

Introduktion till MATLAB

Kompendiet

Lektor: Yury Shestopalov

e-mail: youri.shestopalov@kau.se Tel. 054-7001856

Hemsidan: www.ingvet.kau.se/~youri

Karlstads Universitet

2002

Contents

1	Filer i MATLAB	5
2	MATLAB som miniräknare	6
2.0.1	Exempel	6
2.0.2	Exempel	6
2.0.3	Exempel	6
2.0.4	Exempel	7
2.0.5	Problem	8
2.0.6	Problem	8
3	Aritmetiska uttryck och matematiska funktioner i MATLAB	8
3.1	Aritmetiska operatörer	8
3.1.1	Exempel	8
3.2	Programmering av aritmetiska och algebraiska uttryck i MATLAB	8
3.2.1	Exempel	8
3.3	Mathematiska funktioner i MATLAB	9
3.3.1	Exempel	9
3.4	Programmering av matematiska uttryck	9
3.4.1	Exempel	9
3.4.2	Problem	10
4	Kommandofiler och funktionsfiler i MATLAB	10
4.0.3	Exempel: en funktionsfil	10
4.0.4	Exempel: en funktionsfil	10
4.0.5	Exempel	12
4.0.6	Problem	13
4.0.7	Problem	13
5	Introduktion till matrisoperationer i MATLAB	13
5.0.8	Exempel	13
5.0.9	Exempel	14
5.0.10	Exempel	14
5.0.11	Exempel: Definition av radvektorer och kolonnvektorer	14
5.1	Matriser och matrisoperationer	15
5.2	Elementvisa operationer	16
5.2.1	Exempel	16
5.3	Elementvisa funktioner	16
5.3.1	Exempel	16
5.4	Matrisbyggande funktioner	17
5.4.1	Exempel	17
5.4.2	Exempel	18
5.4.3	Problem	21
5.4.4	Problem	21
5.4.5	Problem	21
5.4.6	Problem	22
5.4.7	Problem	22

5.5	Vektorer och delmatriser	22
5.5.1	Exempel	22
5.5.2	Exempel	23
5.5.3	Exempel: Generering av en funktionstabell (sinus)	23
5.6	Nät och noder	23
5.6.1	Problem	24
5.6.2	Exempel	24
5.6.3	Exempel	24
5.7	Variabelstorlek	24
5.7.1	Exempel	25
5.7.2	Problem	25
5.7.3	Problem	25
6	2D-grafik	25
6.0.4	Exempel	25
6.0.5	Problem	26
7	Strängar i MATLAB	26
7.0.6	Exempel	26
8	Utskrift och inläsning	26
8.0.7	Exempel	27
8.0.8	Problem	27
9	Komplexa tal och funktioner	27
9.0.9	Exempel	28
9.0.10	Problem	28
10	Utskriftformat	28
10.1	Spara resultat, lagra och hämta data	29
10.1.1	Exempel	29
10.1.2	Problem	29
11	Relationsoperatorer	29
11.0.3	Exempel	30
12	Logiska funktioner	30
13	Programmering i MATLAB	31
13.1	Villkorssatser	31
13.1.1	Exempel	31
13.1.2	Exempel	31
13.2	Repetitionssatser	32
13.2.1	Exempel 12.3	32

14 Nollställen till funktioner i MATLAB	36
14.0.2 Exempel	37
14.0.3 Problem	38
14.0.4 Problem	38
14.0.5 Problem	38
15 Integralberäkning i MATLAB	39
15.0.6 Exempel	39
16 Numerisk lösning av ordinära differentialekvationer i MATLAB	39
16.1 Begynnelsevärdesproblem för ODE	40
16.1.1 Exempel	40
16.1.2 Exempel	40
16.1.3 Exempel	41

1 Filer i MATLAB

Man kan skriva MATLAB kommandon i en text fil med någon editor, t ex MATLAB editor som startas med

```
>> edit
```

Dessa kommandon exekveras i MATLAB då användaren skriver filens namn och eventuella parametrar.

Man skriver filer med hjälp av MATLAB editor och sparar filer i kataloger och underkataloger.

En MATLAB fil (M-fil) är en fil med namnet på formen
filnamn.m

dvs det måste ha suffix .m.

Systemkommandot

```
>> dir
```

listar alla filer i en katalog eller underkataloger. Systemkommandot

```
>> cd
```

visar (ändrar) den aktuella katalogen, t ex,

```
>> cd d:\minmatlabkurs
```

Det är en bra ide att skapa ett bibliotek (katalog) som heter matlab (eller minmatlabkurs) och spara alla sina M-filer i detta. MATLAB tittar automatiskt efter denna katalog (matlab), eller katalogen minmatlabkurs om man skriver

```
>> cd d:\minmatlabkurs
```

och hittar då filer.

T ex, MATLABraden

```
>> cd
```

```
C:\MATLAB\minmatlabkurs
```

visar aktuell katalog (dvs minmatlabkurs) som man använder, och MATLABraden

```
>> dir
```

```
.
```

```
..
```

```
a.dat
```

```
mintextfil.txt
```

```
frstkommfil.m
```

```
minmatlabfunk1.m
```

```
minmatlabfunk2.m
```

```
figdraw.mat
```

```
thematrix.m
```

```
exempel106.m
```

```
exempel117.m
```

```
problem213.m
```

```
problem214.m
```

```
problem312.m ...
```

visar alla filer i denna katalog.

2 MATLAB som miniräknare

Man kan använda MATLAB som miniräknare för att räkna aritmetiska uttryck, t ex räkningar.

2.0.1 Exempel

Man kan räkna genom att skriva direkt i MATLABfönstret, t ex,

```
>> 2341 + 201 + 342 + 4556 + 213
```

och trycka *Enter* för att få svaret

```
>> ans =  
      7653
```

Man kan göra det också genom att skapa en vanlig ASCII-text-fil

```
maj2002 = 2341 + 201 + 342 + 4556 + 213
```

och spara den som en MATLAB M-fil (kommandofil), dvs som en fil med namnet rakn-maj2002.m (med suffix .m) i en katalog. Sedan skriver man MATLABraden (filens namn) i MATLABfönstret

```
>> raknmaj2002
```

och får svaret

```
>> maj2002 =  
      7653
```

2.0.2 Exempel

För att beräkna talet $\sqrt{2} + 2 - 0.2 \cdot 1.8$, skriver man en ASCII-text-fil

```
sqrt(2) + 2 - 0.2*1.8
```

och sparar den som en MATLAB M-fil (kommandofil) exempel102.m i sin MATLAB-katalog.

Sedan skriver man filens namn i MATLABfönstret

```
>> exempel102
```

och får svaret

```
>> ans =  
      3.0542
```

Man kan skriva direkt i MATLABfönstret

```
>> sqrt(2) + 2 - 0.2*1.8
```

och trycka *Enter* för att få svaret.

2.0.3 Exempel

Beräkna

$$f(x) = x[\sqrt{x+1} - \sqrt{x}] \quad (1)$$

för växande

```
x =  
1.00000  
10.0000  
100.000  
1000.00  
10000.0  
100000
```

Man kan skriva direkt i MATLABfönstret

```
>> sqrt(1+1) - 1, 10*(sqrt(10+1) - sqrt(10)), 100*(sqrt(100+1) - sqrt(100)), 1000*(sqrt(1000+1)
- sqrt(1000)), 10000*(sqrt(10000+1) - sqrt(10000)),
100000*(sqrt(100000+1) - sqrt(100000))
```

och få svaret (med fyra korrekta decimaler)

```
>> ans =
    0.4142
>> ans =
    1.5434
>> ans =
    4.9876
...
>> ans =
   158.1135
```

Man kan också skapa en funktionsfil

```
function y = ex103(x)
y = x*(sqrt(x+1) - sqrt(x));
```

och sedan, en kommandofil som kör funktionsfilen, t ex en ASCII-text-fil

```
ex103(1), ex103(10), ex103(100), ex103(1000), ex103(10000), ex103(100000)
```

och spara den som en kommandofil ex103kom.m i sin MATLABkatalog. När man skriver i MATLABfönstret

```
>> ex103kom
```

då får man svaret ovan.

2.0.4 Exempel

Beräkna uttrycket

$$\frac{a}{b-c}, \quad a = 0.81534, \quad b = 35.724, \quad c = 35.596.$$

Vi skapar en funktionsfil

```
function t = ex104(x,y,z)
t = x/(y-z);
```

som räknar en funktion av tre variabler, och sedan, en kommandofil som kör funktionsfilen,

```
a=0.81534, b=35.724, c=35.596, ex104(a,b,c)
```

(vi sparar den med namnet ex104kom.m i en MATLABkatalog). När man skriver i MATLABfönstret

```
>> ex104kom
```

då får man

```
>> a =
    0.8153
>> b =
   35.7240
>> c =
   35.5960
>> ans =
    6.3698
```

2.0.5 Problem

Använd MATLAB för att beräkna

$$4 + 5/2 \cdot 7$$

$$4 + (5/2) \cdot 7$$

$$3 + \frac{9 \cdot 4}{3 \cdot 2} \cdot 8/2 - 5$$

2.0.6 Problem

Skriv M-filer som räknar dina kommande räkningar.

