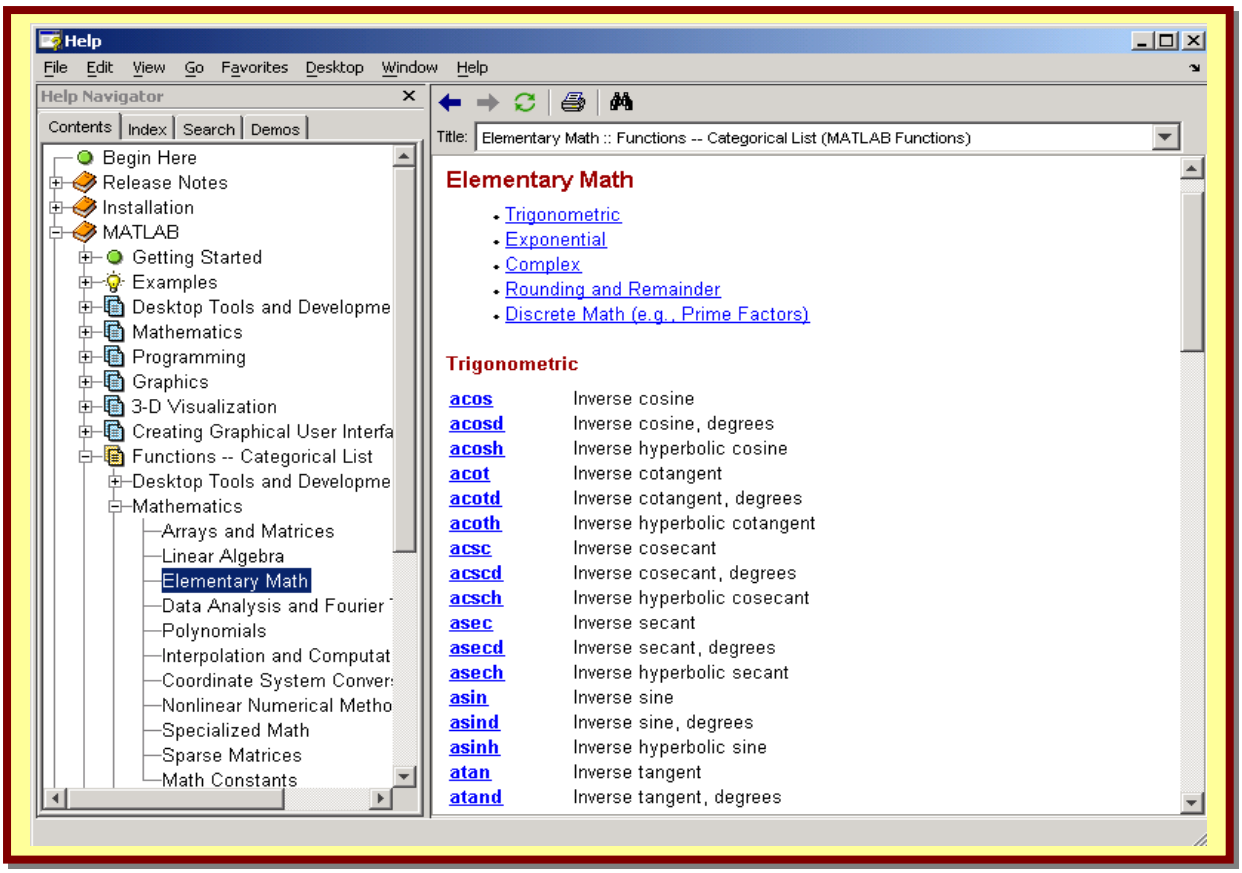
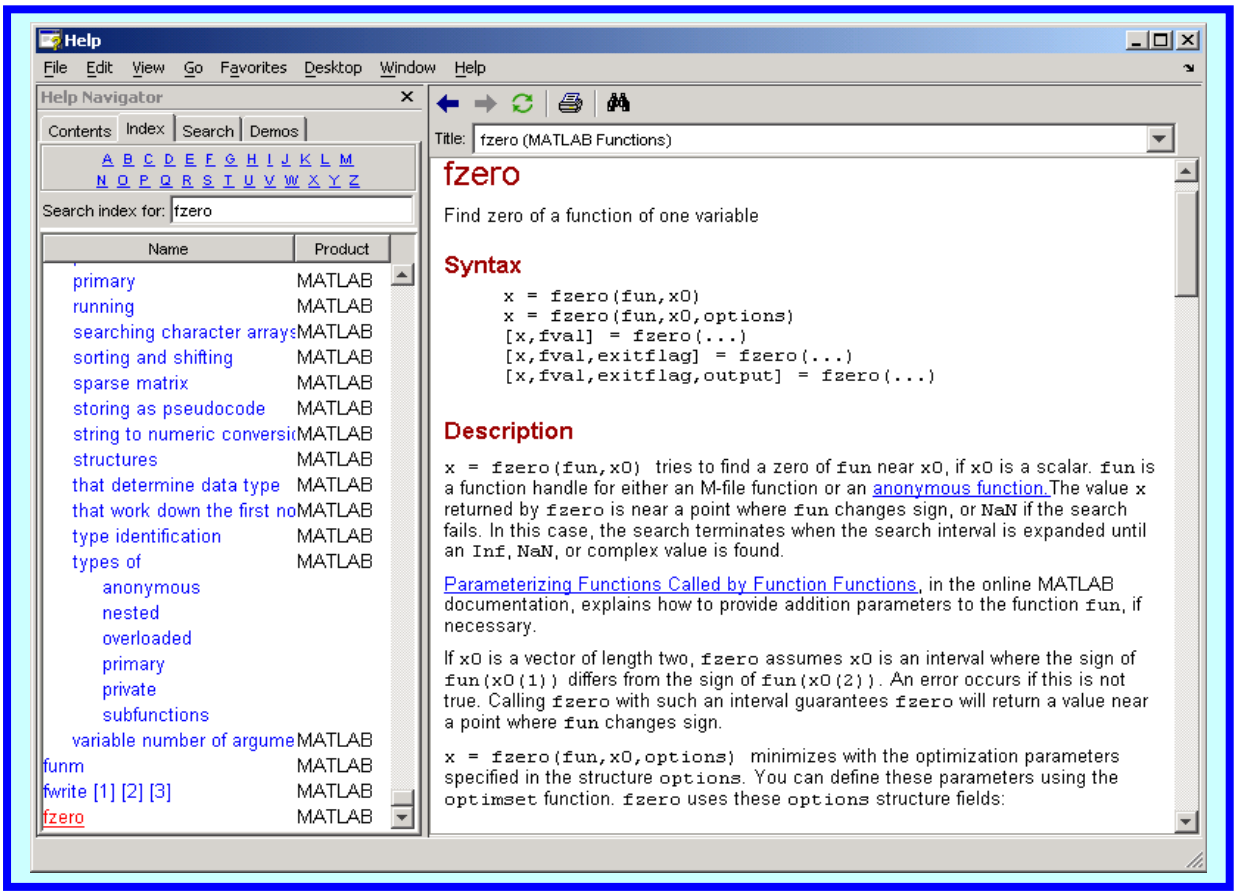


