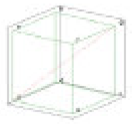

## 📕 Table of Contents



```
begin
        using PlutoUI   , Plots   ,DataFrames   ,HypertextLiteral   ,LaTeXStrings
,Symbolics   ,LinearAlgebra   ,RowEchelon
        gr()
        theme(:bright)
        @htl("""<script src="https://cdn.bootcdn.net/ajax/libs/mathjax/3.2.0/es5/tex-
svg-full.min.js"></script>
        """)
        PlutoUI.TableOfContents()
 end
```

# ch01 sec1.4 高斯消元法

**Outcomes**

- A 寻找矩阵的阶梯形式
- B 从矩阵阶梯形式判断方程组的解
- C 使用高斯消元和回带法解方程组
- D 求矩阵的秩
- E 通过矩阵的秩来判断解集情况

```
md"""
# ch01 sec1.4  高斯消元法

!!! outcomes

    - A 寻找矩阵的阶梯形式
    - B 从矩阵阶梯形式判断方程组的解
    - C 使用高斯消元和回带法解方程组
    - D 求矩阵的秩
    - E 通过矩阵的秩来判断解集情况

"""
```

Definition

阶梯型的定义

在增广矩阵中, 每一行的从最左边开始的的第一个非$0$ 项就称为**先导项**或者**主元项**. 具备如下条件的增广矩阵称为阶梯型矩阵:

1. 所在行元素都为$0$ 的行都在排在矩阵最下方
2. 每一行的主元一定位于上一行主元的右下方.

含有主元的列称为主元列.

关于第二条,这也就是被称为阶梯型的原因. 如果每一行都有主元, 主元会像楼梯一样向下排列.

用图说明更为直观

```
md"""
!!! definition

    阶梯型的定义

    在增广矩阵中, 每一行的从最左边开始的的第一个非$0$ 项就称为**先导项**或者**主元项**. 具备如下条件的增广矩阵称为阶梯型矩阵:

    1. 所在行元素都为$0$ 的行都在排在矩阵最下方
    2. 每一行的主元一定位于上一行主元的右下方.

    含有主元的列称为主元列.

    关于第二条,这也就是被称为阶梯型的原因. 如果每一行都有主元, 主元会像楼梯一样向下排列.

用图说明更为直观
"""
```

Example

example 1 下面的矩阵都是阶梯型的

$$\left[\begin{array}{ccccc|c} 0&5&2&1&3&5\\ 0&0&0&0&1&6\\ 0&0&0&0&0&0\\ 0&0&0&0&0&0 \end{array}\right]$$

$$\left[\begin{array}{ccccc|c} 1&4&0&0&5&0\\ 0&0&1&0&2&0\\ 0&0&0&1&4&0\\ 0&0&0&0&0&1 \end{array}\right]$$

$$\left[\begin{array}{ccc|c} 3&0&6&2\\ 0&1&4&0\\ 0&0&2&1\\ 0&0&0&0 \end{array}\right]$$

```
md"""
!!! example
    example 1
    下面的矩阵都是阶梯型的

    $\left[
    \begin{array}{ccccc|c}
      0&❺&2&1&3&5\\
      0&0&0&0&❶&6\\
      0&0&0&0&0&0\\
      0&0&0&0&0&0
    \end{array}
    \right]$

    $\left[
    \begin{array}{ccccc|c}
      ❶&4&0&0&5&0\\
      0&0&❶&0&2&0\\
      0&0&0&❶&4&0\\
      0&0&0&0&0&❶
    \end{array}
    \right]$

    $\left[
    \begin{array}{ccc|c}
      ❸&0&6&2\\
      0&❶&4&0\\
      0&0&❷&1\\
      0&0&0&0\\
    \end{array}
    \right]$

"""
```

Example

example 2 下面的矩阵不是阶梯型

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

$$\left[\begin{array}{cc|c} 1&2&3\\ 2&4&-6\\ 4&0&7 \end{array}\right]$$

$$\left[\begin{array}{ccc|c} 0&2&3&3\\ 1&5&0&2\\ 0&0&1&0\\ 0&0&0&0 \end{array}\right]$$

第一个矩阵0 行出现在最上方, 第二,第三个矩阵, 主元没有出现在最右方.

这些不符合阶梯型的矩阵都可以通过行变换变为阶梯型矩阵.