3 Aritmetiska uttryck och matematiska funktioner i MATLAB

3.1 Aritmetiska operatörer

potens

* multiplikation

/ högerdivision (vanlig division)

\ vänsterdivision

+ addition

- subtraktion

Parenteser

()

3.1.1 Exempel

Det aritmetiska uttrycket $a/b + c$ tolkar MATLAB som

$$\frac{a}{b} + c$$

men uttrycket $a/(b + c)$ tolkas som

$$\frac{a}{b + c}$$

3.2 Programmering av aritmetiska och algebraiska uttryck i MATLAB

3.2.1 Exempel

Betrakta uttrycken

$$x^3 + bx^2 + cx - 2.3 \quad \frac{a^2 - b^3}{c + 4} \quad \frac{\frac{ab}{x-15}}{d + \frac{3}{5}}$$

och skriv motsvarande MATLAB kommandon

$$x^3 + b*x^2 + c*x - 2.3, \quad (a^2 - b^3)/(c + 4), \quad ((a*b)/(x - 15))/(d + 3/5)$$

Observera att ett uttryck kan vara argument till en funktion, t ex
`sqrt(sin(x) + exp(2*x))`

dvs

$$\sqrt{\sin x + e^{2x}}$$

3.3 Matematiska funktioner i MATLAB

abs(x) |x|
sign(x) -1 (x < 0), 0 (x = 0), 1 (x > 0)
sqrt(x) \sqrt{x}
pow2(x,f) $x * 2^f$
exp(x) e^x
log(x)
log10(x)
log2(x)
sin(x)
cos(x)
tan(x)
cot(x)
...

3.3.1 Exempel

logarithm = log10(100)
returnerar

$$\text{logarithm} = 2$$

e = exp(1)
ger

$$e = 2.7183$$

3.4 Programmering av matematiska uttryck

3.4.1 Exempel

Betrakta uttrycket

$$\sqrt{x^{2f} + b\frac{x^3}{7} + 1} \quad \log(y + \sin x) \quad \cos\left(\frac{a}{x} + e^{2x}\right)$$

och skriv motsvarande MATLAB kommandon

$$\text{sqrt}(x^{(2*f)} + b*((x^3)/7) + 1), \quad \text{log}(y + \sin(x)), \quad \text{cos}(a/x + \text{exp}(2*x))$$

3.4.2 Problem

Skriv MATLAB kommandon som beräknar fem Taylors formel termer

(a)

$$\ln(1+x) \approx \ln_5(x) = x - \frac{x^2}{2} + \frac{x^3}{3} - \frac{x^4}{4} + \frac{x^5}{5}$$

(b)

$$e^x \approx \exp_5(x) = 1 + x + \frac{x^2}{1 \cdot 2} + \frac{x^3}{1 \cdot 2 \cdot 3} + \frac{x^4}{1 \cdot 2 \cdot 3 \cdot 4} + \frac{x^5}{1 \cdot 2 \cdot 3 \cdot 4 \cdot 5}$$

(c)

$$\sin x \approx \sin_5(x) = x - \frac{x^3}{1 \cdot 2 \cdot 3} + \frac{x^5}{1 \cdot 2 \cdot 3 \cdot 4 \cdot 5} - \frac{x^7}{1 \cdot 2 \cdot 3 \cdot \dots \cdot 7} + \frac{x^9}{1 \cdot 2 \cdot 3 \cdot \dots \cdot 9}$$

Utför test beräkningar för ett givet tal $x = x_t$ ($|x_t| \leq 1$) och jämför resultaten med de exakta värdena beräknade med MATLAB standarda matematiska funktioner; dvs bestäm

$$\ln(1+x_t), \quad \ln_5(x_t) \quad err_5 = \ln(1+x_t) - \ln_5(x_t),$$

$$e^{x_t}, \quad \exp_5(x_t) \quad err_5 = e^{x_t} - \exp_5(x_t),$$

$$\sin x_t, \quad \sin_5(x_t) \quad err_5 = \sin x_t - \sin_5(x_t).$$

4 Kommandofiler och funktionsfiler i MATLAB

Det finns två viktiga typer av MATLAB filer (M-filer): *funktionsfiler* och *kommandofiler*.

4.0.3 Exempel: en funktionsfil

```
function y = ex103(x)
    y = x*(sqrt(x+1) - sqrt(x));
```

```
function t = ex104(x,y,z)
    t = x/(y-z);
```

4.0.4 Exempel: en kommandofil

Här är ett avancerat exempel på en kommandofil `frstkommfil.m` som skrivs med hjälp av MATLAB editor.

```
disp('Först kommandofil');
A = [15 -1 2
     0 10 4
     1 1 -1];
A
b=[13; 10; 3];
b
x = A\b
x5 = linspace(-1,0,5); x10 = linspace(-1,0,10);
```

```

x5
y5 = x5.*x5.*x5 -6.*x5.*x5 +5.*x5 + 12;
y5
y10 = x10.*x10.*x10 -6.*x10.*x10 +5.*x10 + 12;
in5 = trapz(x5, y5); in10 = trapz(x10, y10)
x6 = linspace(-1, 0, 6); y6 = x6.*x6.*x6 -6.*x6.*x6 +5.*x6 + 12;
in6 = trapz(x6, y6)
xp = linspace(-3, 3, 30); f=(xp+1)./(xp+5);
yp=(xp.*xp.*xp -4.*xp.*xp +11.*xp + 16)./80;
plot(xp, f, '*', xp, yp, '-')
grid
xp = linspace(-3,3,50); f=(xp+1)./(xp+5);
yp = (xp.*xp.*xp -4.*xp.*xp +11.*xp + 16)./80;
plot(xp,f, '*',xp,yp, '-')
grid

```

I filen frstkommfil.m definierar man en kvadratisk matris av typ 3×3

$$\mathbf{A} = [a_{jk}] = \begin{bmatrix} 15 & -1 & 2 \\ 0 & 10 & 4 \\ 1 & 1 & -1 \end{bmatrix}$$

och kolonnvektorn

$$\mathbf{b} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} = \begin{bmatrix} 13 \\ 10 \\ 3 \end{bmatrix}.$$

Kommandot $x = A \setminus b$ löser linjära ekvationssystem med 3 obekanta $\mathbf{Ax} = \mathbf{b}$:

$$\begin{aligned} 15x_1 - x_2 + 2x_3 &= 13 \\ 10x_2 + 4x_3 &= 10 \\ x_1 + x_2 - x_3 &= 3, \end{aligned}$$

Kommandot $x5 = \text{linspace}(-1,0,5)$; returnerar vektorer med 5 element i intervallet $[-1, 0]$; första element är -1 och sista element är 0 .

Kommandot $y5 = x5.*x5.*x5 -6.*x5.*x5 +5.*x5 + 12$; beräknar polynomet $p(x5) = y5 = (x5)^3 - 6(x5)^2 + 5(x5) + 12$.

Kommandot $\text{in5} = \text{trapz}(x5, y5)$; beräknar integralet $\int_{-1}^0 p(x)dx$ med trapetsformeln.

Kommandot $\text{plot}(xp, f, '*', xp, yp, '-')$ ritar vektorn f mot xp med stjärnorna $*$ och vektorn yp mot xp med $-$.

```

MATLABraden
>> frstkommfil
ger resultatet
Först kommandofil
A =
    15 -1 2
     0 10 4

```

```

1 1 -1
b =
13
10
3
x =
1.0427
1.2735
-0.6838
x5 =
-1.0000 -0.7500 -0.5000 -0.2500 0
y5 =
0 4.4531 7.8750 10.3594 12.0000
in10 =
7.2346
in5 =
7.1719
in6 =
7.2000

```

4.0.5 Exempel

MATLABraden

```

>> type sec
ger
function y = sec(z)
% SEC secant
% SEC(X) is the secant of the element of X
% ...
y = 1./cos(z);

```

Detta är ett exempel på *en funktionsfil* som innehåler en (egendefinierad eller MATLAB-definierad) funktion. Skriv två andra funktionsfiler:

```

function yy = minmatlabfunk1(x)
yy = log(5 + sin(x));

```

Man måste spara den är filen med namn `minmatlabfunk1.m`

```

function z = minmatlabfunk2(x,y)
f1 = x - 2.*sin(x); f2 = y - 2.*sin(y); % ex. 17.2.8
z = y - f2.*((y-x)./(f2 - f1)); % ex. 17.2.8
% z = (1./3).*(x.*x+1); % ex. 17.2.1 g1

```

Man måste spara den är filen med namn `minmatlabfunk2.m`

Kommentaraderna (text som inleds med `%`) används i MATLAB för dokumentation och förklaring.

4.0.6 Problem

Skriv en funktionsfil `problem1.m` som löser den linjära ekvationen

$$ax = b, \quad a \neq 0,$$

för olika a och b . Utför funktionsfilen med olika indata. Kolla resultat.

4.0.7 Problem

Skriv en funktionsfil `problem2.m` som löser den kvadratiske ekvationen

$$x^2 + 2bx - 1 = 0, \quad b \geq 0,$$

för olika b . Utför funktionsfilen med olika indata. Kolla resultat.

5 Introduktion till matrisoperationer i MATLAB

Betrakta ett linjärt ekvationssystem med n obekanta

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + \dots + a_{1n}x_n &= b_1 \\ a_{21}x_1 + a_{22}x_2 + \dots + a_{2n}x_n &= b_2 \\ &\dots \dots \dots \\ a_{m1}x_1 + a_{m2}x_2 + \dots + a_{mn}x_n &= b_m, \end{aligned} \tag{2}$$

eller

$$\mathbf{Ax} = \mathbf{b}, \tag{3}$$

där

$$\mathbf{A} = [a_{jk}] = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \cdot & \cdot & \dots & \cdot \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{bmatrix}$$

är en *rektangulär matris* av typ $m \times n$ (\mathbf{A} har m rader och n kolonner, eller storlek $m \times n$). Om $m = n$, då \mathbf{A} är en *kvadratisk matris* av typ $n \times n$.