Du kan söka hjälp efter innehåll eller efter namn



Skalärer

x = 2
y = 1.234 *(observera decimalpunkt.)*
pi , inf

Vektorer

v = [1 , 2 , 3 , 4] radvektor
u = v' = [1 ; 2 ; 3 ; 4] kolonnvektor

Matriser

A = [1 , 2 , 3 ; 4 , 5 , 6 ; 7 , 8 , 9]

blir matrisen : $\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}$

B(: , : , 1) = A;
B(: , : , 2) = A;
B(: , : , 3) = A;

blir tillsammans det tredimensionella fältet :

$$\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix} \begin{pmatrix} 2 & 3 \\ 5 & 6 \\ 8 & 9 \end{pmatrix} \begin{pmatrix} 2 & 3 \\ 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}$$

A(1 : 3 , 2 : 3) blir matrisen :

$$\begin{pmatrix} 2 & 3 \\ 5 & 6 \\ 8 & 9 \end{pmatrix}$$

Funktioner för generering av fält:

t e x: ones , eye , diag , zeros
diag(ones(1 , 7)) blir enhetsmatrisen av ordning 7
eye(7) blir enhetsmatrisen av ordning 7

Strängar:

text = 'Jan'
mer_text = ' Södersten' *(observera blanktecken före S.)*
namn = [text , mer_text] blir 'Jan Södersten'
namn(1:6) blir 'Jan Sö'

Operatorer och operander skall skrivas i överensstämmelse med reglerna för matris/vektoroperationer.

