

# Linjära ekvationssystem

## 1 Inledning

Denna studioövning börjar med att vi påminner oss om matriser i MATLAB samtidigt som vi börjar se på matriser i matematiken. Sedan ser vi på uppbyggnad av matriser samt matrismultiplikation. Avslutningsvis ser vi på linjära ekvationssystem.

## 2 Matriser

En matris är som ni vet ett rektangulärt talschema:

$$\mathbf{A} = \begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & & \vdots \\ a_{m1} & \cdots & a_{mn} \end{bmatrix}$$

Matrisen ovan har  $m$  rader och  $n$  kolonner, vi säger att den är av typ  $m \times n$ . Ett matriselement i rad nr  $i$ , kolonn nr  $j$  tecknas  $a_{ij}$ , där  $i$  är radindex och  $j$  är kolonnindex. I MATLAB skrivs detta `A(i,j)` och `[m,n]=size(A)` ger matrisens typ, medan `m=size(A,1)` ger endast antal rader och `n=size(A,2)` ger endast antal kolonner.

Indexeringen  $i$  är alltid som i matrisen ovan, dvs. rad- och kolonnindex börjar alltid på ett och vi kan inte ändra på det.

En matris av typ  $m \times 1$  kallas kolonnmatris (kolonnvektor) och en matris av typ  $1 \times n$  kallas radmatris (radvektor):

$$\mathbf{b} = \begin{bmatrix} b_1 \\ \vdots \\ b_m \end{bmatrix}, \quad \mathbf{c} = [c_1 \quad \cdots \quad c_n]$$

Du kommer att se att vi använder oftast kolonnvektorer för att representera kvantiteter som vi beräknar. Element nr  $i$  ges i MATLAB av `b(i)` och antalet element ges av `m=length(b)`. Även för vektorer gäller att indexeringen alltid börjar på ett. Motsvarande gäller för radvektorn  $\mathbf{c}$ .

Som exempel tar vi

$$\mathbf{A} = \begin{bmatrix} 1 & 4 & 7 & 10 \\ 2 & 5 & 8 & 11 \\ 3 & 6 & 9 & 12 \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} 1 \\ 3 \\ 5 \end{bmatrix}, \quad \mathbf{c} = [0 \quad 2 \quad 4]$$

Vi skriver in detta i MATLAB enligt

```
>> A=[1 4 7 10; 2 5 8 11; 3 6 9 12]
>> b=[1; 3; 5]
>> c=[0 2 4]
```

och ser på typerna och några element med

```
>> [m,n]=size(A)
m =
    3
n =
    4

>> A(2,3)
ans =
    8
```

Prova gärna `length` och `size` på `b` och `c`. Någon skillnad? Skriv ut något element också.

En matris kan betraktas som en kollektion av kolonner:

$$\mathbf{A} = \begin{bmatrix} a_{11} & \cdots & a_{1j} & \cdots & a_{1n} \\ \vdots & & \vdots & & \vdots \\ a_{m1} & \cdots & a_{mj} & \cdots & a_{mn} \end{bmatrix} = [\mathbf{a}_1 \quad \cdots \quad \mathbf{a}_j \quad \cdots \quad \mathbf{a}_n]$$

med kolonnerna

$$\mathbf{a}_1 = \begin{bmatrix} a_{11} \\ \vdots \\ a_{m1} \end{bmatrix}, \quad \mathbf{a}_j = \begin{bmatrix} a_{1j} \\ \vdots \\ a_{mj} \end{bmatrix}, \quad \mathbf{a}_n = \begin{bmatrix} a_{1n} \\ \vdots \\ a_{mn} \end{bmatrix}$$

Man kan även betrakta den som en kollektion av rader, men vi använder oftast kolonnrepresentationen. I MATLAB plockar man ut kolonn nr  $j$  med `A(:,j)`. Här är  $j$  kolonnindex medan radindex  $i = 1, \dots, m$  representeras av tecknet kolon `:`. På liknande vis ges rad nr  $i$  av `A(i,:)`.

```
>> a1=A(:,1)
a1 =
    1
    2
    3

>> A2=A(2,:)
A2 =
    2     5     8    11
```

**Uppgift 1.** Skriv in följande matriser i MATLAB.

$$\mathbf{A} = \begin{bmatrix} 1 & 5 & 9 \\ 2 & 6 & 10 \\ 3 & 7 & 11 \\ 4 & 8 & 12 \end{bmatrix}, \quad \mathbf{B} = \begin{bmatrix} 4 & 5 & 6 \\ 3 & 2 & 1 \\ 1 & 1 & 1 \end{bmatrix} \quad \mathbf{c} = \begin{bmatrix} 1 \\ 3 \\ 5 \\ 7 \end{bmatrix}, \quad \mathbf{d} = [0 \quad 2 \quad 4]$$

(a). Skriv ut matriselementen  $a_{23}$ ,  $b_{23}$ ,  $c_2$  och  $d_3$ . Prova `size` och `length` på `A`, `B`, `c` och `d`. Ändra  $b_{23}$  genom att skriva `B(2,3)=5`.

