rightarrow \begin{bmatrix} 1 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 35 \end{bmatrix} \quad (3)$$

(1) 第二个等式也写成类似形式：

$$2 \cdot x_1 + 4 \cdot x_2 = 94 \Rightarrow \begin{bmatrix} 2 & 4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 94 \end{bmatrix} \quad (4)$$

结合 (3) 和 (4)，我们使用矩阵形式写出了鸡兔同笼问题的线性方程组：

$$\begin{cases} 1 \cdot x_1 + 1 \cdot x_2 = 35 \\ 2 \cdot x_1 + 4 \cdot x_2 = 94 \end{cases} \Rightarrow \begin{bmatrix} 1 & 1 \\ 2 & 4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 35 \\ 94 \end{bmatrix} \quad (5)$$

(5) 可以写成：

$$Ax = b \quad (6)$$

其中，

$$A = \begin{bmatrix} 1 & 1 \\ 2 & 4 \end{bmatrix}, \quad x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}, \quad b = \begin{bmatrix} 35 \\ 94 \end{bmatrix} \quad (7)$$

x 是未知变量构成的列向量， A 为方阵且可逆， x 可以利用下式求得：

$$x = A^{-1}b \quad (8)$$

代入具体数值计算得到 x ：

$$x = \begin{bmatrix} 1 & 1 \\ 2 & 4 \end{bmatrix}^{-1} \begin{bmatrix} 35 \\ 94 \end{bmatrix} = \begin{bmatrix} 2 & -0.5 \\ -1 & 0.5 \end{bmatrix} \begin{bmatrix} 35 \\ 94 \end{bmatrix} = \begin{bmatrix} 23 \\ 12 \end{bmatrix} \quad (9)$$

以下代码完成上述运算。



```
# Bk3 Ch23 1

import numpy as np

A = np.array([[1,1],
              [2,4]])

b = np.array([[35],
              [94]])

A_inv = np.linalg.inv(A)
```

```
x = A_inv@b
print(x)


x_ = np.linalg.solve(A,b)
print(x_)

from sympy import *
x1, x2 = symbols(['x1', 'x2'])
sol = solve([x1 + x2 - 35, 2*x1 + 4*x2 - 94], [x1, x2])
print(sol)

from sympy.solvers.solveset import linsolve
sol_ = linsolve([x1 + x2 - 35, 2*x1 + 4*x2 - 94], [x1, x2])
print(sol_)
```

23.2 “鸡” 向量与 “兔” 向量

观察矩阵 A ，它是由两个列向量左右排列构造而成：

$$A = \begin{bmatrix} 1 & 1 \\ 2 & 4 \end{bmatrix} \begin{matrix} \text{Head} \\ \text{Feet} \end{matrix} \quad (10)$$


由此，矩阵 A 可以写成 a_1 和 a_2 两个左右排列的列向量：

$$[a_1 \ a_2] \quad (11)$$

让我们分析一下 a_1 和 a_2 这两个向量的具体含义。

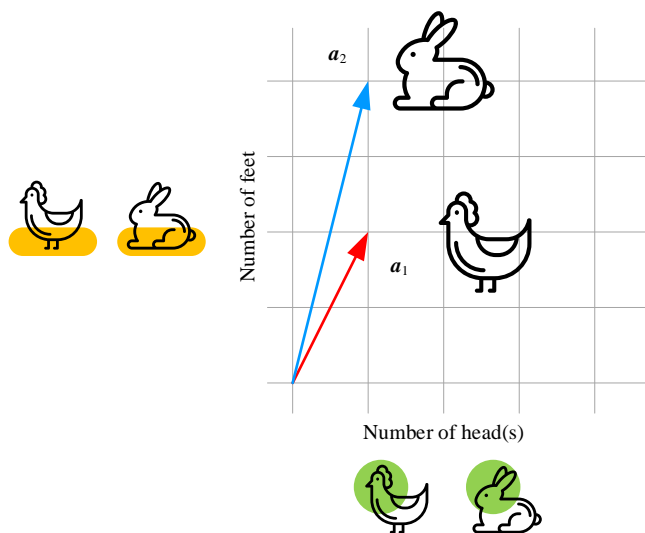


图 2. 鸡向量 a_1 和兔向量 a_2

a_1 代表一只鸡，特征是一个头、两只脚：

$$a_1 = \begin{bmatrix} \text{\# head} \\ 1 \\ \text{\# feet} \\ 2 \end{bmatrix} \quad (12)$$

a_2 代表一只兔，特征是有一个头、四只脚：

$$a_2 = \begin{bmatrix} \text{\# head} \\ 1 \\ \text{\# feet} \\ 4 \end{bmatrix} \quad (13)$$

图 2 所示为鸡向量 a_1 和兔向量 a_2 。图 2 中坐标轴的横轴为头的数量，纵轴为脚的数量。 e_1 代表“头”向量， e_2 代表“脚”向量。显然， e_1 是横轴单位向量， e_2 是纵轴单位向量。

分解

如图 3 所示，鸡向量 a_1 可以写成：

$$a_1 = \begin{bmatrix} \text{\# head} \\ 1 \\ \text{\# feet} \\ 2 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 2 \end{bmatrix} = \begin{bmatrix} e_1 & e_2 \end{bmatrix} \begin{bmatrix} 1 \\ 2 \end{bmatrix} = e_1 + 2e_2 \quad (14)$$