Aritmetiska operatorer: + , - , * , / , \ , ^

Om t ex A är en kvadratisk matris betyder A\B
$A^{-1} \cdot B$

Elementvisa operatorer: .* och ./ och .^

Med X = [1 , 2 , 3] och Y = [2 , 3 , 4] blir
*X.*Y = [2 , 6 , 12]*

Relationsoperatorer: < <= > >= = = /=

Logiska operatorer: & | ~ *(and , or , not)*

Konstanter: 1 pi inf ans eps i

Standardfunktioner: sin(A), exp(A), log(x), plot(x,y)

Uttryck: A + B\C - sin(A*B)

Tilldelning: svar = uttryck
där högerledet först beräknas och sedan tilldelas variabeln i vänsterledet.

---

who **clear**

load **save** för .mat filer

I Desktop finns en ruta Workspace: där Du ser Dina variabler.
Via huvudmenyn: "File" kan Du lagra och hämta variabler.
Via Workspacemenyn: "Delete" kan Du radera variabler.

Kommentarer:

```
% Text
```

Spara på fil vad Du gjort i kommando-fönstret:

```
Diary
Diary filnamn
Diary on , Diary off
```

Du skriver ut innehållet i kommandofönstret genom att klicka på kommandofönstret och välj File/Print i huvudmenyn.

In- och utmatning:

```
x = input( 'Text' )
Disp( x )
```

Läs i Help

```
format long
format short
format compact
```

.m FILER

Finns i två varianter:

Script-filer

(Kommandofiler)

och:

Funktions - filer

Se i Help:

Contents / MATLAB / Getting Started /
Programming / Scripts and Functions

Hittar MATLAB Dina .m filer? **path**

addpath( ) t ex **addpath(A:)**

Set Path via huvudmenyn: File / Set Path

Funktionsfiler

Är

.m -filer

och inleds med

function

till skillnad från script-filer.

ex. :

Funktionsfilen: C:\funk.m

```
function y = funk(x)
    global a
    y = x.^2 - 5*x +3*a;
```

funktionens argument:
x

global a gör det möjligt att ta med ett värde på a till funktionen om a inte skall vara argument.

Där filen anropas skall a också vara global-deklarerad.

Anropa funktionen med filens namn, utan ".m".

Ex.:

y = funk(3.65)

x = fzero(@funk , x0) för beräkning av nollställe.

I funktionsanropet fzero(@funk , x0) är @ ett "funktionshandtag", Function Handle.

Man kan även skriva: 'funk' istället för @funk
(Så skriver man i MATLAB 5 och tidigare versioner.)

Fler ex.:

function [a , b] = jan(x , y , z)

är en funktion med tre argument, och som ger a och b som svar.

Om

function y = emil(funk , x0)

har en funktion funk som argument:

emil.m

```
function y = emil( funk , x0 )
  .
  .
  .
y = feval( funk , x0 )
```

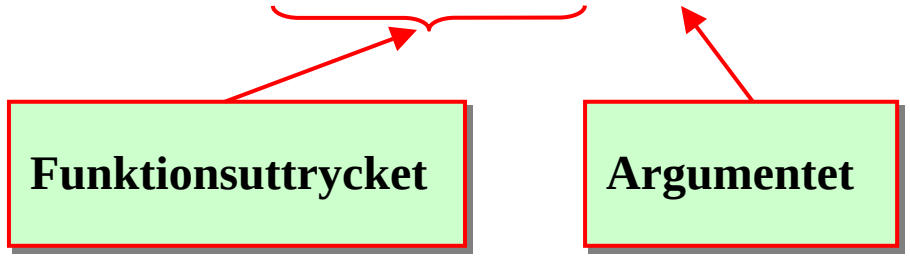
Här måste framgå att funk är en funktion, som skall anropas.
Man kan inte skriva y = funk(x0), funk tolkas då som ett fält.

feval(funk , x0) evaluerar (beräknar) funk för argumentet x0.
Och så här anropar man emil för g(z) :
emil(@g , z)

Inline Function

Om Du vill skapa en funktion funk utan att lägga den i en egen fil:

```
funk = inline( 'x.^2 - 5*x + 3' , 'x' )
```



I filen används då funk utan @ .