Elementen, dvs talen a_{jk} i matrisen är i de flesta fall reela tal. Kolonnvektorerna

$$\mathbf{x} = \begin{bmatrix} x_1 \\ \dots \\ x_n \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} b_1 \\ \dots \\ b_m \end{bmatrix}.$$

5.0.8 Exempel

(a) Betrakta en matris av typ 2×3

$$\mathbf{a} = [a_{jk}] = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \end{bmatrix} = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}$$

\mathbf{a} har 2 rader och 3 kolonner. Den första raden är [1 2 3]. De andra och tredje kolonner är

$$\begin{bmatrix} 2 \\ 5 \end{bmatrix}, \begin{bmatrix} 3 \\ 6 \end{bmatrix}.$$

Matriselementen

$$a_{11} = 1, \quad a_{12} = 2, \quad \dots \quad a_{23} = 6, \quad \text{etc.}$$

I MATLAB, definition av matrisen \mathbf{a} av typ 2×3 kan ske genom att (i) man ger elementen radvis:

```
>> A = [1 2 3  
4 5 6];
```

(ii) ger elementen på samma kommandorad, där semikolon skiljer matrisraderna,

```
>> A = [1 2 3; 4 5 6];
```

eller (iii) elementvis:

```
>> a(1,1) = 1; a(1,2) = 2; a(1,3) = 3; a(2,1) = 4; a(2,2) = 5; a(2,3) = 6;
```

5.0.9 Exempel

Kommandofilen `thematrix.m` består av MATLABraden

```
A = [-9 -3 -16; 13 7 16; 3 3 10];
```

Kommandot

```
>> thematrix
```

skapar matrisen \mathbf{A} . Genom att skriva

```
>> A
```

man får den här matrisen

```
A =  
    -9    -3   -16  
    13     7    16  
     3     3    16
```

5.0.10 Exempel

Antag att ASCII-filen `a.dat` skapats med en editor:

```
3 4 5
```

```
4 2 9
```

I MATLAB, får vi då med MATLABraden

```
>> load a.dat, a
```

```
a =
```

```
 3 4 5
```

```
 4 2 9
```

Då filen `a.dat` definierar matrisen a av typ 2×3 .

5.0.11 Exempel: Definition av radvektorer och kolonnvektorer

```
>> radvek = [1.2 3.2 4];
```

```
>> kolvek = [2.7; 3.4; -9.2];
```

Genom att skriva

```
>> kolvek
```

så visas
kolvek =
2.7000
3.4000
-9.2000

5.1 Matriser och matrisoperationer

Låt $\mathbf{A} = [a_{jk}]$ vara en matris av typ $n \times m$. I MATLAB,

$\mathbf{C} = \mathbf{A} \pm \mathbf{B}$ ger summan (differensen) av matriser \mathbf{A} och \mathbf{B} om $\mathbf{B} = [b_{jk}]$ är en matris av samma storlek som \mathbf{A} .

Matrismultiplikation, dvs matrisprodukten

$$\mathbf{C} = \mathbf{AB}$$

av matriser \mathbf{A} och \mathbf{B} , $\mathbf{C} = [c_{jk}]$, är definierad om antalet kolonner i \mathbf{A} är lika med antalet rader i \mathbf{B} , dvs $\mathbf{A} = [a_{jk}]$ är en matris av typ $n \times m$ (n rader, m kolonner) och $\mathbf{B} = [b_{jk}]$ är en matris av typ $m \times n$ (m rader, n kolonner). Elementet c_{jk} är skalärprodukten mellan j -te raden i \mathbf{A} och k -te kolonnen i \mathbf{B} .

Särskild, matrismultiplikation är definierad om \mathbf{A} och \mathbf{B} är kvadratiska matriser som har samma storlek och om $\mathbf{A} = [a_{jk}]$ är en matris av typ $n \times m$ och $\mathbf{B} = \mathbf{x}$ är en kolonnvektor med m rader (och 1 kolonn). Då matrisprodukten $\mathbf{b} = \mathbf{Ax}$ blir en kolonnvektor med n rader (och 1 kolonn) som skrivs i formen av ekvationssystemet (2)

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + \dots + a_{1m}x_m &= b_1 \\ a_{21}x_1 + a_{22}x_2 + \dots + a_{2n}x_m &= b_2 \\ &\dots \dots \dots \\ a_{n1}x_1 + a_{n2}x_2 + \dots + a_{nm}x_m &= b_n, \end{aligned}$$

I MATLAB,

$\mathbf{C} = \mathbf{A} * \mathbf{B}$ ger matrisprodukten $\mathbf{C} = \mathbf{AB}$ av matriser \mathbf{A} och \mathbf{B} .

$\text{norm}(\mathbf{A}, 'fro')$ ger den Frobeniusnormen av \mathbf{A}

$$\|\mathbf{A}\| = \|\mathbf{A}\|_F = \sqrt{\sum_{j=1}^n \sum_{k=1}^m a_{jk}^2}.$$

$\text{norm}(\mathbf{A}, 1)$ ger den kolonnsumnormen (den största kolonnbeloppssumman) av \mathbf{A}

$$\|\mathbf{A}\| = \|\mathbf{A}\|_1 = \max_k \sum_{j=1}^n |a_{jk}|.$$

$\text{norm}(\mathbf{A}, \text{inf})$ ger den radsumnormen (den största radbeloppssumman) av \mathbf{A}

$$\|\mathbf{A}\| = \|\mathbf{A}\|_\infty = \max_j \sum_{k=1}^m |a_{jk}|.$$

5.2 Elementvisa operationer

+ addition

– subtraktion

.* multiplikation

./ höger division

.\ vänster division

power

Matriserna måste ha samma storlek.

5.2.1 Exempel

$$\mathbf{A} = \begin{bmatrix} 1 & 2 \\ -1 & 5 \end{bmatrix}, \quad \mathbf{B} = \begin{bmatrix} 7 & 2 \\ 1 & 0 \end{bmatrix}, \quad \mathbf{C} = \begin{bmatrix} 1+2i & 5-2i \\ 3+i & 1+3i \end{bmatrix}$$

A.*B ger

$$\begin{bmatrix} 1 \cdot 7 & 2 \cdot 2 \\ -1 \cdot 1 & 5 \cdot 0 \end{bmatrix} = \begin{bmatrix} 7 & 4 \\ -1 & 0 \end{bmatrix}$$

B./A ger

$$\begin{bmatrix} 7/1 & 2/2 \\ 1/(-1) & 0/5 \end{bmatrix} = \begin{bmatrix} 7 & 1 \\ -1 & 0 \end{bmatrix}$$

B. 2 = B.*B ger

$$\begin{matrix} 49 & 4 \\ 1 & 0 \end{matrix}$$

C.' ger

$$\begin{matrix} 1.0000 + 2.0000i & 3.0000 + 1.0000i \\ 5.0000 - 2.0000i & 1.0000 + 3.0000i \end{matrix}$$

5.3 Elementvisa funktioner

Om f är en matematiska standardfunktion och $\mathbf{A} = [a_{ij}]$ så gäller

$$f(\mathbf{A}) = f(a_{ij}).$$

5.3.1 Exempel

$$\mathbf{A} = \begin{bmatrix} 0 & -3 \\ 5 & 1 \\ 4 & -6 \end{bmatrix}, \quad \mathbf{B} = \begin{bmatrix} \pi & 0 \\ \pi/2 & \pi/4 \end{bmatrix}, \quad \mathbf{C} = \begin{bmatrix} 1+i & -\pi i \\ 0 & 2-i \end{bmatrix}.$$

abs(A) ger

$$\begin{bmatrix} \text{abs}(0) & \text{abs}(-3) \\ \text{abs}(5) & \text{abs}(1) \\ \text{abs}(4) & \text{abs}(-6) \end{bmatrix} = \begin{bmatrix} 0 & 3 \\ 5 & 1 \\ 4 & 6 \end{bmatrix}.$$

cos(B) ger

$$\begin{matrix} -1.0000 & 1.0000 \\ 0.0000 & 0.7071 \end{matrix}$$

abs(C) ger

1.4142	3.1416
0	2.2361

sin(abs(C)) ger

0.9878	0.0000
0	0.7867

Man kan definiera egna elementvisa funktioner som lagras som MATLAB M-filer, t ex minmatlabfunk1(A) ger

$$\begin{bmatrix} \text{minmatlabfunk1}(0) & \text{minmatlabfunk1}(-3) \\ \text{minmatlabfunk1}(5) & \text{minmatlabfunk1}(1) \\ \text{minmatlabfunk1}(4) & \text{minmatlabfunk1}(-6) \end{bmatrix} = \begin{bmatrix} \log(5 + \sin 0) & \log(5 - \sin 3) \\ \log(5 + \sin 5) & \log(5 + \sin 1) \\ \log(5 + \sin 4) & \log(5 - \sin 6) \end{bmatrix}.$$

5.4 Matrisbyggande funktioner

ones(n) ger en $n \times n$ matris med ettor:

$$\text{ones}(n): \begin{bmatrix} 1 & 1 & \dots & 1 \\ 1 & 1 & \dots & 1 \\ \cdot & \cdot & \dots & \cdot \\ 1 & 1 & \dots & 1 \end{bmatrix}.$$

ones(n,m) ger en $n \times m$ matris med ettor.

ones(size(A)) ger en matris med ettor av samma storlek som **A**.

zeros(n) ger en $n \times n$ matris med nollor:

$$\text{zeros}(n): \begin{bmatrix} 0 & 0 & \dots & 0 \\ 0 & 0 & \dots & 0 \\ \cdot & \cdot & \dots & \cdot \\ 0 & 0 & \dots & 0 \end{bmatrix}.$$

eye(n) ger en $n \times n$ enhetsmatris

$$\text{eye}(n): \begin{bmatrix} 1 & 0 & \dots & 0 \\ 0 & 1 & \dots & 0 \\ \cdot & \cdot & \dots & \cdot \\ 0 & 0 & \dots & 1 \end{bmatrix} = \mathbf{I}.$$

5.4.1 Exempel

Betrakta ett linjärt ekvationssystem med 2 obekanta

$$\mathbf{Ax} = \mathbf{b}, \tag{4}$$

$$2x_1 + x_2 = 3$$

$$x_1 + 2x_2 = 4$$

där 2×2 koefficientmatrisen

$$\mathbf{A} = [a_{jk}] = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}.$$

har 2 rader och 2 kolonner. Kolonnvektorererna

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

och

$$\mathbf{b} = \begin{bmatrix} 3 \\ 4 \end{bmatrix}.$$

Ekvationssystemet $\mathbf{Ax} = \mathbf{b}$ löses i MATLAB med satsen

$\mathbf{x} = \mathbf{A} \setminus \mathbf{b}$

För att lösa (4), skriv en kommandofil `exempel241.m`

```
disp('Lös ekvationssystem med koefficientmatrisen och högerledet');
```

```
a = [2 1
```

```
1 2];
```

```
A
```

```
b=[3; 4];
```

```
b
```

```
x = A \ b
```