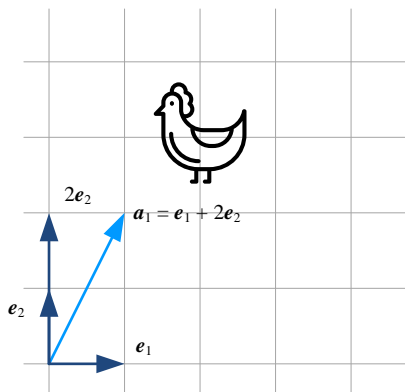
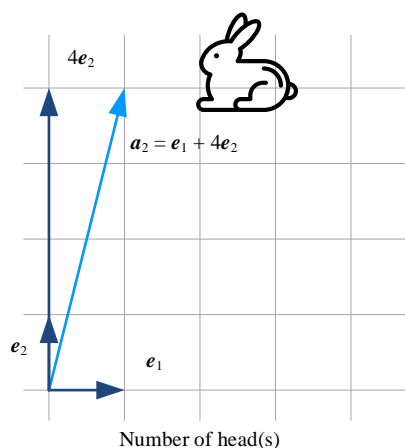


图 3. 鸡向量 a_1

如图 4 所示，兔向量 a_2 可以写成：

$$a_2 = \begin{bmatrix} \text{\# head} \\ 1 \\ \text{\# feet} \\ 4 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 4 \end{bmatrix} = \begin{bmatrix} e_1 & e_2 \end{bmatrix} \begin{bmatrix} 1 \\ 4 \end{bmatrix} = e_1 + 4e_2 \quad (15)$$

图 4. 兔向量 a_2

再谈鸡兔同笼

回到鸡兔同笼问题， x_1 代表鸡的数量， x_2 为兔的数量；将 $A = [a_1, a_2]$ 代入 (6)，得到：

$$\begin{bmatrix} 1 & 1 \\ 2 & 4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 35 \\ 94 \end{bmatrix} \quad \begin{matrix} \text{鸡} & \text{兔} \\ \text{鸡} & \text{兔} \end{matrix} \quad (16)$$

白话说，(16) 代表 x_1 份 a_1 和 x_2 份 a_2 组合，得到 b 向量。

为了方便可视化，将向量 b 改为如下具体值。也就是鸡兔同笼问题条件为——有 3 个头、8 只脚。线性方程组写成：

$$\begin{bmatrix} 1 & 1 \\ 2 & 4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 3 \\ 8 \end{bmatrix} \quad \begin{matrix} \text{鸡} & \text{兔} \\ \text{鸡} & \text{兔} \end{matrix} \quad (17)$$

我们思考这个问题， x 和 b 具体代表什么？

(17) 等式左边的列向量 $x = [x_1, x_2]^T$ 代表鸡兔数量，而 (17) 右侧 b 代表头、脚数量。

坐标系角度

从坐标系的角度来看， x 在“鸡-兔系”中，而 b 在“头-脚系”中。

图 5 左侧方格就是“头-脚系”，而图 5 右侧平行四边形网格便是“鸡-兔系”。

“头-脚系”中，“头”向量 e_1 和“脚”向量 e_2 ，张成了方格面。白话说，在“头-脚系”中，我们看到的是鸡兔的头和脚数。

“鸡-兔系”中，“鸡”向量 a_1 和“兔”向量 a_2 ，张成了平行四边形网格。在“鸡-兔系”中，我们关注的是鸡兔具体只数。

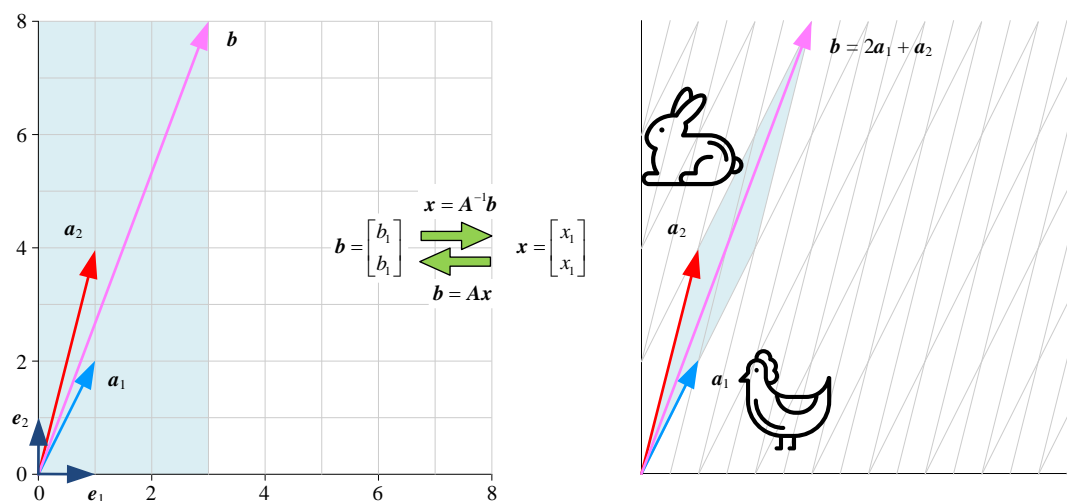


图 5. “头-脚系”和“鸡-兔系”相互转换

A 作为桥梁，完成从“鸡-兔系” x 向“头-脚系” b 转换。

$$x \rightarrow b: Ax = b \quad (18)$$

反方向来看， A^{-1} 完成“头-脚” b 向“鸡-兔” x 转换。

$$b \rightarrow x: A^{-1}b = x \quad (19)$$

以下代码绘制图 5。



```
# Bk3 Ch23 2
import numpy as np
import matplotlib.pyplot as plt

def draw_vector(vector, RGB):
    array = np.array([[0, 0], vector[0], vector[1]], dtype=object)
    X, Y, U, V = zip(*array)
    plt.quiver(X, Y, U, V, angles='xy', scale_units='xy', scale=1, color = RGB)

x1 = np.arange(-25, 25 + 1, step=1);
x2 = np.arange(-25, 25 + 1, step=1);

XX1, XX2 = np.meshgrid(x1, x2);

X = np.column_stack((XX1.ravel(), XX2.ravel()))

A = np.matrix([[1, 1],
               [2, 4]]);
```