I funktionsfiler får Du skriva "underfunktioner", men inte i scriptfiler.

En underfunktion skrivs som den skulle skrivas i en egen fil och läggs sist i funktionsfilen.

En underfunktion och en inline kan endast anropas i den fil där den står.

Några exempel på repetitioner:

- ```
  for i = [1, 2, 3]
        satsgrupp
  end
  ```

```
for variabel = repetitionslista
    satsgrupp
end
```

```
for i = 1 : -0.5 : -4.4
      satsgrupp
end
```

```
for i = 1 : 10
      satsgrupp
      for j = 1 : 10
          satsgrupp
      end
end
```

Indentera för läsbarhetens skull!!

- ```
  i = 1;
  while i <= 10
    satsgrupp
    i = i+1
  end
  ```

```
while villkor
    satsgrupp
end
```

Observera att villkoret måste kunna beräknas när repetitionen börjar. Satsgruppen utförs så länge villkoret är sant. Risk för "idiotloop".

Arbeta helst med vektorer om det går:

Följande är ekvivalent:

```
v = [ ];
for i = 1 : 10
   v = [v , i];
end;
```

```
v = [1 : 10];
```

Exempel på selektion:

```
if   villkor
     satsgrupp
end
```

```
if   villkor
     satsgrupp
else
     satsgrupp
end
```

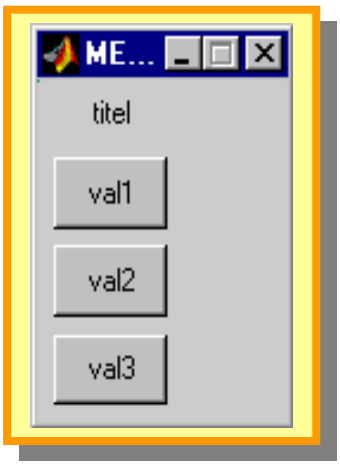
```
if   villkor
     satsgrupp
elseif villkor
          satsgrupp
       else
          satsgrupp
end
```

Om elseif skrivs som ett ord, räcker det med ett end. ***Annars ett end per if.***

p = menu('titel', 'val1', 'val2','val3')

ger en "inmatningsruta"

Klickar Du på:
- **översta knappen (första) blir p = 1**
- **mellanknappen (andra) blir p = 2**
- **nedre knappen (tredje) blir p = 3**



switch

Ett alternativ till if - satsen när man har ett uppräkneligt antal fall är switch - satsen:

I ex. får x värdet 10, om val = 1
värdet 100, om val = 3, 5 eller 7
värdet 1000, om val =10 tom 15
värdet 0 annars.

```
switch val

case 1
   x=10

case {3,5,7}
   x=100

case num2cell(10:15)
   x=1000

otherwise
   x=0

end
```

Testa värdet på val

Flera fall skrivs i en "cellvektor"

vektorn [10 : 15] omvandlad till cellvektorn {10,11,12,13,14,15}

Några kommandon för att bryta exekverings ordningen:

- **Man bör skriva kommandon och satser i en rak följd och undvika att bryta exekveringsordningen.**
- **Skriv gärna m-filer (script- eller funktionsfiler), som sedan anropas i en naturlig följd.**
- **Lägg gärna in kommentarer för läsbarhetens skull. %**