MATLABraden

```
>> exempel241
```

```
ger resultatet
```

Lös ekvationssystem med koefficientmatrisen och högerledet

```
A =
```

```
2 1
```

```
1 2
```

```
b =
```

```
3
```

```
4
```

```
x =
```

```
0.6667
```

```
1.6667
```

5.4.2 Exempel

Låt

$$\mathbf{A} = [a_{ij}] = \begin{bmatrix} 1 & 3 & 5 \\ 1 & 2 & 4 \\ 0 & 5 & 1 \end{bmatrix}$$

och

$$\mathbf{b} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} = \begin{bmatrix} 22 \\ 17 \\ 13 \end{bmatrix}.$$

vara en kvadratisk matris av typ 3×3 och en kolonnvektor. Då \mathbf{A} är en inverterbar matris, eftersom

$\det \mathbf{A} =$

$$\begin{vmatrix} 1 & 3 & 5 \\ 1 & 2 & 4 \\ 0 & 5 & 1 \end{vmatrix} = 1 \cdot (2 \cdot 1 - 4 \cdot 5) - 3 \cdot (1 \cdot 1 - 4 \cdot 0) + 5 \cdot (1 \cdot 5 - 2 \cdot 0) = -18 - 3 + 25 = 4.$$

Lös motsvarande ekvationssystemet

$$\mathbf{Ax} = \mathbf{b},$$

eller

$$\begin{aligned} x_1 + 3x_2 + 5x_3 &= 22 \\ x_1 + 2x_2 + 4x_3 &= 17 \\ 5x_2 + x_3 &= 13 \end{aligned} \tag{5}$$

med koefficientmatrisen \mathbf{A} och högerledet \mathbf{b} . där bildar man först matrisen \mathbf{A} och kolonnvektorn \mathbf{b} och sedan löser ekvationssystemet med satsen $\mathbf{x} = \mathbf{A} \setminus \mathbf{b}$ För att lösa (5), skriv en kommandofil `exempel17.m`

```
disp('Lös ekvationssystem med koefficientmatrisen och högerledet');
A = [1 3 5; 1 2 4; 0 5 1];
A
b = [22; 17; 13];
b
x = A\b
```

MATLABraden

```
>> exempel17
```

ger resultatet

Lös ekvationssystem med koefficientmatrisen och högerledet

```
A =
```

```
1 3 5
```

```
1 2 4
```

```
0 5 1
```

```
b =
```

```
22
```

```
17
```

```
13
```

```
x =
```

```
1.0000
```

```
2.0000
```

```
3.0000
```

Kolla att

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

är en rätt lösning genom att utföra Gausselimination.

(I) Subtrahera andra ekvationen i (5) från första ekvationen:

$$\begin{aligned}x_2 + x_3 &= 5 \\5x_2 + x_3 &= 13\end{aligned}\tag{6}$$

(II) Byta raderna 2 och 1 i (6) för att få koefficientmatrisen med maximelementet a_{11} (göra pivotering)

$$\begin{aligned}5x_2 + x_3 &= 13 \\x_2 + x_3 &= 5\end{aligned}\tag{7}$$

(III) Dividera första ekvationen (7) med 5:

$$\begin{aligned}x_2 + 1/5x_3 &= 13/5 \\x_2 + x_3 &= 5\end{aligned}\tag{8}$$

(IV) Subtrahera första ekvationen i (8) från andra ekvationen:

$$\begin{aligned}x_2 + 1/5x_3 &= 13/5 \\4/5x_3 &= 12/5\end{aligned}\tag{9}$$

Bestäm lösningen x_1, x_2, x_3 med substitution i omvänd ordning (bakåt substitution).

(V) andra ekvationen i (9) ger x_3 :

$$\begin{aligned}x_2 + 1/5x_3 &= 13/5 \\x_3 &= 3\end{aligned}\tag{10}$$

(VI) första ekvationen i (10) ger x_2 :

$$\begin{aligned}x_2 &= 13/5 - 1/5 \cdot 3 = 2 \\x_3 &= 3\end{aligned}$$

(VII) första ekvationen i begynnelsemet (5) ger x_1 :

$$\begin{aligned}x_1 &= 22 - 3 \cdot 2 - 5 \cdot 3 = 1 \\x_2 &= 2 \\x_3 &= 3\end{aligned}$$

Man kan också beräkna lösningen enligt Cramers regel för att se att den är en mödosam procedur.

$$x_1 = \frac{\begin{vmatrix} b_1 & a_{12} & a_{13} \\ b_2 & a_{22} & a_{23} \\ b_3 & a_{32} & a_{33} \end{vmatrix}}{\begin{vmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{vmatrix}} = \frac{\begin{vmatrix} 22 & 3 & 5 \\ 17 & 2 & 4 \\ 13 & 5 & 1 \end{vmatrix}}{\begin{vmatrix} 1 & 3 & 5 \\ 1 & 2 & 4 \\ 0 & 5 & 1 \end{vmatrix}} =$$

$$\begin{aligned}
&= \frac{22 \cdot (2 \cdot 1 - 4 \cdot 5) - 3 \cdot (17 \cdot 1 - 4 \cdot 13) + 5 \cdot (17 \cdot 5 - 2 \cdot 13)}{1 \cdot (2 \cdot 1 - 4 \cdot 5) - 3 \cdot (1 \cdot 1 - 4 \cdot 0) + 5 \cdot (1 \cdot 5 - 2 \cdot 0)} = \\
&= \frac{22 \cdot (-18) - 3 \cdot (-35) + 5 \cdot 59}{1 \cdot (-18) - 3 + 25} = \frac{-396 + 105 + 295}{-21 + 25} = \frac{4}{4} = 1.
\end{aligned}$$

$$x_2 = \frac{\begin{vmatrix} 1 & 22 & 5 \\ 1 & 17 & 4 \\ 0 & 13 & 1 \end{vmatrix}}{\begin{vmatrix} 1 & 3 & 5 \\ 1 & 2 & 4 \\ 0 & 5 & 1 \end{vmatrix}} = \frac{(17 - 4 \cdot 13) - (22 - 5 \cdot 13)}{4} = \frac{13 - 5}{4} = 2.$$

$$\begin{aligned}
x_3 &= \frac{\begin{vmatrix} 1 & 3 & 22 \\ 1 & 2 & 17 \\ 0 & 5 & 13 \end{vmatrix}}{\begin{vmatrix} 1 & 3 & 5 \\ 1 & 2 & 4 \\ 0 & 5 & 1 \end{vmatrix}} = \\
&= \frac{(2 \cdot 13 - 5 \cdot 17) - (3 \cdot 13 - 5 \cdot 22)}{4} = \frac{-13 + 5(22 - 17)}{4} = 3.
\end{aligned}$$

5.4.3 Problem

Skriv en $n \times n$ matris med elementen

- (a) $n = 3$; $a_{jj} = 1$ och $a_{jk} = 0$, $j \neq k$. (en diagonalmatris: enhetsmatris).
- (b) $n = 5$; $a_{jj} = 4$ och $a_{jk} = 0$ på alla övriga platser så när som på (45) där $a_{jk} = 10$.
- (c) $n = 4$; den första kolonnen består av nollor och varje följande k kolonn består av talen k , $k = 1, 2, 3$.

Skriv kommandofiler som assemblerar matriserna (a), (b) och (c).

5.4.4 Problem

Bestäm dimensionen av matrisen

$$\mathbf{A} = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 2 & 1 & 0 & 0 \\ 0 & 1 & 3 & 1 & 0 \\ 0 & 0 & 1 & 4 & 1 \\ 0 & 0 & 0 & 1 & 5 \end{bmatrix}$$

och skriv elementen (t ex, $a_{11} = 5$, etc.). Skriv kommandofilen som assemblerar matrisen \mathbf{A} .

5.4.5 Problem

Skriv en funktionsfil `problem245.m` som löser ett linjärt ekvationssystem med 2 obekanta

$$\begin{aligned}
ax_1 + bx_2 &= c_1 \\
bx_1 + ax_2 &= c_2,
\end{aligned}$$

$a \neq b$. Skriv koefficientmatrisen. Utför funktionsfilen med olika indata. Kolla resultat.

5.4.6 Problem

Skriv en funktionsfil `problem246.m` som löser det linjära ekvationssystemet med symmetriska trediodagonala koefficientmatrisen

$$\mathbf{A} = [a_{jk}] = \begin{bmatrix} a & b & 0 & 0 & 0 \\ b & a & b & 0 & 0 \\ 0 & b & a & b & 0 \\ 0 & 0 & b & a & b \\ 0 & 0 & 0 & b & a \end{bmatrix}, \quad a \neq b.$$

Skriv koefficientmatrisen. Utför funktionsfilen med olika indata. Kolla resultat.

5.4.7 Problem

Skriv en funktionsfil `problem247.m` som löser linjära ekvationssystemet

$$\begin{aligned} x_1 + x_2/3 &= 5/3 \\ x_1/3 + x_2 &= 7/3 \end{aligned} \tag{11}$$

med `JI` och kontrollerar att `JI` konvergerar för varje begynnelsevektor $\mathbf{x}^{(0)}$ (matrisnormen av iterationsmatrisen $\mathbf{Q} = \mathbf{I} - \mathbf{A}$ är mindre än 1: $\|\mathbf{Q}\| < 1$). Utför funktionsfilen med olika indata. Kolla resultat.

5.5 Vektorer och delmatriser

`i:k` ger en följd av värden från i till k i steg om ett, dvs

$i, i + 1, \dots, k$;

Om $i > k$, får vi ingen värdeföljd utan tomma mängden. Om i och k är hela tal ($k > i$), då består följden av $k - i + 1$ tal.