把矩阵变换为阶梯型矩阵就是高斯消元法要做的工作

```
md"""
!!! example
    example 2
    下面的矩阵不是阶梯型

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

    $\left[
    \begin{array}{cc|c}
      ❶&2&3\\
      ❷&4&-6\\
      ❹&0&7
    \end{array}
     \right]$

    $\left[
    \begin{array}{ccc|c}
      0&❷&3&3\\
      ❶&5&0&2\\
      0&0&❶&0\\
```

```
      0&0&0&0\\
    \end{array}
    \right]$


第一个矩阵$0$ 行出现在最上方, 第二,第三个矩阵, 主元没有出现在最右方.

这些不符合阶梯型的矩阵都可以通过行变换变为阶梯型矩阵.

把矩阵变换为阶梯型矩阵就是高斯消元法要做的工作

"""
```

**Algorithm**

高斯消元法

1. 从矩阵最左边开始, 找到第一个非0 的元素, 作为主元, 如果第一元素为0,通过行交换获取第一个主元
2. 通过行操作使得主元列中比主元位置低其他位置元素都为0
3. 完成上面 1,2 步之后, 在第二行继续执行同样的操作,遍历完整个矩阵,确保找到所有的主元

```
md"""
!!! algorithm
    高斯消元法
    1. 从矩阵最左边开始, 找到第一个非$0$ 的元素, 作为主元, 如果第一元素为$0$,通过行交换获取第一个主元
    2. 通过行操作使得主元列中比主元位置低其他位置元素都为$0$
    3. 完成上面 1,2 步之后, 在第二行继续执行同样的操作,遍历完整个矩阵,确保找到所有的主元
"""
```

**Example**

example 3 解下列方程组

$$x+4y+3z=11$$

$$2x+10y+7z=27$$

$$x+y+2z=5$$

```
md"""
!!! example
    example 3
    解下列方程组

    $x+4y+3z=11$
    $2x+10y+7z=27$
    $x+y+2z=5$
"""
```

提出系数和常数项组成增广矩阵

```
3×4 Matrix{Int64}:
 1   4  3  11
 2  10  7  27
 1   1  2   5
```

```
store["augmentmatrix"]
```

由于第一列第一个元素为1 非0,所以可以作为第一个主元,下面执行操作找到第二行主元, 第二行的主元如果存在, 必须位于第二行第二列的位置. 第二行第一列的元素必须为0. 第二行, 第三行的第一个元素也必须为0

所以换元操作就是要完成这几项任务

```
md"""
由于第一列第一个元素为$1$ 非$0$,所以可以作为第一个主元,下面执行操作找到第二行主元, 第二行的主元如果存在, 必须位于第二行第二列的位置. 第二行第一列的元素必须为$0$. 第二行,第三行的第一个元素也必须为$0$

所以换元操作就是要完成这几项任务

"""
```

```
3×4 Matrix{Int64}:
 1   4  3  11
 2  10  7  27
 1   1  2   5
```

```
store["augmentmatrix"]
```

1. 第二行减去两倍的第一行得到

```
md"1. 第二行减去两倍的第一行得到"
```

```
3×4 Matrix{Int64}:
 1  4  3  11
 0  2  1   5
 1  1  2   5
```

```
store["rowchange1"]
```

2. 第三行减去第一行得到:

```
3×4 Matrix{Int64}:
 1   4   3  11
 0   2   1   5
 0  -3  -1  -6
```

```
store["rowchange2"]
```

3. 第三行乘以 2

```
3×4 Matrix{Int64}:
 1   4   3   11
 0   2   1    5
 0  -6  -2  -12
```

4. 3倍第二行加到第三行

```
3×4 Matrix{Int64}:
 1  4  3  11
 0  2  1   5
 0  0  1   3
```

```
store["rowchange4"]
```

现在矩阵的形式就是阶梯型.方程就变形为:

$$x+4y+3z=11①$$

$$2y+z=5②$$

$$z=3③$$

所以$z=3$, 回带入②式, $2y+3=5$, 所以$y=1$, $z=3, y=1$ 回带入① 式, $x+4(1)+3(3)=11$,所以 $x=-2$

讲解带入原方程组检验结果如下:

```
md"""

现在矩阵的形式就是阶梯型.方程就变形为:

$x+4y+3z=11 ①$
$2y+z=5 ②$
$z=3 ③$

所以$z=3$, 回带入$②$ 式, $2y+3=5$, 所以$y=1$, $z=3,y=1$ 回带入$①$ 式, $x+4(1)+3(3)=11$,所以 $x=-2$

讲解带入原方程组检验结果如下:
"""
```

```
[true, true, true]
```

```
store["example3check"]
```

```
[true, true, true]
```

```
let
    @variables x,y,z
    r1=[1 4 3 11]
    r2=[2 10 7 27]
    r3=[1 1 2 5]
    augmentmatrix=[r1;r2;r3]
    row,col = size(augmentmatrix)
    save("augmentmatrix",augmentmatrix)

    r2=r2-2*r1
    m1=[r1;r2;r3]
    save("rowchange1",m1)
    r3=r3-r1
    m2=[r1;r2;r3]
    save("rowchange2",m2)
    r3=2*r3
    m3=[r1;r2;r3]
    save("rowchange3",m3)
    r3=r3+3*r2
    m4=[r1;r2;r3]
    save("rowchange4",m4)

    function f1(sol)
        x,y,z=sol[1],sol[2],sol[3]
        return   1x+4y+3z
    end
    function f2(sol)
        x,y,z=sol[1],sol[2],sol[3]
        return   2x+10y+7z
    end

    function f3(sol)
        x,y,z=sol[1],sol[2],sol[3]
        return   x+y+2z
    end


    es=[f1,f2,f3]
    sol=[-2,1,3]
    function check(es,sol,matrix)
       row,col=size(matrix)
       res=[]
       for  i in 1:row
           check=es[i](sol)==matrix[i,col]
           push!(res,check)
       end
       return  res
    end

    save("example3check",check(es,sol,augmentmatrix))





end
```