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


```

Z = X@A.T;

ZZ1 = Z[:,0].reshape((len(x1), len(x2)))
ZZ2 = Z[:,1].reshape((len(x1), len(x2)))

### base: e1 and e2

fig, ax = plt.subplots()

plt.plot(XX1,XX2,color = [0.8,0.8,0.8])
plt.plot(XX1.T,XX2.T,color = [0.8,0.8,0.8])

a1 = A[:,0].tolist()
a2 = A[:,1].tolist()
b = [3,8]

draw_vector(a1, np.array([0,112,192])/255)
draw_vector(a2, np.array([255,0,0])/255)

draw_vector(b, np.array([255,125,255])/255)

plt.xlabel('$e_1$')
plt.ylabel('$e_2$')

plt.axis('scaled')
ax.set_xlim([0, 8])
ax.set_ylim([0, 8])

plt.xticks(np.arange(0, 8 + 1, step=2))
plt.yticks(np.arange(0, 8 + 1, step=2))

ax.spines['top'].set_visible(False)
ax.spines['right'].set_visible(False)

### base: a1 and a2

fig, ax = plt.subplots()

plt.plot(ZZ1,ZZ2,color = [0.8,0.8,0.8])
plt.plot(ZZ1.T,ZZ2.T,color = [0.8,0.8,0.8])

draw_vector(a1, np.array([0,112,192])/255)
draw_vector(a2, np.array([255,0,0])/255)

draw_vector(b, np.array([255,125,255])/255)

plt.axis('scaled')
ax.set_xlim([0, 8])
ax.set_ylim([0, 8])

plt.xticks(np.arange(0, 8 + 1, step=2))
plt.yticks(np.arange(0, 8 + 1, step=2))

ax.spines['top'].set_visible(False)
ax.spines['right'].set_visible(False)

```

23.3 那几只毛绒耳朵

农夫看了看同处一笼的鸡兔，突然发现在头、脚之外，赫然独立几只可爱至极的毛绒耳朵。

他突然想到，除了查头数、脚数之外，查毛绒耳朵的数量应该更容易确定兔子的数量！虽然，生理学角度，鸡也有耳朵，但是极不容易被发现。

加了毛绒耳朵这个特征之后，二维向量就变成了三维向量。

“鸡”向量 a_1 变为：

$$a_1 = \begin{bmatrix} 1 \\ 2 \\ 0 \end{bmatrix} \begin{matrix} \text{鸡头} \\ \text{鸡脚} \\ \text{鸡耳朵} \end{matrix} \quad (20)$$

“兔”向量 a_2 变为：

$$a_2 = \begin{bmatrix} 1 \\ 4 \\ 2 \end{bmatrix} \begin{matrix} \text{兔头} \\ \text{兔脚} \\ \text{兔耳朵} \end{matrix} \quad (21)$$

在平面直角坐标系中，升起第三个维度——毛绒耳朵数量，我们便得到如图 6 所示的三维直角坐标系。其中， e_3 代表“毛绒耳朵”向量。

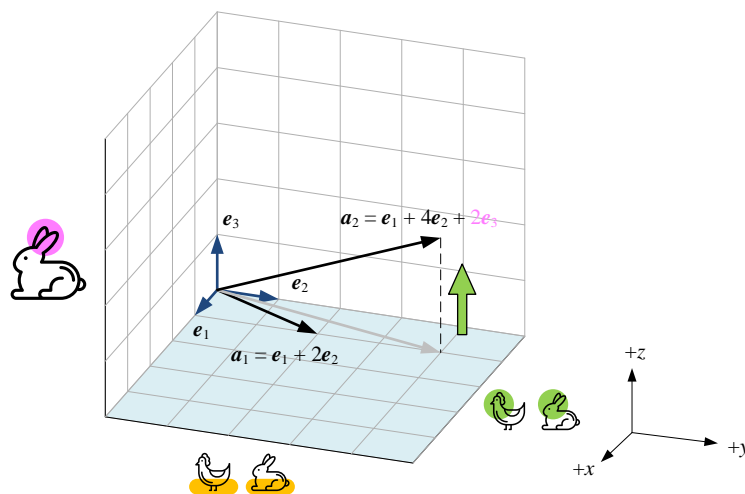


图 6. 三维直角坐标系中的鸡向量 a_1 和兔向量 a_2

图 6 中，一只鸡一个头、两只脚、没有毛绒耳朵，因此鸡向量 a_1 为：

$$a_1 = e_1 + 2e_2 \quad (22)$$

观察图 6，鸡向量 a_1 还“趴”在水平面上，这是因为鸡没有毛绒耳朵！

一只兔一个头、四只脚、两个毛绒耳朵， a_2 写成：

$$a_2 = e_1 + 4e_2 + 2e_3 \quad (23)$$

而兔向量 a_2 还已经“立”在水平面之外，就是因为那两只毛绒耳朵（撸撸）。

计算头、脚、毛绒耳朵数量

如果给定一笼鸡兔的鸡和兔的数量，让大家求解头、脚、毛绒耳朵数量，就是从“鸡-兔系”到“头-脚-毛绒耳朵系”的转化。

假设有鸡 10 只 (x_1)、兔 5 只 (x_2)，可以通过下式计算头、脚和毛绒耳朵数量：

$$\mathbf{b} = [\mathbf{a}_1 \quad \mathbf{a}_2] \mathbf{x} = \begin{bmatrix} 1 & 1 \\ 2 & 4 \\ 0 & 2 \end{bmatrix} \begin{bmatrix} 10 \\ 5 \end{bmatrix} = \begin{bmatrix} 15 \\ 40 \\ 10 \end{bmatrix} \quad (24)$$