`i:j:k` ger en följd av värden från i till k i steg om j , dvs

$i, i + j, i + 2j, \dots, k$

För $j = 0$ returneras tomma mängden.

5.5.1 Exempel

MATLAB kommandon

```
>> xx = 0:1:10;
```

```
>> xx
```

ger 11 hela tal (radvektor)

```
xx =
```

```
0 1 2 3 4 5 6 7 8 9 10
```

Man kan använda kolon för att definiera matriser.

5.5.2 Exempel

MATLAB kommandot

```
>> Mat1 = [2:4 0.1:1:2.1; 1:6]
returnerar en matris
Mat1 =
```

```
2.0000  3.0000  4.0000  0.1000  1.1000  2.1000
1.0000  2.0000  3.0000  4.0000  5.0000  6.0000
```

5.5.3 Exempel: Generering av en funktionstabell (sinus)

MATLAB kommandon

```
>> a = 0.0; b = 2*pi; n = 10; h = (b-a)/n; x = (a:h:b)'; y = sin(x); Ftable = [x,y]
genererar funktionstabellen för sin x som består av två 10-element kolonnvektorer x och y
Ftable =
```

```
0      0
0.6283 0.5878
1.2566 0.9511
...
```

linspace(a,b) returnerar en vektor med 100 element i $[a, b]$ (som delar $[a, b]$ i lika stora intervall).

linspace(a,b,n) returnerar en vektor med n element i $[a, b]$ (som delar $[a, b]$ i lika stora intervall).

5.6 Nät och noder

Låt a och b , $a < b$, var två givna tal. $n + 1$ punkter

$$x_j = a + jh, \quad j = 0, 1, 2, \dots, n, \quad h = \frac{b - a}{n},$$

kallas *ett likformigt nät* och x_j kallas nätets *noder*. Observera att

$$x_0 = a, \quad x_n = b.$$

$x_0 = a$ och $x_n = b$ kallas *randnoder* (yttre noder), och $n - 1$ punkterna x_1, x_2, \dots, x_{n-1} kallas *inre noder*.

Man kan skriva ett nät som en $n + 1$ -dimensionell radvektor

$$\mathbf{x} = [x_j] = [x_0, x_1, \dots, x_{n-1}, x_n].$$

MATLAB kommandon

```
>> h = (b-a)/n; x = a:h:b;
>> x = linspace(a,b,n);
returnerar nätet x [ $a, b$ , och  $n$  (eller  $h$ ) är givna tal].
```

5.6.1 Problem

Generera ett likformigt nät av n punkter som tillhör intervallet $[a, b]$. Använd kommandon `a:h:b` och `linspace(a,b,n)`.

- (a) $a = -\pi/2, b = \pi/2, n = 10$
- (b) $a = -10, b = 10, n = 20$
- (c) $a = -2, b = 2, n = 20$

Delmatriser (Kommandon 37)

- `A(i,j)` ger elementet på plats (i, j) i matrisen **A**
- `A(:,j)` ger den j :te kolonnen i **A**
- `A(i,:)` ger den i :te raden i **A**
- `A(:,j:k)` ger delmatrisen till **A** bestående av kolonnerna $j, j + 1, \dots, k$
- `A(i:k,:)` ger delmatrisen till **A** bestående av raderna $i, i + 1, \dots, k$
- `A(i:k,j:l)` ger delmatrisen till **A** bestående av raderna $i, i + 1, \dots, k$ och kolonnerna $j, j + 1, \dots, l$
- `A(:,:)` ger tillbaka **A** själv.

5.6.2 Exempel

Antag att **Ftable** i Exempel 5.5.3 är definierad (som en matris bestående av 10 rader och 2 kolonner).

- (a) Då kommandot
`>> Submatrix = Ftable(2:4,:)`
returnerar en delmatris till matrisen **Ftable** bestående av raderna 2, 3 och 4
`Ftable =`

```
0.6283  0.5878
1.2566  0.9511
1.8850  0.9511
```

5.6.3 Exempel

Flera kommand i samma raden:

$$x = 7; \quad y = 4.6735; \quad z = x + y;$$

Man kan skriva ett långt MATLAB kommand (t ex en radvektor `mat1`) i flera MATLAB rader (med hjälp av `...`):

$$\text{mat1} = [1.2 \quad 1.2 \quad 1.4 \quad -1.5 \dots \\ -1.1 \quad -1.3 \quad 1.7]$$

5.7 Variabelstorlek

`size(A)`

$$[m, n] = \text{size}(A)$$

returnerar en radvektor; m är antalet rader och n är antalet kolonner.

5.7.1 Exempel

Kommandot

```
thesize = size(A)
```

där A är en 2×3 matris

```
thesize =
```

```
2 3
```

5.7.2 Problem

Generera värdetabellen för funktionen $f(x)$ beräknad i ett likformigt nät av n punkter $x = x_i$ som tillhör intervallet $[a, b]$.

(a) $f(x) = 2 \cos x$, $a = -\pi/2$, $b = \pi/2$, $n = 10$.

(b) $f(x) = \frac{x - 1.96}{x^2 + 1.15}$, $a = -10$, $b = 10$, $n = 20$.

(c) $f(x) = 2x^3 + x^2 + 5x + 17$, $a = -2$, $b = 2$, $n = 20$.

5.7.3 Problem

- (a) Returnera den första kolonnen av värdetabellen för funktionen $f(x)$ i problemet (a) ovan.
(b) Returnera den tredje raden av tabellen för funktionen $f(x)$ i problemet (b) ovan.
(c) Returnera deltabellen för funktionen $f(x)$ i problemet (c) ovan som består av rader 11 till 15.

6 2D-grafik

I MATLAB kan man rita ut funktions graf som en mängd av ordnade koordinatpar.

`plot(x,y)` ritar grafen till funktionen $y(x)$, dvs vektorn y mot den givna vektorn x

`plot(y)` ritar de ordnade talparen (j, y_j) , dvs vektorn $y = \{y_j\}$ mot hela talen j

`plot(z)` ritar $(\Re z_j, \Im z_j)$, dvs koordinatparen av den komplexa talen (vektorn) $z = \{z_j\}$,
 $z_j = x_j + iy_j$, $i = \sqrt{-1}$

`plot(A, x)` ritar en rad- eller kolonn- vektor x mot rader eller kolonner av $m \times n$ matrisen

A

6.0.4 Exempel

```
x = [-4 -2 0 1 3 5]
y = [16 4 0 1 9 25]
```

```
x = -pi:0.05:pi;
plot(x,sin(x).*cos(x),'o')
```

6.0.5 Problem

Rita funktionerna (polynom) med (1) två, (2) tre, och (3) fyra Taylors formel termer i Problem 3.4.2. Rita alla kurvor i samma diagram.

7 Strängar i MATLAB

En sträng i MATLAB definieras med hjälp av apostrofer
variabelnamn = 'text'

7.0.6 Exempel

(a)

```
namn = 'John Smith'  
ger på skärmen  
namn =  
John Smith
```

(b)

```
name1 = 'Joan'; name2 = 'John'; heart = 'is in love with';  
sentence = [name1 ' ' heart ' ' name2]  
ger på skärmen  
sentence =  
Joan is in love with John
```

8 Utskrift och inläsning

`disp(A)` skriver ut matrisen **A** utan att skriva ut matrisens namn. Om **A** är en sträng, så skrivs alltså texten ut.

Inläsning från terminalen kan göras med kommandot `input`.

`input(str)` skriver ut texten i strängen **str** på skärmen och väntar på inmatning (`input`) från tangentbordet. Det inlästa värdet returneras av `input`.

`input(str,'s')` skriver ut texten i strängen **str** på skärmen och returnerar värdet av inmatningen från tangentbordet som en sträng.

Example 5.4

(a) Läs in ett reelt tal `x`:

```
x = input('Ge talet x: ')  
ger  
Ge talet x : 2.0944  
x =  
2.0944
```

(b) Läs in en matris **A**

En semikolon efter kommandot `input` medför att det som läses in inte skrivs ut:

```
A = input('Ge matrisen A radvis: ');
ger
Ge matrisen A radvis : [1 2; 3 5]
```

8.0.7 Exempel

Betrakta MATLAB kommandon i filen **draw.m**

```
% Program plotting functions
disp('This program plots f(x) in the interval [a,b]');
ftext = input('Give a function, e.g. sin(x) : ','s');
lb = input('Give lower bound : ');
ub = input('Give upper bound : ');
% Plot the function
clf; fplot(ftext, [lb ub]);
```

MATLAB kommandot

`fplot(ftext, lim)` ritar ut funktionen som specificeras av strängen `ftext` (en standardfunktion eller en av användaren definierad funktion som ligger lagrad på M-filen `ftext.m`) i intervallet `[lb ub]` Den två element vektorn `lim = [lb ub]` talar om gränserna där funktionen skall ritas.

Kommandot `draw` utför programmet. Betrakta ett exempel.

```
>> draw
This program plots f(x) in the interval [a,b]
Give a function, e.g. sin(x) : 2.*x.*cos(x.*x)
Give lower bound, a : 0
Give upper bound, b : 2*pi
<plot of 2x cos x^2 >
```

8.0.8 Problem

Använd `draw` för att rita funktionerna (polynom) i Problem 1.9(a), (b), (c) med (1) två, (2) tre, och (3) fyra Taylors formel termer.

9 Komplexa tal och funktioner

Funktioner på komplexa tal $z = x + iy$, $i = \sqrt{-1}$:

`real(z)` ger reeldelen av z , $x = \Re z$
`imag(z)` ger imaginärdelen av z , $y = \Im z$
`abs(z)` ger absolutbeloppet av z , $r = |z| = \sqrt{x^2 + y^2}$
`conj(z)` ger komplexkonjugatet av z , $\bar{z} = x - iy$
`angle(z)` ger fasvinkeln θ av $z = re^{i\theta}$.
 ...