Example

example 4

下面的方程组是无解的, 在寻找主元的过程中可以发现无解的原因:

$$y + 2z = 2$$

$$2x + y - 2z = 3$$

$$4x - y - 10z = 4$$

增广矩阵为:

```
3×4 Matrix{Int64}:
 0   1    2  2
 2   1   -2  3
 4  -1  -10  4
```

1. 由于第一行第一列元素为0,所以首先首先换行,交换第一行和第二行

```
3×4 Matrix{Int64}:
 2   1   -2  3
 0   1    2  2
 4  -1  -10  4
```

```
store["m4rowchange1"]
```

2. 第三行减去 2 倍的第一行

```
3×4 Matrix{Int64}:
 2   1  -2   3
 0   1   2   2
 0  -3  -6  -2
```

```
store["m4rowchange2"]
```

3. 第三行加上 3倍的第二行

```
3×4 Matrix{Int64}:
 2  1  -2  3
 0  1   2  2
 0  0   0  4
```

```
store["m4rowchange3"]
```

现在,矩阵已经是阶梯型, 但是最后一行的方程变换为:

$$0x+0y+0z=4$$

不符合数学逻辑的, 所以方程组无解. 这是短路条件, 一旦不符合条件,其他的判断就不用在进行

```
md"""
现在,矩阵已经是阶梯型, 但是最后一行的方程变换为:

$0x+0y+0z=4$

不符合数学逻辑的, 所以方程组无解.  这是短路条件, 一旦不符合条件,其他的判断就不用在进行
"""
```

```
3×4 Matrix{Int64}:
 2  1  -2  3
 0  1   2  2
 0  0   0  4
```

```
let
    @variables x,y,z
    r1=[0 1 2 2]
    r2=[2 1 -2 3]
    r3=[4 -1 -10 4]
    augmentmatrix4=[r1;r2;r3]
    row,col = size(augmentmatrix4)
    save("augmentmatrix4",augmentmatrix4)
    r1,r2=r2,r1
    matrix=[r1;r2;r3]
    save("m4rowchange1",matrix)
    r3=r3-2*r1
    matrix=[r1;r2;r3]
    save("m4rowchange2",matrix)
    r3=r3+3*r2
    matrix=[r1;r2;r3]
    save("m4rowchange3",matrix)
end
```

Example

example 4: 有无限个解的方程组

$$3x - y + 5z = 8$$

$$y - 10z = 1$$

$$6x - y = 17$$

增广矩阵为:

```
md"""
!!! example
    example 4: 有无限个解的方程组

    $3x-y+5z=8$
    $y-10z=1$
    $6x-y=17$

增广矩阵为:
"""
```

```
3×4 Matrix{Int64}:
 3  -1    5   8
 0   1  -10   1
 6  -1    0  17
```

1. 第一行已经有主元, 第二行第一列为 0, 所有通过行操作讲第三行第一一列元素变为 0, 第三行减去 2 倍的第二行得到:

```
md"1. 第一行已经有主元, 第二行第一列为 0, 所有通过行操作讲第三行第一一列元素变为 0,第三行减去 2 倍的第二行得到:"
```

```
3×4 Matrix{Int64}:
 3  -1    5  8
 0   1  -10  1
 0   1  -10  1
```

```
store["m5rowchange1"]
```

2. 第二行第二列的为主元,值为1, 通过行变换将第三行第二列变为 0,第三行减去第二行

```
md"2. 第二行第二列的为主元,值为1, 通过行变换将第三行第二列变为 0,第三行减去第二行"
```

```
3×4 Matrix{Int64}:
 3  -1    5  8
 0   1  -10  1
 0   0    0  0
```

```
store["m5rowchange2"]
```