(b). Skriv ut kolonn nr 1, 2 och 3 ur matrisen `A`. Sätt in kolonnvektorn `c` som 2:a kolonn i `A` genom att skriva `A(:,2)=c`.

(c). Radera matrisen **B** (`clear B`) och skriv in den igen genom att först bilda kolonnerna

$$\mathbf{b}_1 = \begin{bmatrix} 4 \\ 3 \\ 1 \end{bmatrix}, \quad \mathbf{b}_2 = \begin{bmatrix} 5 \\ 2 \\ 1 \end{bmatrix}, \quad \mathbf{b}_3 = \begin{bmatrix} 6 \\ 1 \\ 1 \end{bmatrix}$$

och sedan sätta in dem i matrisen  $\mathbf{B} = [\mathbf{b}_1 \ \mathbf{b}_2 \ \mathbf{b}_3]$ .

Vi kan ta ut ett block ur en matris med `A(iv,jv)` där `iv` är en vektor med radindex och `jv` är en vektor med kolonnindex. Resultatet blir en matris med `length(iv)` rader och `length(jv)` kolonner.

```
>> A=[1 3 5; 7 9 11; 2 4 6]      >> B=A([2 3],[1 3])
A =                                B =
     1     3     5                  7    11
     7     9    11                  2     6
     2     4     6
```

Transponatet  $\mathbf{A}^\top$  av en matris **A** ges av apostrof (`'`).

```
>> A=[1 3 5; 7 9 11]      >> B=A'
A =                        B =
     1     3     5          1     7
     7     9    11          3     9
                     5    11
```

**Uppgift 2.** Låt som i uppgift 1

$$\mathbf{A} = \begin{bmatrix} 1 & 5 & 9 \\ 2 & 6 & 10 \\ 3 & 7 & 11 \\ 4 & 8 & 12 \end{bmatrix}, \quad \mathbf{B} = \begin{bmatrix} 4 & 5 & 6 \\ 3 & 2 & 1 \\ 1 & 1 & 1 \end{bmatrix} \quad \mathbf{c} = \begin{bmatrix} 1 \\ 3 \\ 5 \\ 7 \end{bmatrix}, \quad \mathbf{d} = [0 \ 2 \ 4]$$

(a). Sätt in kolonnvektorn **c** som 3:e kolonn i **A** och sätt in radvektorn **d** som 2:a rad i **B**.

(b). Låt 1:a och 4:e raden i **A** byta plats och låt därefter den 2:a och 3:e kolonnen byta plats.

### 3 Bygga upp matriser

Med funktionerna `zeros` och `ones` kan man i MATLAB bilda matriser med nollor och ettor. Exempelvis `zeros(m,n)` ger en matris av storleken  $m \times n$  fylld med nollor. Med `zeros(size(A))` får vi en matris fylld med nollor av samma storlek som **A**. Motsvarande gäller för `ones`.

Enhetsmatriser bildas med funktionen `eye`, med `eye(n)` får vi enhetsmatrisen av storleken  $n \times n$ . Man kan också använda `eye` för att bilda rektangulära matriser med ettor på huvuddiagonalen och nollor för övrigt. Med `eye(m,n)` får vi en sådan matris av storleken  $m \times n$  och med `eye(size(A))` får vi en av samma storlek som **A**.

Med `diag` bildas diagonalmatriser. En vektor kan läggas in på en viss diagonal enligt

```
>> d=[2 3 7];
>> A=diag(d,0) % eller A=diag(d)
A =
    2     0     0
    0     3     0
    0     0     7

>> d=[2 3 7];
>> A=diag(d,1)
A =
    0     2     0     0
    0     0     3     0
    0     0     0     7
    0     0     0     0
```

Huvuddiagonalen markeras med 0, diagonalen ovanför till höger med 1, diagonalen nedanför till vänster med -1, osv. Matrisen blir så stor att vektorns alla element får plats längs angiven diagonal.

Vi kan sätta ihop två matriser **A** och **B** horisontellt med [**A B**] om antal rader i de två matriserna är lika. Två matriser **A** och **B** kan sättas ihop vertikalt med [**A; B**] om antal kolonner i de två matriserna är lika.