break **avbryter en repetition.**

pause **gör ett uppehåll.**

return **lämnar en .m-fil, och exekvering fortsätter där filen anropades.**

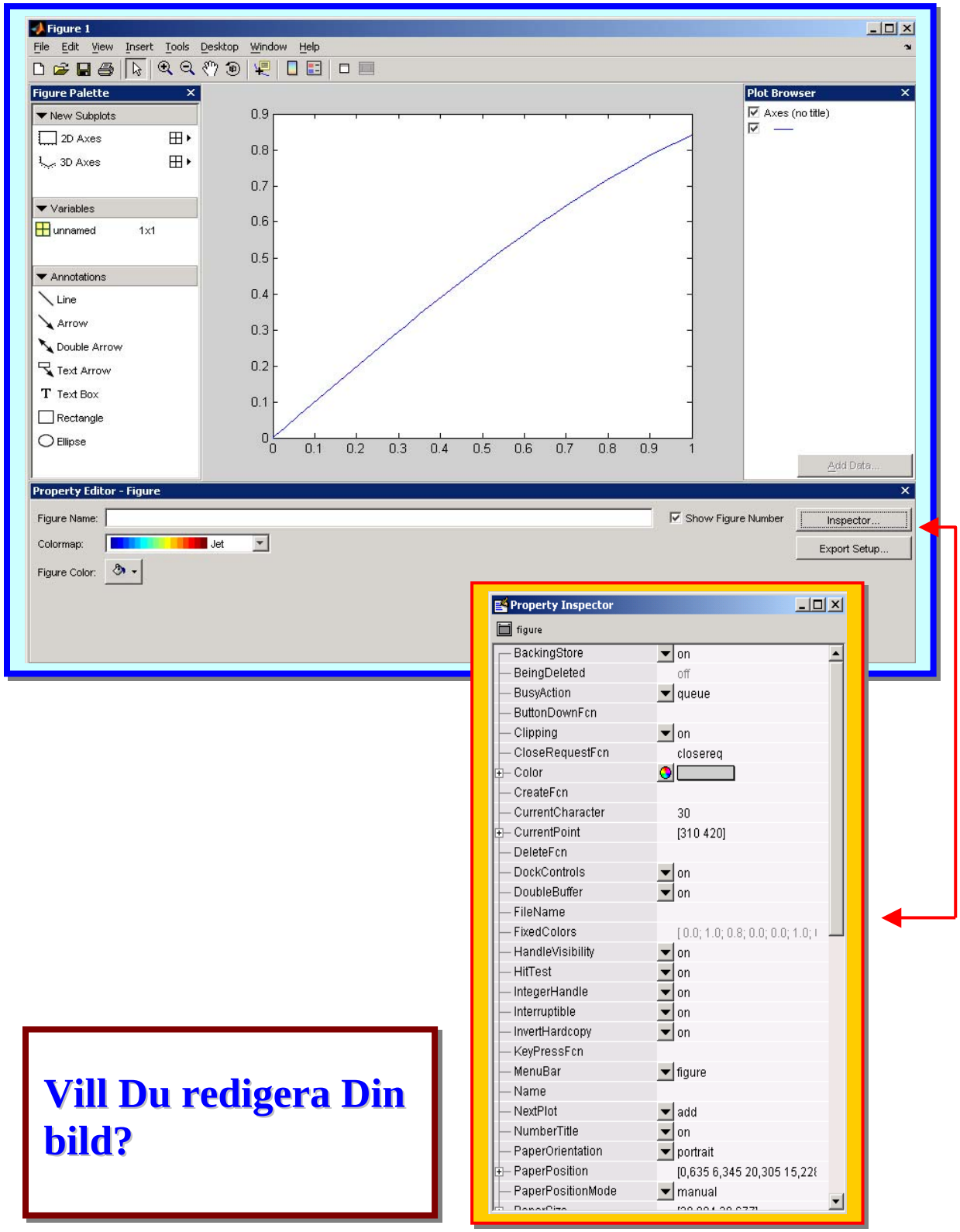
<ctrl><c> **exekveringsavbrott om Du trycker på dessa tangenter samtidigt.**

2 - dimensionell GRAFIK

Rita linjer och punkter:	**plot( )**
För fler grafer i samma bild:	**hold , hold on**
För grafen i ny bild:	**hold off**
Rutnät i bilden:	**grid on/off**
Rubrik på grafen:	**title('Rubriken')**
Markering av axlar:	**xlabel('x-axel')** **ylabel('y-axel')**
Text till bilden:	**text(x, y, 'Text')** **gtext('Text')**
Hämta koordinater ur bilden:	**p = ginput( )**
Zoomning:	**zoom on/out**
Dela upp grafikfönstret i småbilder:	**subplot(, ,)**

Se även menyn ovanför bilden.

```
>> fplot('sin(x)',[0,1])
>> plottools
>> shg
```



Vill Du redigera Din bild?