矩阵现在是阶梯型, 第一行, 第二行有主元, 第三行没有主元. 方程组现在化简为:

$$3x-y+5z=8$$

$$y-10z=1$$

$x,y$ 称为主变量, $z$ 称为自由变量

$z$ 可以取任何值,如果用另一个变量$t$ 表示$z$ 的取值, 方程组可以变形为参数方程形式, 将 $z=t$ 带入方程得到:

$$z=t$$

$$x=\frac{5}{3}t$$

$$y=10t+1$$

```
md"""
矩阵现在是阶梯型, 第一行, 第二行有主元, 第三行没有主元.
方程组现在化简为:

$3x-y+5z=8$
$y-10z=1$

 $x,y$ 称为主变量, $z$ 称为自由变量

 $z$ 可以取任何值,如果用另一个变量$t$ 表示$z$ 的取值, 方程组可以变形为参数方程形式, 将$z=t$ 带入方程得到:

$z=t$

$x=\frac{5}{3}t$
$y=10t+1$
"""
```

```
3×4 Matrix{Int64}:
 3  -1    5  8
 0   1  -10  1
 0   0    0  0
```

```
let
    @variables x,y,z
    r1=[3 -1 5 8]
    r2=[0 1 -10 1]
    r3=[6 -1 0 17]
    augmentmatrix5=[r1;r2;r3]
    save("augmentmatrix5",augmentmatrix5)
    r3=r3-2*r1
    matrix=[r1;r2;r3]
    save("m5rowchange1",matrix)

    r3=r3-r2
    matrix=[r1;r2;r3]
    save("m5rowchange2",matrix)

end
```

## 线性方程组解集的总结:

1. 如果方程做经过行操作, 得到了形如下式:

$$\left[\begin{array}{ccc|c} 0 & 0 & 0 & b \end{array}\right]$$

的行, 方程组无解

2. 如果每一列都有主元, 那么方程组有一个解

$$\left[\begin{array}{ccc|c} 1 & 1 & -2 & 5 \\ 0 & 2 & 3 & 0 \\ 0 & 0 & 1 & 2 \\ 0 & 0 & 0 & 0 \end{array}\right]$$

3. 如果矩阵中有的列没有主元, 那么方程组有无限多个解:

$$\left[\begin{array}{ccc|c} 1 & 0 & 1 & 5 \\ 0 & 1 & 2 & -3 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{array}\right]$$

第三列没有主元, 所有方程组有多个解

**Definition**

方程组中主元的数量定义为:方程组的秩(Rank)

```
md"""
## 线性方程组解集的总结:

1. 如果方程做经过行操作, 得到了形如下式:

$\left[
    \begin{array}{ccc|c}
      0&0&0&b\\

    \end{array}
\right]$

的行, 方程组无解

2. 如果每一列都有主元, 那么方程组有一个解

$\left[
    \begin{array}{ccc|c}
      ❶&1&-2&5\\
      0&❷&3&0\\
      0&0&❶&2 \\
      0&0&0&0
    \end{array}
```

```
\right]$


3. 如果矩阵中有的列没有主元，那么方程组有无限多个解:

$\left[
    \begin{array}{ccc|c}
      ❶&0&1&5\\
      0&❶&2&-3\\
      0&0&0&0 \\
      0&0&0&0
    \end{array}
\right]$

第三列没有主元，所有方程组有多个解


!!! definition
    方程组中主元的数量定义为:方程组的秩(Rank)
"""
```

**Example**

example 5 下面矩阵的秩为?

```
3×3 Matrix{Int64}:
 1  2  3
 1  5  9
 2  4  6
```

经过行变换,矩阵变形为:

```
3×3 Matrix{Int64}:
 3  0  -3
 0  3   6
 0  0   0
```

第一列,第二列有主元, 第三列没有主元, 所以主元数为2, 矩阵的秩=2

在线性方程组中, 方程的数量(m),变量的数量(n) ,方程组(矩阵)的秩(rank,r)之间紧密联系.

线性方程组(矩阵)也像其他数学对象一样是是实实在在的对象, $m, n, r$ 度量了矩阵的三种性质.

不在这里详细讨论, 等学习到矩阵的时候再讨论这个问题.

```
3×3 Matrix{Int64}:
 3  0  -3
 0  3   6
 0  0   0
```

```julia
let
    r1=[1 2 3]
    r2=[1 5 9]
    r3=[2 4 6]
    orimatrix=[r1;r2;r3]
    save("rankexample",orimatrix)
    r2=r2-r1
    r3=r3-2*r1
    r1=3*r1-2*r2
    matrix=[r1;r2;r3]
    save("m6rowchange1",matrix)

end
```

```
read (generic function with 1 method)
```

```julia
begin
        store=Dict()

        function save(key::String, dict)
            store[key]=dict
        end

        function read(key::String)
            return  store[key]
        end
end
```