9.0.9 Exempel

Betrakta det komplexa talet $z = 1 + 2i$.

- (a) Reeldelen av $z = 1 + 2i$ och imaginärdelen av z är
realpart = real(z), imagpart = imag(z)
Kommandon returnerar (på skärmen)

realpart =

1

imagpart =

2

- (b) Komplexkonjugatet av z ,
conjugate = conj(z)
returnerar (på skärmen)

conjugate =

1.0000 - 2.0000i

9.0.10 Problem

Betrakta två komplexa talen $a = 2 + 3i$ och $b = 1 - i$. Bestäm $\Re z$, $\Im z$, $|z|$, \bar{z} och $\arg z$ av följande komplexa tal:

- (a) $z = a^2 - 2b^2$;
(b) $z = \sin a + \frac{3-i}{2+b}$

10 Utskriftformat

Antag att

$p = 1 + 1/3$;

och att vi först definierar ett format och sedan skriver ut p på skärmen (t ex, vi skriver

>> format long

>> p

och trycker 'return')

format short 1.3333 4 decimaler

format long 1.33333333333333 14 decimaler

format short e 1.3333e + 00 4 decimaler

format long e 1.33333333333333e + 00 15 decimaler

format rat 4/3 ett rationellt tal

format + + (ett positivt tal; - ett negativt tal)

10.1 Spara resultat, lagra och hämta data

`save` sparar alla variabler på filen **matlab.mat**

`save filename` sparar alla variabler på filen **filename.mat**

`save filename v1 v2` sparar variablerna v_1, v_2 etc. på filen **filename.mat**

`save filename v -ascii` sparar värden i variabeln v i läsbart ASCII-format på filen **filename.mat** Skriver ut 8 decimaler.

`load` hämtar alla variabler från filen **matlab.mat**.

`load filename` hämtar alla variabler från filen **filename.mat**

För att spara en bild (figure) (t ex efter kommandot `plot`), använd kommandot `save as` i 'figure' fönstret (menu window), tryck `save as` , och spara som en M-fil *.m t ex **figure1.m**. Kommandot

```
>> figure1
```

hämtar den här bilden.

10.1.1 Exempel

Antag att ASCII-filen **A.dat** skapas med en editor eller med ett utskriftsats i ett program:

```
1 4 5
4 2 9
```

I MATLAB får vi då matrisen **A** med kommandot

```
load A.dat, A
```

```
A =
```

```
1 4 5
4 2 9
```

10.1.2 Problem

Spara alla matriser i Exempel 2.4.2 och Problem 2.4.3 i olika ASCII-filer *.dat och sedan hämta de här filerna.

11 Relationsoperatorer

`<` mindre än

`<=` mindre än eller lika med

`>` större än

`>=` större än eller lika med

`==` lika med

`~=` skilt från

Relationsoperatorerna jämför motsvarande element och skapar en matris (ett tal) av samma storlek innehållande enbart ettor och nollor. Elementen är

1 om jämförelsen är **sann**

0 om jämförelsen är **falsk**

11.0.3 Exempel

Kommandon

```
a=2; b=3; a<b
```

```
ger
```

```
ans =
```

```
1
```

```
a>b
```

```
ans =
```

```
0
```

```
etc.
```

Man kan jämföra uttryck; t ex, kommandon

```
a=2; b=3; (a+b)*(a*b + 3) > 42
```

```
ger
```

```
ans =
```

```
1
```

eftersom talet $(2+3)*(2*3 + 3) = 45 > 42$

Man kan också skriva

```
a=2; b=3; c = ((a+b)*(a*b + 3) > 42);
```

```
c
```

```
c =
```

```
1
```

12 Logiska funktioner

& **och** (logisk multiplikation)

| **eller** (logisk addition)

~ **negation** (logisk minus)

xor **exklusivt eller**

T ex, kommandon

```
a=1; b=0; a&b
```

```
ger
```

```
ans =
```

```
0
```

```
a|b
```

```
ans =
```

```
1
```

```
xor(a,b)
```

```
ans =
```

```
1
```

```
~b
```

```
ans =
```

```
1
```

13 Programmering i MATLAB

13.1 Villkorssatser

Villkorskommandot i MATLAB är `if`. Den enklaste formen av if-sats är

```
if logiskt uttryck
    satsgrupp
end
```

eller, i en rad,

```
if logiskt uttryck, satsgrupp, end
```

Kommandon (satserna) i *satsgrupp* utförs enbart om *logiskt uttryck* är **sant**.

Logiska uttryck kan vara en skalär (ett tal), en vektor, eller en matris, och ett logiskt uttryck sägs vara **sant** om alla dess element är nollskilda.

13.1.1 Exempel

Kommandon

```
if x==2
    x = 2*x
end
```

fördubblar talet x om $x = 2$.

'return' och 'x' efter 'end' ger det givna x-värdet om $x \neq 2$ och värdet

```
x =
    4
```

om $x = 2$

13.1.2 Exempel

Antag att vi har en $\mu \times \nu$ matris \mathbf{A} . Om alla element i \mathbf{A} 's första kolonn är nollor så ska den tas bort:

```
if A(:,1) == 0
    A(:,1) = A(1:m,2:n)
end
```

Observera att

$A(:,j)$ returnerar j kolonn av den givna matrisen \mathbf{A}

och

$A(i:k,j:l)$ ger delmatrisen till \mathbf{A} bestående av raderna $i, i + 1, \dots, k$ och kolonnerna $j, j + 1, \dots, l$.

if-satsen används i kombination med else-satsen och elseif-satsen:

```
if logiskt uttryck
    satsgrupp 1
else
    satsgrupp 2
end
```

Kommandon (satserna) i *satsgrupp 1* utförs enbart om *logiskt uttryck* är **sant** annars utförs kommandon i *satsgrupp 2* (enbart om *logiskt uttryck* är **falskt**).

I if-satsen

```

if logiskt uttryck 1
    satsgrupp 1
elseif logiskt uttryck 2
    satsgrupp 2
end

```

utförs satserna i *satsgrupp 1* om *logiskt uttryck 1* är **sant**. Satserna i *satsgrupp 2* utförs om *logiskt uttryck 1* är **falskt** och *logiskt uttryck 2* är **sant**.

elseif-satsen kräver inget end men det gör else och if

En annan variant kan se ut så här:

```

if logiskt uttryck 1
    satsgrupp 1
elseif logiskt uttryck 2
    satsgrupp 2
else
    satsgrupp 3
end

```

13.2 Repetitionssatser

MATLAB har två olika kommandon för repeterad exekvering av satser, **for** och **while**.

Kommandot **for** repeterar en satsgrupp n gånger: . Slutet på gruppen markeras med **end**:

```

for variabel = uttryck
    satsgrupp
end

```

eller, i en rad,

```

for variabel = uttryck, satsgrupp, end

```

Man kan exekvera repetitionssatser rekursivt:

```

for variabel I = uttryck A
    satsgrupp 1
    for variabel II = uttryck B
        satsgrupp 2
    end
    satsgrupp 3
end

```

13.2.1 Exempel 12.3

Betrakta en 5×5 symmetrisk tre-diagonal matris

$$\mathbf{A} = \begin{bmatrix} 5 & 1 & 0 & 0 & 0 \\ 1 & 5 & 1 & 0 & 0 \\ 0 & 1 & 5 & 1 & 0 \\ 0 & 0 & 1 & 5 & 1 \\ 0 & 0 & 0 & 1 & 5 \end{bmatrix}.$$

Skriv flera olika repetitionssatser ('for-loop') som bildar symmetriska trediodagonal matrisen **A**:

Version 1

```
A = zeros(5);
for k = 1:4
    A(k,k) = 5;
    A(k,k+1) = 1;
    A(k+1,k) = 1;
end % in k
A(5,5) = 5;
```

eller steg-för-steg:

```
k = 1
      5 0 0 0 0    5 1 0 0 0    5 1 0 0 0
      0 0 0 0 0    0 0 0 0 0    1 0 0 0 0
      0 0 0 0 0    0 0 0 0 0    0 0 0 0 0
      0 0 0 0 0    0 0 0 0 0    0 0 0 0 0
      0 0 0 0 0    0 0 0 0 0    0 0 0 0 0

k = 2
      5 1 0 0 0    5 1 0 0 0    5 1 0 0 0
      1 5 0 0 0    1 5 1 0 0    1 5 1 0 0
      0 0 0 0 0    0 0 0 0 0    0 1 0 0 0
      0 0 0 0 0    0 0 0 0 0    0 0 0 0 0
      0 0 0 0 0    0 0 0 0 0    0 0 0 0 0
```

etc.

Version 2

```
A = [];
for k = 1:5
    for j = 1:5
        if k == j
            A(k,k) = 5;
        elseif abs(k-j) == 1
            A(k,j) = 1;
        else A(k,j) = 0;
        end % in if
    end % in j
end % in k
```

Observera att

(i) k-repetitionssatsen (k-loop) exekveras i rader och j-repetitionssatsen (j-loop) i kolonner.
(ii) Satsgruppen 1 exekveras om *det logiska uttrycket 1* är **sant**. Satsgruppen 2 exekveras om *det logiska uttrycket 1* är **falskt** och *det logiska uttrycket 2* är **sant**.

