

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:

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

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 $\mathbf{A}(:,j)$. Här är j kolonnindex medan radindex $i = 1, \dots, m$ representeras av tecknet kolon $:$. På liknande vis ges rad nr i av $\mathbf{A}(i,:)$.

```
>> 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 $\mathbf{A}$, $\mathbf{B}$, $\mathbf{c}$ och $\mathbf{d}$. Ändra b_{23} genom att skriva $\mathbf{B}(2,3)=5$.

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

(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 Matris-vektorprodukt

Matris-vektorprodukten $\mathbf{y} = \mathbf{Ax}$ av en $m \times n$ -matris och en n -kolonnvektor är en m -kolonnvektor som ges av

$$\begin{array}{c} \begin{bmatrix} y_1 \\ \vdots \\ y_m \end{bmatrix} \\ \mathbf{y} \end{array} = \begin{array}{c} \begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & & \vdots \\ a_{m1} & \cdots & a_{mn} \end{bmatrix} \\ \mathbf{A} \end{array} \begin{array}{c} \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} \\ \mathbf{x} \end{array} = \begin{array}{c} \begin{bmatrix} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n \\ \vdots \\ a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n \end{bmatrix} \\ \mathbf{Ax} \end{array}$$

eller elementvis

$$y_i = \sum_{j=1}^n a_{ij}x_j = a_{i1}x_1 + a_{i2}x_2 + \cdots + a_{in}x_n$$

Matris-vektorprodukten $\mathbf{y} = \mathbf{Ax}$ kan beräknas i MATLAB med den inbyggda matrismultiplikationen (*) enligt $\mathbf{y}=\mathbf{A}*\mathbf{x}$ eller med lite egen programmering (som bygger upp $\mathbf{y}$ elementvis)

```
>> y=zeros(m,1);
>> for i=1:m
    s=0;
    for j=1:n
        s=s+A(i,j)*x(j);
    end
    y(i)=s;
end
```

Ett alternativt sätt att introducera matris-vektorprodukt är att definiera $\mathbf{Ax}$ som en linjärkombination av kolonnerna i $\mathbf{A}$, (se Lay avsnitt 1.4)

$$\begin{aligned}\mathbf{y} = \mathbf{Ax} &= [\mathbf{a}_1 \quad \cdots \quad \mathbf{a}_n] \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} = x_1 \mathbf{a}_1 + x_2 \mathbf{a}_2 + \cdots + x_n \mathbf{a}_n = \\ &= \begin{bmatrix} a_{11} \\ \vdots \\ a_{m1} \end{bmatrix} x_1 + \begin{bmatrix} a_{12} \\ \vdots \\ a_{m2} \end{bmatrix} x_2 + \cdots + \begin{bmatrix} a_{1n} \\ \vdots \\ a_{mn} \end{bmatrix} x_n\end{aligned}$$

I MATLAB skulle vi, för t.ex. $n = 3$, skriva

```
>> y=A(:,1)*x(1)+A(:,2)*x(2)+A(:,3)*x(3)
```

och för ett större värde på n skulle vi kunna bilda linjärkombinationen enligt

```
>> y=zeros(m,1);
>> for j=1:n
    y=y+A(:,j)*x(j);
end
```

Uppgift 3. Skriv in följande matriser i MATLAB.

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

(a). Beräkna följande produkter, både för hand, dvs. med penna och papper, och med MATLAB, dvs. med inbyggda matrismultiplikationen (*),

$$\mathbf{Ax}, \quad \mathbf{Bx}, \quad \mathbf{AB}, \quad \mathbf{ax}, \quad \mathbf{xa}, \quad \mathbf{aB}.$$

(b). Beräkna produkten $\mathbf{Ax}$ även genom att ni skriver en egen programkod i MATLAB. Skriv snyggt och tydligt.

5 Matris-matrisprodukt

Matris-matrisprodukten $\mathbf{C} = \mathbf{AB}$ av en $m \times n$ -matris $\mathbf{A}$ och en $n \times p$ -matris $\mathbf{B}$, med kolonner $\mathbf{b}_1, \dots, \mathbf{b}_p$, är en $m \times p$ -matris som ges av

$$\mathbf{C} = \mathbf{AB} = \mathbf{A}[\mathbf{b}_1, \dots, \mathbf{b}_p] = [\mathbf{Ab}_1, \dots, \mathbf{Ab}_p]$$

eller elementvis

$$c_{ij} = \sum_{k=1}^n a_{ik} b_{kj}, \quad i = 1, \dots, m, \quad j = 1, \dots, p$$

Matrismultiplikationen $\mathbf{C} = \mathbf{AB}$ kan beräknas i MATLAB med den inbyggda matrismultiplikationen (*) enligt $\mathbf{C}=\mathbf{A*B}$ eller med lite egen programmering (som bygger upp $\mathbf{C}$ elementvis)