这样，通过上述计算，我们便完成了从“鸡-兔系”到“头-脚-毛绒耳朵系”的转换。这个过程是从二维到三维，相当于“升维”。

23.4 “鸡兔”套餐

村子里来个小贩卖小鸡和小兔，但可惜不单独售卖。

小贩提供两种套餐捆绑销售：A 套餐，3 鸡 1 兔；B 套餐，1 鸡 2 兔。

这可难坏了老农，因为他想买 10 只鸡、10 只兔。该怎么组合 A、B 两种套餐？下面，我们也用线性代数知识来帮帮他。

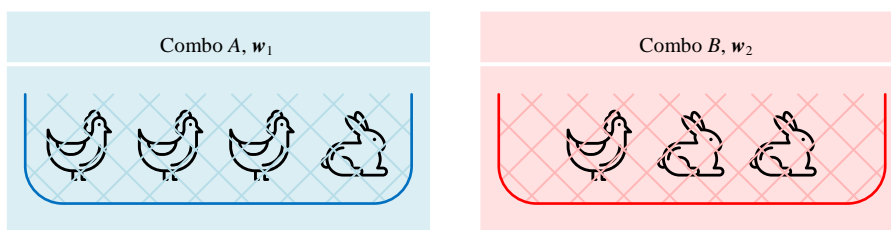


图 7. 鸡兔 A、B 套餐

A-B 套餐系

将 A、B 套餐记做列向量 \mathbf{w}_1 和 \mathbf{w}_2 ，具体取值如下：

$$\mathbf{w}_1 = \begin{bmatrix} 3 \\ 1 \end{bmatrix}, \quad \mathbf{w}_2 = \begin{bmatrix} 1 \\ 2 \end{bmatrix} \quad (25)$$

老农想买 10 只鸡、10 只兔，记做 \mathbf{a} ：

$$\mathbf{a} = \begin{bmatrix} 10 \\ 10 \end{bmatrix} \quad (26)$$

令所需套餐 A 的数量为 x_1 ，套餐 B 的数量为 x_2 ，构造如下等式：

$$x_1 \mathbf{w}_1 + x_2 \mathbf{w}_2 = x_1 \begin{bmatrix} 3 \\ 1 \end{bmatrix} + x_2 \begin{bmatrix} 1 \\ 2 \end{bmatrix} = \begin{bmatrix} 10 \\ 10 \end{bmatrix} \quad (27)$$

即：

$$\begin{bmatrix} 3 & 1 \\ 1 & 2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 10 \\ 10 \end{bmatrix} \quad (28)$$

如图 8 所示，向量 \mathbf{a} 在“鸡-兔系”到“A-B 套餐系”的不同意义。

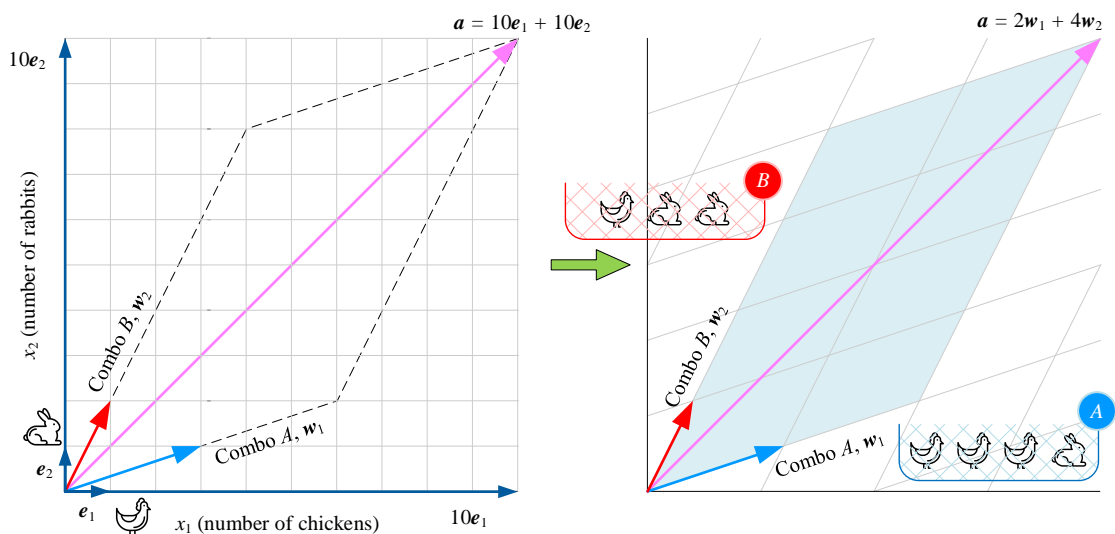


图 8. 向量 \mathbf{a} 在“鸡-兔系”到“A-B 套餐系”的不同意义

这样求得向量 \mathbf{x} 为。

$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 3 & 1 \\ 1 & 2 \end{bmatrix}^{-1} @ \begin{bmatrix} 10 \\ 10 \end{bmatrix} = \begin{bmatrix} 0.4 & -0.2 \\ -0.2 & 0.6 \end{bmatrix} @ \begin{bmatrix} 10 \\ 10 \end{bmatrix} = \begin{bmatrix} 2 \\ 4 \end{bmatrix} \quad (29)$$