Beskriva exekvering steg-för-steg:

A = [] =

1 (1, 1)	2 (1, 2)	3 (1, 3)	4 (1, 4)	5 (1, 5)
6 (2, 1)	7 (2, 2)	8 (2, 3)	9 (2, 4)	10 (2, 5)
11 (3, 1)	12 (3, 2)	13 (3, 3)	14 (3, 4)	15 (3, 5)
16 (4, 1)	17 (4, 2)	18 (4, 3)	19 (4, 4)	20 (4, 5)
21 (5, 1)	22 (5, 2)	23 (5, 3)	24 (5, 4)	25 (5, 5)

Steg 1 Sätt k=1 (första raden); exekvera j-loop: sätt j=1 (första kolonnen); kolla om k är lika med j, här k=1 och j=1, och satsgruppen 1 A(1,1) = 5 exekveras eftersom det logiska uttrycket k == j är sant; sedan end i if-satsen. **1** ger

5	2 (1, 2)	3 (1, 3)	4 (1, 4)	5 (1, 5)
6 (2, 1)	7 (2, 2)	8 (2, 3)	9 (2, 4)	10 (2, 5)
11 (3, 1)	12 (3, 2)	13 (3, 3)	14 (3, 4)	15 (3, 5)
16 (4, 1)	17 (4, 2)	18 (4, 3)	19 (4, 4)	20 (4, 5)
21 (5, 1)	22 (5, 2)	23 (5, 3)	24 (5, 4)	25 (5, 5)

Steg 2 <k=1> Sätt j=j+1=2; kolla om k är lika med j, här k=1 och j=2, det logiska uttrycket k == j är falskt och satsgruppen 1 A(1,1) = 5 exkveras inte; vi exkverar satsgruppen 2 med abs(k-j) = abs(1-2) = 1, och satsgruppen 2 A(1,2) = 1 exekveras eftersom det logiska uttrycket abs(k-j) == 1 är sant; sedan end i if-satsen. **2** ger

5	1	3 (1, 3)	4 (1, 4)	5 (1, 5)
6 (2, 1)	7 (2, 2)	8 (2, 3)	9 (2, 4)	10 (2, 5)
11 (3, 1)	12 (3, 2)	13 (3, 3)	14 (3, 4)	15 (3, 5)
16 (4, 1)	17 (4, 2)	18 (4, 3)	19 (4, 4)	20 (4, 5)
21 (5, 1)	22 (5, 2)	23 (5, 3)	24 (5, 4)	25 (5, 5)

Steg 3 <k=1> Sätt j=j+1=3; kolla om k är lika med j, här k=1 och j=3, det logiska uttrycket k == j är falskt och satsgruppen 1 A(1,1) = 5 exkveras inte; vi exkverar satsgruppen 2 med abs(k-j) = abs(1-3) = 2, det logiska uttrycket abs(k-j) == 1 är falskt, och satsgruppen 2 A(1,3) = 1 exkveras inte och satsgruppen 3 A(1,3) = 0 exkveras; sedan end i if-satsen. **3** ger

5	1	0	4 (1, 4)	5 (1, 5)
6 (2, 1)	7 (2, 2)	8 (2, 3)	9 (2, 4)	10 (2, 5)
11 (3, 1)	12 (3, 2)	13 (3, 3)	14 (3, 4)	15 (3, 5)
16 (4, 1)	17 (4, 2)	18 (4, 3)	19 (4, 4)	20 (4, 5)
21 (5, 1)	22 (5, 2)	23 (5, 3)	24 (5, 4)	25 (5, 5)

Steg 4 och **Steg 5** sammanfaller med **Steg 3**. end i j-repetitionssatsen. Vi har skapat den första raden 5 1 0 0 0 (k = 1, j = 1, 2, 3, 4, 5)

5	1	0	0	0
7 (2, 1)	7 (2, 2)	8 (2, 3)	9 (2, 4)	10 (2, 5)
11 (3, 1)	12 (3, 2)	13 (3, 3)	14 (3, 4)	15 (3, 5)
16 (4, 1)	17 (4, 2)	18 (4, 3)	19 (4, 4)	20 (4, 5)
21 (5, 1)	22 (5, 2)	23 (5, 3)	24 (5, 4)	25 (5, 5)

Steg 6 Sätt k=k+1=2; exekvera j-loop; sätt j=1; kolla om k är lika med j, här k=2 och j=1, det logiska uttrycket k == j är falskt och satsgruppen 1 A(1,1) = 5 exekveras inte; vi exkverar

satsgruppen 2 med $\text{abs}(k-j) = \text{abs}(2-1) = 1$, det logiska uttrycket $\text{abs}(k-j) == 1$ är sant, och satsgruppen 2 $A(2,1) = 1$ exekveras; sedan end i if-satsen. **6** ger

5	1	0	0	0
1	7 (2, 2)	8 (2, 3)	9 (2, 4)	10 (2, 5)
11 (3, 1)	12 (3, 2)	13 (3, 3)	14 (3, 4)	15 (3, 5)
16 (4, 1)	17 (4, 2)	18 (4, 3)	19 (4, 4)	20 (4, 5)
21 (5, 1)	22 (5, 2)	23 (5, 3)	24 (5, 4)	25 (5, 5)

Steg 7 $\langle k=2 \rangle$ Sätt $j=j+1=2$; kolla om k är lika med j , här $k=2$ och $j=2$, det logiska uttrycket $k == j$ är sant och satsgruppen 1 $A(1,1) = 5$ exekveras, sedan end i if-satsen. **7** ger

5	1	0	0	0
1	5	8 (2, 3)	9 (2, 4)	10 (2, 5)
11 (3, 1)	12 (3, 2)	13 (3, 3)	14 (3, 4)	15 (3, 5)
16 (4, 1)	17 (4, 2)	18 (4, 3)	19 (4, 4)	20 (4, 5)
21 (5, 1)	22 (5, 2)	23 (5, 3)	24 (5, 4)	25 (5, 5)

Steg 8 $\langle k=2 \rangle$ Sätt $j=j+1=3$; kolla om k är lika med j , här $k=2$ och $j=3$, det logiska uttrycket $k == j$ är falskt och satsgruppen 1 $A(1,1) = 5$ exekveras inte, vi exekverar satsgruppen 2 med $\text{abs}(k-j) = \text{abs}(2-3) = 1$, det logiska uttrycket $\text{abs}(k-j) == 1$ är sant och satsgruppen 2 $A(2,3) = 1$ exekveras; sedan end i if-satsen. **8** ger

5	1	0	0	0
1	5	1	9 (2, 4)	10 (2, 5)
11 (3, 1)	12 (3, 2)	13 (3, 3)	14 (3, 4)	15 (3, 5)
16 (4, 1)	17 (4, 2)	18 (4, 3)	19 (4, 4)	20 (4, 5)
21 (5, 1)	22 (5, 2)	23 (5, 3)	24 (5, 4)	25 (5, 5)

9 $\langle k=2 \rangle$ Sätt $j=j+1=4$; kolla om k är lika med j , här $k=2$ och $j=4$, det logiska uttrycket $k == j$ är falskt och satsgruppen 1 $A(1,1) = 5$ exekveras inte; vi exekverar satsgruppen 2 med $\text{abs}(k-j) = \text{abs}(2-4) = 2$, det logiska uttrycket $\text{abs}(k-j) == 1$ är falskt, satsgruppen 2 $A(2,4) = 1$ exekveras inte och satsgruppen 3 $A(2,4) = 0$ exekveras; sedan end i if-satsen. **9** ger

5	1	0	0	0
1	5	1	0	10 (2, 5)
11 (3, 1)	12 (3, 2)	13 (3, 3)	14 (3, 4)	15 (3, 5)
16 (4, 1)	17 (4, 2)	18 (4, 3)	19 (4, 4)	20 (4, 5)
21 (5, 1)	22 (5, 2)	23 (5, 3)	24 (5, 4)	25 (5, 5)

Steg 10 sammanfaller med **Steg 9**. end i j-repetitionssatsen. Vi har skapat den andra raden 1 5 1 0 0 ($k = 2, j = 1, 2, 3, 4, 5$)

5	1	0	0	0
1	5	1	0	0
11 (3, 1)	12 (3, 2)	13 (3, 3)	14 (3, 4)	15 (3, 5)
16 (4, 1)	17 (4, 2)	18 (4, 3)	19 (4, 4)	20 (4, 5)
21 (5, 1)	22 (5, 2)	23 (5, 3)	24 (5, 4)	25 (5, 5)

...

Steg 25 $\langle k=5 \rangle$ (den femdje raden); Sätt $j=5$ (den femdje kolonnen); kolla om k är lika med j , här $k=5$ och $j=5$ det logiska uttrycket $k == j$ är sant och satsgruppen 1 $A(5,5) = 5$ exekveras, sedan end i if-satsen, end i j -repetitionssatsen och end i k -repetitionssatsen.

Resultatet är

$A =$

$$\begin{matrix} 5 & 1 & 0 & 0 & 0 \\ 1 & 5 & 1 & 0 & 0 \\ 0 & 1 & 5 & 1 & 0 \\ 0 & 0 & 1 & 5 & 1 \\ 0 & 0 & 0 & 1 & 5 \end{matrix}$$

14 Nollställen till funktioner i MATLAB

Skriv olika MATLAB satser som beräknar nollställen till en given funktion

$$f(x) = 0. \tag{12}$$

Funktionen $f(x)$ sparas (representeras) i en M-fil, t ex **ff.m**.

Skriv (12) på formen

$$x = g(x), \tag{13}$$

där $g(x)$ är en deriverbar funktion, och iterera

$$x_{n+1} = g(x_n), \quad n = 0, 1, 2, \dots \tag{14}$$

med en startgissning $x_0 = x_0$ (börja med $n = 0$ och repetera (14) flera gånger) sådan att

$$|g(x)'| < 1, \quad x \in (x_0 - \delta, x_0 + \delta). \tag{15}$$

Funktionen $g(x)$ sparas (representeras) i en M-fil, t ex **gg.m**.

När man använder Newtons metod, itererar man enligt

$$x_{n+1} = G(x_n), \quad G(x_n) = \frac{f(x_n)}{f'(x_n)}, \quad n = 0, 1, 2, \dots \tag{16}$$

Funktionen

$$G(x) = \frac{f(x)}{f'(x)} \tag{17}$$

beräknas och sparas i en M-fil, t ex **gnewt.m**.