## 4 Linjärt ekvationssystem

Matriser används bland annat för att skriva ned linjära ekvationssystem. Vi tar som exempel: Ekvationssystemet

$$\begin{cases} x_1 + 2x_2 + 3x_3 = 14 \\ 3x_1 + 2x_2 + x_3 = 10 \\ 7x_1 + 8x_2 = 23 \end{cases}$$

kan skrivas på matrisform

$$\begin{bmatrix} 1 & 2 & 3 \\ 3 & 2 & 1 \\ 7 & 8 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 14 \\ 10 \\ 23 \end{bmatrix},$$

dvs.

$$\mathbf{Ax} = \mathbf{b}, \quad \text{med } \mathbf{A} = \begin{bmatrix} 1 & 2 & 3 \\ 3 & 2 & 1 \\ 7 & 8 & 0 \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} 14 \\ 10 \\ 23 \end{bmatrix}.$$

Vi bildar koefficientmatrisen **A** och högerledsvektorn **b** med

```
>> A=[1 2 3;3 2 1;7 8 0]
A =
    1     2     3
    3     2     1
    7     8     0

>> b=[14;10;23]
b =
    14
    10
    23
```

Med kommandot **rref** kommer vi till radreducerad trappstegsform (row-reduced-echelon form) så att vi kan läsa av lösningen till **Ax = b**.

Först bildar vi den utökade matrisen **E** = [**A b**] med

```
>> E=[A b]
E =
     1     2     3    14
     3     2     1    10
     7     8     0    23
```

och sedan får vi den reducerade matrisen med

```
>> R=rref(E)
R =
     1     0     0     1
     0     1     0     2
     0     0     1     3
```

Lösningen ser vi i sista kolonnen i R och vi har

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}$$

Som ytterligare ett exempel ser vi på följande ekvationssystem med oändligt många lösningar

$$\begin{cases} x_1 + 2x_2 + 3x_3 = 10 \\ 3x_1 + 2x_2 + x_3 = 14 \\ 7x_1 + 8x_2 + 9x_3 = 46 \end{cases}$$

eller på matrisform

$$\mathbf{Ax} = \mathbf{b} \quad \begin{bmatrix} 1 & 2 & 3 \\ 3 & 2 & 1 \\ 7 & 8 & 9 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 10 \\ 14 \\ 46 \end{bmatrix}$$

```
>> A=[1 2 3;3 2 1;7 8 9]
A =
     1     2     3
     3     2     1
     7     8     9
```

```
>> b=[10;14;46]
b =
    10
    14
    46
```

Vi reducerar utökande matrisen med

```
>> R=rref([A b])
R =
     1     0    -1     2
     0     1     2     4
     0     0     0     0
```

Vi har en fri variabel. Om vi sätter  $x_3 = t$  får vi

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 2 + t \\ 4 - 2t \\ t \end{bmatrix}$$

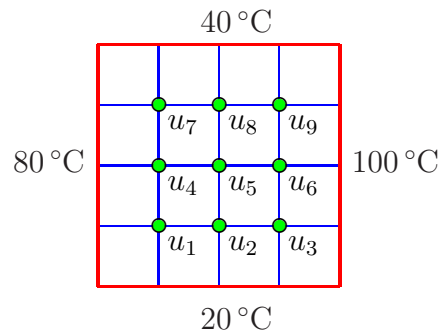
där  $t$  är ett godtyckligt reellt tal.

**Uppgift 3.** Skriv följande ekvationssystem på matrisform och lös dem sedan med `rref`.

$$\begin{cases} x_1 + 5x_2 + 9x_3 = 29 \\ 2x_1 + 5x_3 = 26 \\ 3x_1 + 7x_2 + 11x_3 = 39 \end{cases} \quad \begin{cases} x_1 + x_2 + 3x_3 + 4x_4 = 2 \\ -2x_1 + 2x_2 + 2x_3 = -4 \\ x_1 + x_2 + 2x_3 + 3x_4 = 1 \\ x_1 - x_2 - 2x_3 - x_4 = 1 \end{cases}$$

Skulle det finnas oändligt många lösningar skriv upp en formel för samtliga lösningar.

**Uppgift 4.** Vi skall beräkna temperaturen på en stålplatta där plattans kanter hålls vid temperaturer enligt figuren.



Antag att temperaturen i en nodpunkt är medelvärdet av temperaturena i de närmsta nodpunkterna i *väster*, *öster*, *söder* och *norr*. Låt  $u_1, u_2, \dots, u_9$  beteckna temperaturerna i de olika nodpunkterna. Sätt upp de ekvationer som ger temperaturen i de olika nodpunkterna. Skriv det linjära ekvationssystemet på matrisform  $\mathbf{A}\mathbf{u} = \mathbf{b}$  och lös detta i MATLAB.