线性组合

也就是说，农夫可以买 2 份 A 套餐、4 份 B 套餐，这样一共买到 10 只鸡、10 只兔，对应算式为。

$$2\mathbf{w}_1 + 4\mathbf{w}_2 = 2 \times \begin{bmatrix} 3 \\ 1 \end{bmatrix} + 4 \times \begin{bmatrix} 1 \\ 2 \end{bmatrix} = \begin{bmatrix} 10 \\ 10 \end{bmatrix} \quad (30)$$

(30) 这个等式就叫线性组合 (linear combination)。我们管 \mathbf{w}_1 和 \mathbf{w}_2 叫做基底 (basis)，写成 $\{\mathbf{w}_1, \mathbf{w}_2\}$ 。也就是说，图 9 左图的基底为 $\{\mathbf{a}_1, \mathbf{a}_2\}$ ，右图的基底为 $\{\mathbf{w}_1, \mathbf{w}_2\}$ 。

白话说，就是用 2 份 w_1 向量、4 份 w_2 向量混合得到新的向量。通过线性组合的向量仍在平面之内。

如图 9 所示，如果只看网格的话，上述数学运算完成了“鸡-兔系”到“A-B 套餐系”的坐标系转化。

(10, 10) 是向量 a 在“鸡-兔系”的坐标。

而 2 份 A 套餐、4 份 B 套餐相当于 (2, 4) 是向量 a 在“A-B 套餐系”的坐标。

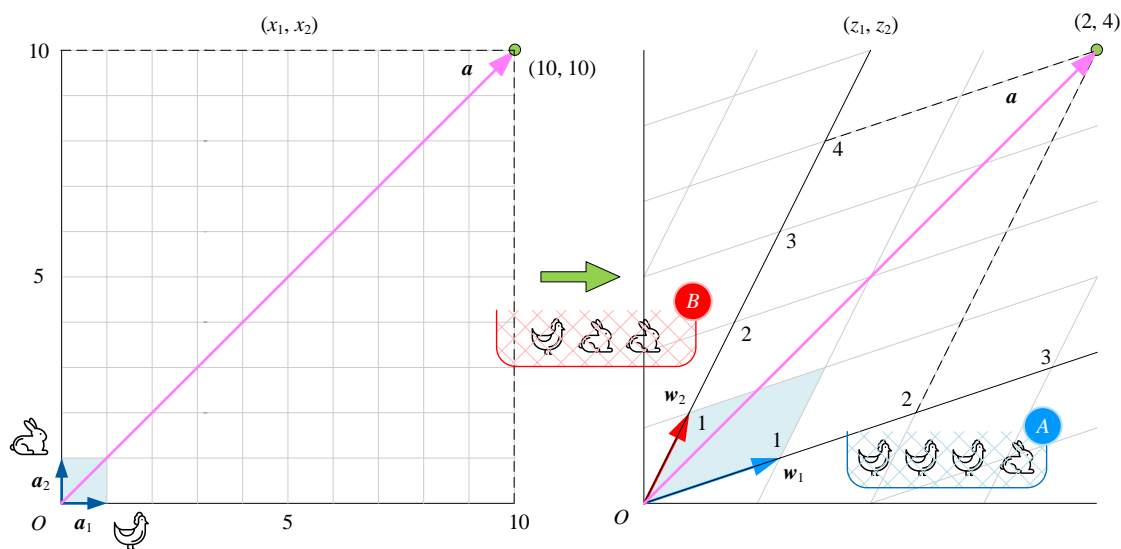


图 9. 坐标系转换，“鸡-兔系”到“A-B 套餐系”

基底变换

不管是从“头、脚系”到“鸡-兔系”，还是从“鸡-兔系”到“A-B 套餐系”，都叫做基底变换 (change of basis)。

对于向量 a ，在基底 $\{a_1, a_2\}$ 下，坐标值为 $[x_1, x_2]^T$ ：

$$a = x = Ix = \begin{bmatrix} a_1 & a_2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = x_1 a_1 + x_2 a_2 \quad (31)$$

也就是：

$$a = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 10 \\ 10 \end{bmatrix} = \begin{bmatrix} 10 \\ 10 \end{bmatrix} \quad (32)$$

同一个向量 a ，在基底 $\{w_1, w_2\}$ 下，坐标值为 $[z_1, z_2]^T$ ：

$$\boldsymbol{a} = \boldsymbol{W}\boldsymbol{z} = \underbrace{\begin{bmatrix} \boldsymbol{w}_1 & \boldsymbol{w}_2 \end{bmatrix}}_{\boldsymbol{W}} \underbrace{\begin{bmatrix} z_1 \\ z_2 \end{bmatrix}}_{\boldsymbol{z}} = z_1 \boldsymbol{w}_1 + z_2 \boldsymbol{w}_2 \quad (33)$$

即：

$$\boldsymbol{a} = \underbrace{\begin{bmatrix} 3 & 1 \\ 1 & 2 \end{bmatrix}}_{\boldsymbol{W}} \underbrace{\begin{bmatrix} 2 \\ 4 \end{bmatrix}}_{\boldsymbol{z}} = \begin{bmatrix} 10 \\ 10 \end{bmatrix} \quad (34)$$

联立 (31) 和 (33) 得到：

$$\boldsymbol{x} = \boldsymbol{W}\boldsymbol{z} \quad (35)$$

即

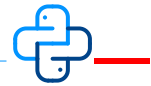
$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \underbrace{\begin{bmatrix} w_1 & w_2 \end{bmatrix}}_{\boldsymbol{W}} \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} \quad (36)$$