```
>> C=zeros(m,p);
>> for i=1:m
    for j=1:p
        cij=0;
        for k=1:n
            cij=cij+A(i,k)*B(k,j);
        end
        C(i,j)=cij;
    end
end
```

Alternativt bygger vi upp kolonnvis enligt

```
>> C=zeros(m,p);
>> for j=1:p
    C(:,j)=A*B(:,j);
end
```

Uppgift 4. Skriv in följande matriser i MATLAB.

$$\mathbf{A} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix}, \quad \mathbf{B} = \begin{bmatrix} 1 & 0 & 0 \\ -2 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad \mathbf{C} = \begin{bmatrix} 2 & 1 & 1 \\ 4 & 1 & 0 \\ -2 & 2 & 1 \end{bmatrix}$$

(a). Kontrollera att associativa och distributiva lagarna gäller för dessa matriser.

Du skall alltså se att $\mathbf{A}(\mathbf{BC}) = (\mathbf{AB})\mathbf{C}$ respektive $\mathbf{A}(\mathbf{B} + \mathbf{C}) = \mathbf{AB} + \mathbf{AC}$ och $(\mathbf{B} + \mathbf{C})\mathbf{A} = \mathbf{BA} + \mathbf{CA}$.

(b). Vanligtvis är matrismultiplikation inte kommutativ. Tex är $\mathbf{AC} \neq \mathbf{CA}$ och $\mathbf{BC} \neq \mathbf{CB}$ (kontrollera gärna), men vad gäller för $\mathbf{AB}$ och $\mathbf{BA}$?

6 Linjärt ekvationssystem

Matriser används bland annat för att skriva ned linjära ekvationssystem. 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 ska lära oss hur man löser sådana ekvationssystem. I MATLAB finns backslash-kommandot (`\`) eller alternativt kommandot `rref` (row-reduced-echelon form) som löser systemet, $\mathbf{Ax} = \mathbf{b}$:

```
>> x=A\b
>> rref([A b])
```

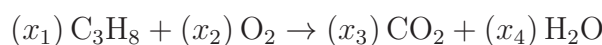
I det första fallet fungerar det bra om lösningen är entydig men sämre om det finns fria variabler eller inga lösningar alls. I det andra fallet reducerar MATLAB den utökade matrisen $[A \ b]$ till reducerad trappstegsform.

Uppgift 5. Skriv följande ekvationssystem på matrisform och lös dom sedan med både $\backslash$ och `rref`.

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

Varför fungerar inte $\backslash$ för det andra ekvationssystemet?

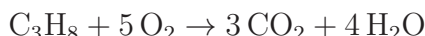
Uppgift 6. En liten stökiometriuppgift. Vid förbränning av propan (C_3H_8) gäller följande kemiska reaktionsformel (obalanserad)



För att balansera formeln representerar vi C_3H_8 med vektorn $(3, 8, 0)$, O_2 med vektorn $(0, 0, 2)$, osv. På detta sätt kan den obalanserade ekvationen skrivas

$$\begin{bmatrix} 3 \\ 8 \\ 0 \end{bmatrix} x_1 + \begin{bmatrix} 0 \\ 0 \\ 2 \end{bmatrix} x_2 - \begin{bmatrix} 1 \\ 0 \\ 2 \end{bmatrix} x_3 - \begin{bmatrix} 0 \\ 2 \\ 1 \end{bmatrix} x_4 = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

Lösningen $\mathbf{x} = (1, 5, 3, 4)$ (den minimala heltalslösningen) ger den balanserade reaktionsformeln



(a). Balansera nu den kemiska formeln

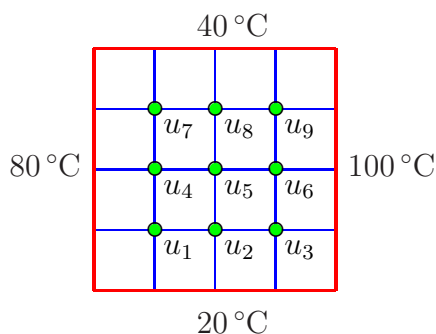


som är av intresse bl.a. vid framställning av arsin (AsH_3).

Med `format rat` skrivs beräkningsresultat med rationella tal och det blir enklare att tolka svaret. Standardformatet får vi sedan tillbaka med `format short`.

(b). Skriv ned den balanserade formeln på papper.

Uppgift 7. 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.