För att avbryta (14) eller (16) vid ett lämpligt x_n , $n = 1, 2, \dots$, betraktar man differensen

$$d_{n,n-1} = |x_n - x_{n-1}| \tag{18}$$

och värdet

$$F_n = |f(x_0)|. \tag{19}$$

Ligger $d_{n,n-1}$ inom den från början uppställda toleransgränsen ϵ_x , dvs

$$d_{n,n-1} < \epsilon_x, \quad (20)$$

eller blir funktionsvärdet tillräckligt litet, dvs

$$F_n < \epsilon_f, \quad (21)$$

har man löst den numeriska uppgiften (12) att beräkna ett nollställe (som ligger i en omgivning till x_0) till en given funktion f . I praktiken $\epsilon_f < \epsilon_x$ och $\epsilon_f = 10^{-m}\epsilon_x$, $m = 2, 3, \dots$, t ex $\epsilon_x = 10^{-4}$ och $\epsilon_f = 10^{-6}$.

Ibland kallas (20) och (21) *stopregel*.

Bestämning av nollställen till funktioner av en variabel kan göras med MATLAB kommandot `fzero`. För polynom bör man istället använda MATLAB kommandot `roots`. Algoritmen för `fzero` är iterativ och kräver att användaren ger ett x -värde nära det nollställe man söker.

`fzero(fkn,x0)` returnerar ett av nollställena till funktionen som är angiven i strärngen **fkn**. Kommandot kräver att man ger en startgissning x_0 . Approximationens relativa fel är MATLAB variabeln *eps*.

`fzero(fkn,x0,tol)` returnerar ett av nollställena till funktionen som är angiven i strärngen **fkn** med relativa felet *tol*. Kommandot kräver att man ger en startgissning x_0 .

14.0.2 Exempel

För att bestämma en skärmingspunkt mellan funktionen $\sin x$ och den linära funktionen $2x - 2$, dvs hitta en lösning till ekvationen $\sin x = 2x - 2$, eller

$$\text{sinm}(x) \equiv \sin x - 2x + 2 = 0,$$

bildar vi en M-fil **sinm.m** som beräknar funktionen

$$\text{sinm}(x) = \sin x - 2x + 2$$

```
function s = sinm(x)
s = sin(x) - 2.*x + 2;
```

Att rita ut kurvan kan vara ett bra sätt att finna en startgissning x_0 , och vi skriver

```
fplot('sinm', [-10,10])
grid on; title('Funktionen sin(x) - 2.*x + 2');
```

Vi ser att $x_0 = 2$ kan vara en god startgissning, och skriver

```
xzero = fzero('sinm',2)
```

som ger skärmingspunkten

$$\begin{aligned} xzero = \\ 1.4987 \end{aligned}$$

som alltså är en lösning till ekvationen $\sin x = 2x - 2$.

14.0.3 Problem

Lös ekvationen

$$f(x) \equiv x^3 - 5.00x^2 + 1.01x - 1.88 = 0$$

med iterationsmetoden (14).

Lösning. Skriv

$$f(x) = x^3 - 5.00x^2 + 1.01x - 1.88$$

på formen

$$x = g(x) = \frac{5.00x^2 - 1.01x - 1.88}{x^2}.$$

Iterationsföljden $x_{n+1} = g(x_n)$, $n = 0, 1, 2, \dots$, med startgissningen $x_0 = 1$ är

$$x_0 = 1, \quad x_1 = 2.110, \quad x_2 = 4.099, \quad x_3 = 4.612, \quad x_4 = 4.6952, \quad x_5 = 4.700;$$

Iterationsföljden med startgissningen $x_0 = 5$ ger samma rot

$$x_0 = 5, \quad x_1 = 4.723, \quad x_2 = 4.702, \quad x_3 = 4.700.$$

14.0.4 Problem

Skriv Newtons iterationsföljd och bestäm $7^{1/3}$.

Lösning. $7^{1/3}$ är ett nollställe till funktionen $x^3 - 7$.

Newtons iterationsföljd

$$x_{n+1} = G(x_n), \quad G(x_n) = x_n - \frac{f(x_n)}{f'(x_n)}, \quad n = 0, 1, 2, \dots$$

ger, med $f(x) = x^3 - 7$,

$$f'(x) = 3x^2; \quad x_{n+1} = \frac{2x_n^3 + 7}{3x_n^2}, \quad n = 0, 1, 2, \dots$$

Iterationsföljden med startgissningen $x_0 = 1$ ger resultatet

$$x_4 = 1.912931.$$

Kolla det med `fzero` MATLAM Kommandot.

14.0.5 Problem

Bestäm den positiva rotten till andragradsekvationen $x^2 - 2 = 0$ med toleransgränserna (i) $\epsilon_x = 0.001$ och (ii) $\epsilon_x = 0.00001$. Jämför antalet operationer. Rita kurvor. Kolla resultatet med `fzero` MATLAM kommandot och `roots` MATLAM kommandot.

15 Integralberäkning i MATLAB

I MATLAB kan vi beräkna integraler av typen

$$J = \int_a^b y(x) dx.$$

`trapz(x,y)` beräknar integralen av \mathbf{y} som en funktion av \mathbf{x} . Här är $\mathbf{x} = \{x_i\}_{i=1}^n$ och $\mathbf{y} = \{y_i\}_{i=1}^n$ vektorer med samma längd n och (x_i, y_i) representerar punkter på kurvan $y = y(x)$.

Man kan bilda $\{x_i\}_{i=1}^n$ enligt

$$x_0 = a, \quad x_n = b, \quad a < x_i < b, \quad i = 1, 2, \dots, n-1,$$

och sätt

$$y_i = y(x_i), \quad i = 0, 1, 2, \dots, n-1, n.$$

15.0.6 Exempel

Beräkna integralen

$$J = \int_0^1 e^{-x^2} dx$$

med trapetsformeln.

Först, skapa vektorn $\mathbf{x} = \{x_i\}_{i=1}^n$ med längder 5 ($\mathbf{x5} = \{x_i\}_{i=1}^5$) och 10 ($\mathbf{x10} = \{x_i\}_{i=1}^{10}$):

```
x5 = linspace(0,1,5);  
x10 = linspace(0,1,10);
```

Sedan, skapa vektorn \mathbf{y} som en funktion av \mathbf{x} :

```
y5 = exp(-x5.*x5);  
y10 = exp(-x10.*x10);
```

och beräkna integralen:

```
integral5 = trapz(x5,y5), integral10 = trapz(x10,y10)
```

Kommandon returnerar

```
integral5 =  
    0.74298409780038  
integral10 =  
    0.74606686791267
```

16 Numerisk lösning av ordinära differentialekvationer i MATLAB

MATLAB använder *Runge-Kutta-Fehlberg-metoder* för att lösa begynnelsevärdesproblem för ordinära differentialekvationer (ODE).

Lösningen beräknas i ett ändligt antal punkter. Färre punkter används där lösningen inte varierar så kraftigt, och fler punkter där den gör det.

16.1 Begynnelsevärdesproblem för ODE

`[t,X] = ode23(str, t0, tt, x0)` beräknar lösningen till den ODE eller det system av ODE som ges av strängen **str**. Lösningen ges dels i vektorn **t** som ger *t*-värdena, och dels i matrisen **X**, vars kolonner är lösningar till respektive ekvation i dessa punkter. För ett skalärart problem ges lösningen i vektorn **x**.

Lösningen beräknas från *t0* till *tt* och som begynnelsevärde används $\mathbf{x0} = \mathbf{x}(t0)$. Systemet bestäms av funktionen i den M-fil som anges i **str**. Denna funktion ska ha två inparametrar, skalären (tal) *a* och vektorn **x**, och ska returnera vektorn **x'** (derivatan). För en skalär ODE, är *x* och *x'* skalärer (tal).

Approximationen har relativt fel som är högst 10^{-3} .

16.1.1 Exempel

Lös begynnelsevärdesproblemet

$$x' = -x^2, \quad x(0) = 1.$$

Lösning. Skapa funktionen **xprim1**, lagrad på M-filen **xprim1.m**:

```
function xprim1 = xprim1(t,x)
xprim1 = -x.*x;
```

Sedan anropar vi MATLAB:s ODE-lösare

```
[t,x] = ode23('xprim1',0,1,1);
Slutligen plottar vi lösningen
plot(t,x,'-o');
xlabel('time t0 = 0, tt = 1'); ylabel('x values x(0) = 1');
```

16.1.2 Exempel

Lös begynnelsevärdesproblemet

$$x' = x^2, \quad x(0) = 1.$$

Lösning. Skapa funktionen **xprim2**, lagrad på M-filen **xprim2.m**

```
function xprim2 = xprim2(t,x)
xprim1 = x.*x;
```

Sedan anropar vi MATLAB:s ODE-lösare

```
[t,x] = ode23('xprim2',0,0.95,1);
och plottar lösningen
plot(t,x,'-o');
xlabel('time t0 = 0, tt = 0.95'); ylabel('x values x(0) = 1');
```

Notera att MATLAB beräknar punkter tätare i området med hög derivata.

16.1.3 Exempel

Lös begynnelsevärdesproblemet för systemet av ODE

$$\begin{aligned}x_1' &= x_1 - 0.1x_1x_2 + 0.01t \\x_2' &= -x_2 + 0.02x_1x_2 + 0.04t \\x_1(0) &= 30 \\x_2(0) &= 20\end{aligned}$$

Lösning. Skapa funktionen `xprim3`, lagrad på M-filen `xprim3.m`. Notera att den är en vektor-funktion.

```
function xprim = xprim3(t,x)
xprim(1) = x(1) - 0.1*x(1)*x(2) + 0.01*t;
xprim(2) = -x(2) - 0.02*x(1)*x(2) + 0.04*t;
```

Sedan anropar vi MATLAB:s ODE-lösare

```
[t,x] = ode23('xprim3',0,20,[30;20]);
```

och plottar lösningen

```
plot(t,x);
```

```
xlabel('time t0 = 0, tt = 20'); ylabel('x values x1(0) = 30, x2(0) = 20');
```

Man kan rita upp x_1 som en funktion av x_2

```
plot(x(:,2), x(:,1));
```