新坐标 \boldsymbol{z} ，可以通过下式得到。

$$\boldsymbol{z} = \boldsymbol{W}^{-1}\boldsymbol{x} \quad (37)$$

也就是说， \boldsymbol{W} 是新旧坐标转换的桥梁。如图 9 所示，转换前后，网格形状发生变化，但是平面还是那个平面。

以下代码绘制本节两个坐标系网格图像。



```
# Bk3_Ch23_3

import numpy as np
import matplotlib.pyplot as plt

def draw_vector(vector, RGB):
    array = np.array([[0, 0, vector[0], vector[1]]])
    X, Y, U, V = zip(*array)
    plt.quiver(X, Y, U, V, angles='xy', scale_units='xy', scale=1, color = RGB)

x1 = np.arange(-25, 25 + 1, step=1);
x2 = np.arange(-25, 25 + 1, step=1);

XX1, XX2 = np.meshgrid(x1, x2);

X = np.column_stack((XX1.ravel(), XX2.ravel()))

R = np.matrix('1,3;2,1');

Z = X @ R.T;

ZZ1 = Z[:,0].reshape((len(x1), len(x2)))
ZZ2 = Z[:,1].reshape((len(x1), len(x2)))

%%
fig, ax = plt.subplots()

plt.plot(XX1, XX2, color = [0.8, 0.8, 0.8])
```

```
plt.plot(XX1.T,XX2.T,color = [0.8,0.8,0.8])
draw_vector([1,2],np.array([0,112,192])/255)
draw_vector([3,1],np.array([255,0,0])/255)

draw_vector([10,10], '#FF99FF')

plt.xlabel('$x_1$ (number of chickens)')
plt.ylabel('$x_2$ (number of rabbits)')

plt.axis('scaled')
ax.set_xlim([0, 10])
ax.set_ylim([0, 10])

plt.xticks(np.arange(0, 10 + 1, step=2))
plt.yticks(np.arange(0, 10 + 1, step=2))

ax.spines['top'].set_visible(False)
ax.spines['right'].set_visible(False)

%%

fig, ax = plt.subplots()

plt.plot(ZZ1,ZZ2,color = [0.8,0.8,0.8])
plt.plot(ZZ1.T,ZZ2.T,color = [0.8,0.8,0.8])

draw_vector([1,2],np.array([0,112,192])/255)
draw_vector([3,1],np.array([255,0,0])/255)

draw_vector([10,10], '#FF99FF')

plt.xlabel('$z_1$ (combo A)')
plt.ylabel('$z_2$ (combo B)')

plt.axis('scaled')
ax.set_xlim([0, 10])
ax.set_ylim([0, 10])

plt.xticks(())
plt.yticks(())

ax.spines['top'].set_visible(False)
ax.spines['right'].set_visible(False)
```

23.5 套餐转换：基底转换

前来买鸡兔的村民在小贩周围越聚越多，大家都说套餐 A 和 B 组合太繁琐，纷纷抱怨。

为了方便村民买鸡兔，小贩推出两个新套餐 C 和 D ：套餐 C ，两只小鸡；套餐 D ，两只小兔。也就是，鸡兔都是成对販售。

用我们刚刚学过的基底转换思路来看看这个新基底。

令第三个基底 $\{v_1, v_2\}$ 代表“ C - D 套餐系”。在基底 $\{v_1, v_2\}$ 中，向量 a 可以写成。

$$a = Vs = \underbrace{\begin{bmatrix} v_1 & v_2 \end{bmatrix}}_V \begin{bmatrix} s_1 \\ s_2 \end{bmatrix} = s_1 v_1 + s_2 v_2 \quad (38)$$

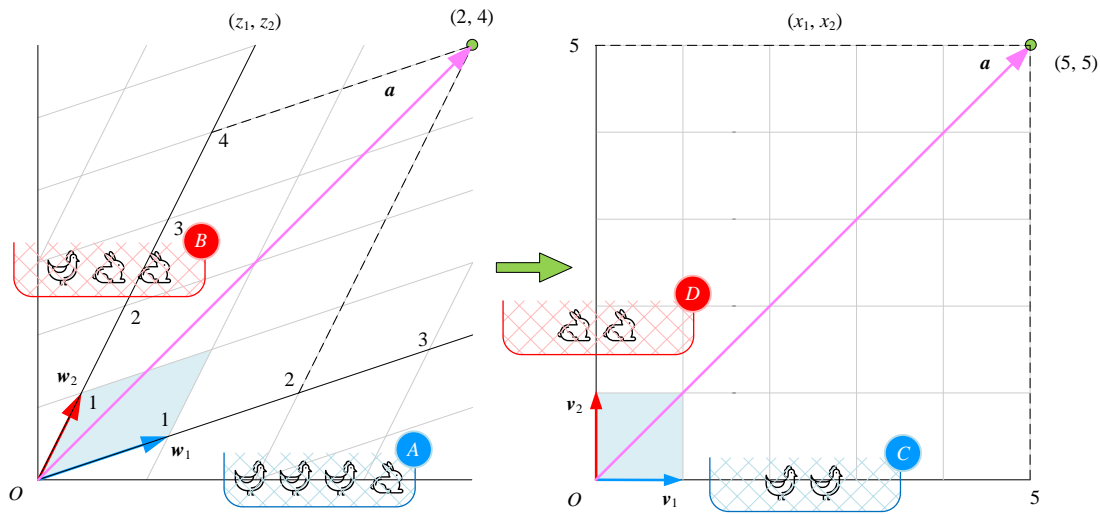


图 10. 套餐转换，“A-B 套餐系”到“C-D 套餐系”

联立 (33) 和 (38)，得到。

$$Wz = Vs \quad (39)$$

也就是说， s 可以通过下式得到。

$$s = V^{-1}Wz \quad (40)$$

而 V 为：

$$V = \underbrace{\begin{bmatrix} v_1 & v_2 \end{bmatrix}}_V = \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix} \quad (41)$$

这样，向量 a 从 $\{w_1, w_2\}$ 基底到 $\{v_1, v_2\}$ 基底，新坐标 s 为：

$$s = \underbrace{\begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix}}_V^{-1} \underbrace{\begin{bmatrix} 3 & 1 \\ 1 & 2 \end{bmatrix}}_W \underbrace{\begin{bmatrix} 2 \\ 4 \end{bmatrix}}_z = \begin{bmatrix} 5 \\ 5 \end{bmatrix} \quad (42)$$

也就是说，老农想要买 10 只鸡、10 只兔的话，需要 5 份套餐 C 和 5 份套餐 D。

23.6 猪引发的投影问题

农夫突然改了主意，他对小贩说，我想买 10 只鸡、10 只兔，还要买 5 只猪！

小贩很无奈，说小猪早就卖断货了。

老农略有所思，说了句，“我和你之间，存在 5 只猪的距离。”

从向量角度，农夫想买 10 只鸡、10 只兔、5 只猪，可以写成向量 y ：

$$y = \begin{bmatrix} 10 \\ 10 \\ 5 \end{bmatrix} \quad (43)$$

然而，小贩提供的“A-B 套餐”只能满足农夫部分需求，记做向量 a ：

$$a = x_1 w_1 + x_2 w_2 = x_1 \begin{bmatrix} 3 \\ 1 \\ 0 \end{bmatrix} + x_2 \begin{bmatrix} 1 \\ 2 \\ 0 \end{bmatrix} = \begin{bmatrix} 10 \\ 10 \\ 0 \end{bmatrix} \quad (44)$$

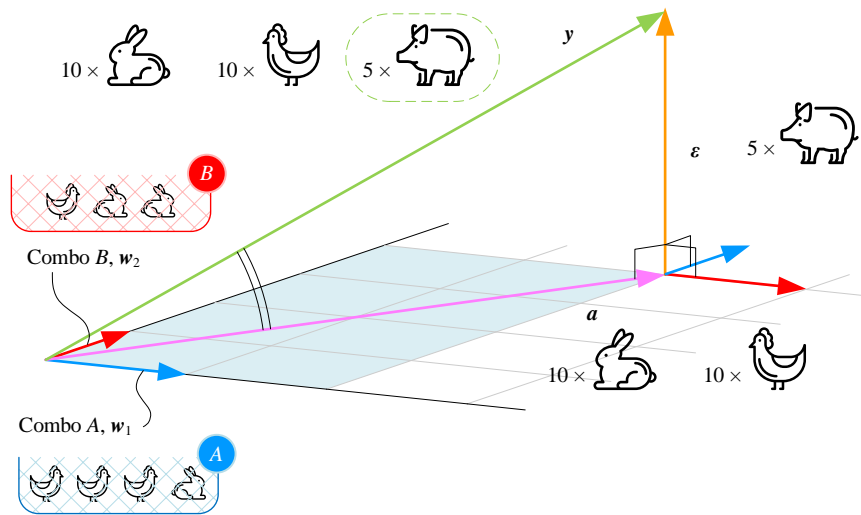


图 11. 老农的需求和小贩提供的“A-B 套餐”平面存在 5 只猪的距离

农夫的需求 y 和 a 的差距记做 ε ，计算得到具体值：

$$\varepsilon = y - a = \begin{bmatrix} 10 \\ 10 \\ 5 \end{bmatrix} - \begin{bmatrix} 10 \\ 10 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 5 \end{bmatrix} \quad (45)$$

垂直

如图 11 所示，容易发现 ε 垂直于 w_1 、 w_2 、 a ；下面，利用向量内积证明一下。

首先， ε 垂直于 w_1 ：

$$\mathbf{w}_1 \cdot \boldsymbol{\varepsilon} = \begin{bmatrix} 3 \\ 1 \\ 0 \end{bmatrix} \cdot \begin{bmatrix} 0 \\ 0 \\ 5 \end{bmatrix} = 3 \times 0 + 1 \times 0 + 0 \times 5 = 0 \quad (46)$$

$\boldsymbol{\varepsilon}$ 垂直于 \mathbf{w}_2 :

$$\mathbf{w}_2 \cdot \boldsymbol{\varepsilon} = \begin{bmatrix} 1 \\ 2 \\ 0 \end{bmatrix} \cdot \begin{bmatrix} 0 \\ 0 \\ 5 \end{bmatrix} = 1 \times 0 + 2 \times 0 + 0 \times 5 = 0 \quad (47)$$

$\boldsymbol{\varepsilon}$ 垂直于 \mathbf{a} :

$$\mathbf{a} \cdot \boldsymbol{\varepsilon} = \begin{bmatrix} 10 \\ 10 \\ 0 \end{bmatrix} \cdot \begin{bmatrix} 0 \\ 0 \\ 5 \end{bmatrix} = 10 \times 0 + 10 \times 0 + 0 \times 5 = 0 \quad (48)$$

也就是说, $\boldsymbol{\varepsilon}$ 垂直于 \mathbf{w}_1 和 \mathbf{w}_2 张成的平面。

从投影的角度来看, 向量 \mathbf{y} 在“A-B 套餐”平面的投影为 \mathbf{a} 。

23.7 “鸡飞兔脱”与超定方程组

夜黑风高, 农夫突然听到鸡叫犬吠!

他赶紧捡了件衣服披在身上, 提起油灯、夺门而出。在赶去鸡窝路上, 他发现了黄鼠狼的脚印。农夫慌忙跑到鸡兔窝, 看到鸡飞兔跳、惊慌失措。

担心黄鼠狼抓走了鸡兔, 农夫心急如焚, 他举高油灯, 凑近笼子, 数了又数。几遍下来, 数字都对不上, 自己更是头晕眼花。

他找来隔壁的甲、乙、丙、丁四人, 让甲、乙数头, 让丙、丁数脚。过了一阵, 甲说有 30 个头, 乙说有 35 个头; 丙说有 90 只脚, 丁说有 110 只脚。

这可难坏了农夫, 他可怎么估算鸡兔各自的数量?

用线性代数这个工具, 我们帮他试试看。

先列出来方程组:

$$\begin{cases} x_1 + x_2 = 30 \\ x_1 + x_2 = 35 \\ 2x_1 + 4x_2 = 90 \\ 2x_1 + 4x_2 = 110 \end{cases} \quad (49)$$

首先拿出图解法这个利器!

图 12 所示为四条直线对应的图像, 发现它们一共存在 4 个交点, 没有一组确切解。

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

代数角度，上述方程组叫做超定方程组 (overdetermined system)。两个方程两个未知数，显然所需的方程组远超未知数数量。

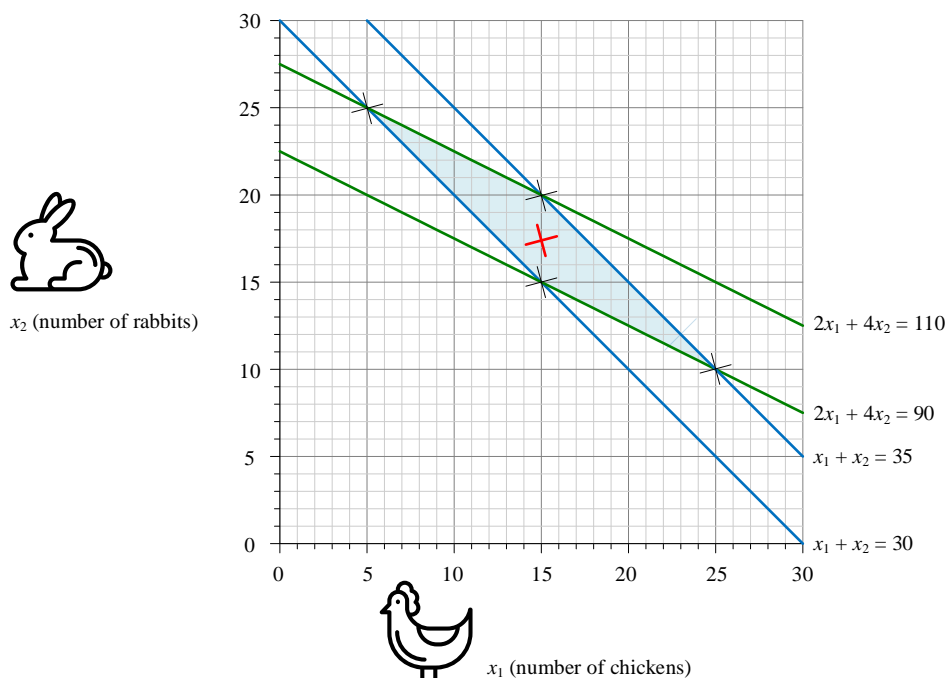


图 12. 超定方程组图像

将 (49) 写成矩阵的形式：

$$\underbrace{\begin{bmatrix} 1 & 1 \\ 1 & 1 \\ 2 & 4 \\ 2 & 4 \end{bmatrix}}_A \underbrace{\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}}_x = \underbrace{\begin{bmatrix} 30 \\ 35 \\ 90 \\ 110 \end{bmatrix}}_b \quad (50)$$

也就是：

$$Ax = b \quad (51)$$

A 不是方阵，显然不存在逆。

于是，我们采用一个全新的解法；(51) 左右分别乘 A^T 。

$$A^T Ax = A^T b \quad (52)$$

$A^T A$ 为 2×2 方阵，存在逆。

这样，(52) 可以整理为。

$$\mathbf{x} = (\mathbf{A}^T \mathbf{A})^{-1} \mathbf{A}^T \mathbf{b} \quad (53)$$

代入具体值，得到 \mathbf{x} 的估算解。

$$\mathbf{x} = \left(\begin{bmatrix} 1 & 1 \\ 1 & 1 \\ 2 & 4 \\ 2 & 4 \end{bmatrix}^T \begin{bmatrix} 1 & 1 \\ 1 & 1 \\ 2 & 4 \\ 2 & 4 \end{bmatrix} \right)^{-1} \begin{bmatrix} 1 & 1 \\ 1 & 1 \\ 2 & 4 \\ 2 & 4 \end{bmatrix}^T \begin{bmatrix} 30 \\ 35 \\ 90 \\ 110 \end{bmatrix} = \begin{bmatrix} 10 & 18 \\ 18 & 34 \end{bmatrix}^{-1} \begin{bmatrix} 465 \\ 865 \end{bmatrix} = \begin{bmatrix} 15 \\ 17.5 \end{bmatrix} \quad (54)$$

发现这个解恰好在图 12 四个交点构成平行四边形的中心位置。

以下代码完成上述矩阵运算。



```
# Bk3 Ch23 4
import numpy as np

A = np.array([[1,1],
              [1,1],
              [2,4],
              [2,4]])

b = np.array([[30],
              [35],
              [90],
              [110]])

x = np.linalg.inv(A.T@A)@A.T@b

print(x)
```



明月松间照，清泉石上流。微风丝丝缕缕，细雨点点滴滴。

微风夹着细雨，掠过田间地头，摇晃着杨柳梢，吹洗一池荷花。杨柳依依，荷风香气。微风轻轻悄悄地划过鸡舍兔笼，舞动着跳跃的烛火。它看了一眼月下独酌的农夫，踮着脚尖走过睡熟的牧童。

折腾了一天一夜，小村可算安静下来。殊不知，远处一场风暴正在酝酿。

未完待续。