plot3(x, y, z)	**linjer i 3-dim**
[X , Y] = meshgrid(x, y)	**gitter för konstr. av nivåkurvor och ytor.**
contour(X, Y, Z)	**nivåkurvor**
mesh(X, Y, Z)	**nätyta**
meshc(X, Y, Z)	**nätyta med nivåkurvor**
surf(X, Y, Z)	**yta**
Waterfall(X, Y, Z)	**yta med punkter bundna i en riktning**
hidden on/off	**döljer/visar skymda linjer**
rotate3d on/off	**möjlighet att rotera bilden med musen**

Se bildens meny: View / Camera Toolbar
för att variera sättet att betrakta bilden.

print -filformat filnamn	**lagra bilden**

Linjära ekvationssystem

Betrakta problemet att lösa det linjära ekvationssystemet:

$$Ax = y$$

- **Ett kvadratiskt system, y: vektor eller matris.**

  det(A) **inv(A)** **eye(5)** **rank(A)**

  A(2:4,5:8) plockar en submatris ur A.

  Gausseliminering: **Ax = y** $\Rightarrow$ **Rx = z**
  i MATLAB: **x = A\y**
  se även lu(A) och rref(A)

- **Ett överbestämt system: A rektangulär matris.**

  x =A\y i MATLAB ger det x som minimerar $\|Ax-y\|_2$ **,**
  "minstakvadratmetoden"

norm(x) **norm(A)** **norm(x, 2)**

tic **toc** $\Leftrightarrow$ Tidtagning för uppskattning av beräkningsvolymen

Polynomhantering

Ett polynom av grad ≤ n :

$$p(x)=a_1x^n+a_2x^{n-1}+...+a_nx+a_{n+1}$$

kan definieras med en vektor:

p = $\left[a_1,a_2,...,a_{n+1}\right]$ **av dimensionen (n+1).**

MATLAB's **polyval(p, x)** **beräknar p(x):**
en skalär om x är en skalär
en vektor om x är en vektor.

MATLAB's **roots(p)** **beräknar nollställena till polynomet.**

MATLAB's **a = polyfit(x, y, n)** **beräknar det polynom p(x) av grad ≤ n som bäst approximerar punktmängden** $\left\{x_i,y_i\right\}_{i=1}^{m}$ **i minsta kvadratmetodens mening, där a, x, y är vektorer, med:**

$$p(x)=a_1x^n+a_2x^{n-1}+...+a_nx+a_{n+1}$$

Numerisk integrering:

Sök det numeriska värdet av $\int_a^b f(x)dx$

ex.:

trapz(x ,y)

quad(@fkn, a, b) **quadl(@fkn, a, b)**

Numerisk lösning av ordinära differentialekvationer:

Beräkna $y(b)$ **om**

$$\begin{cases} y' = f(x,y) \\ y(a) = c \end{cases}$$

ex.:

ode45(@der, [a b], c) ***(Blanktecken mellan a och b)***

Läs om ode, quad och trapz i Help och i användarhandledningen.

Ett exempel på lösning av ett linjärt ekvationssystem:

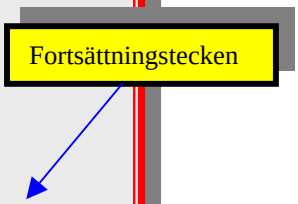
Lös ekvationssystemet: Ax = B , där

$$A=\begin{pmatrix}1 & 2 & 3\\ 3 & 0 & 2\\ 0 & 5 & -1\end{pmatrix}$$ **och** $$B=\begin{pmatrix}1\\ 2\\ 0\end{pmatrix}$$ **.**

Vi använder MATLAB operatorn \ , och undviker att beräkna matrisens invers.

Scriptfilen: **A:\Foerelaesning1\linalgex.m**

```
%linjär algebra
%1x+2y+3z=1
%3x+   2z=2
%   5y- z=0
clear all
A=[1 2 3;3 0 2;0 5 -1]
B=[1;2;0]
lsg=A\B;
x=lsg(1)
y=lsg(2)
z=lsg(3)
disp('Med dessa insatta på vänstersidan ...
      får vi högerledet=')
disp([1*x+2*y+3*z;3*x+2*z;5*y-z])
```



Ett exempel på integralberäkningar

Beräkna numeriskt:

$$\int_a^b (2\sin(x) - \frac{\cos(x)}{1+x^2})\, dx$$

för olika val av integrationsgränserna.

Vi använder MATLAB's: quadl och trapz

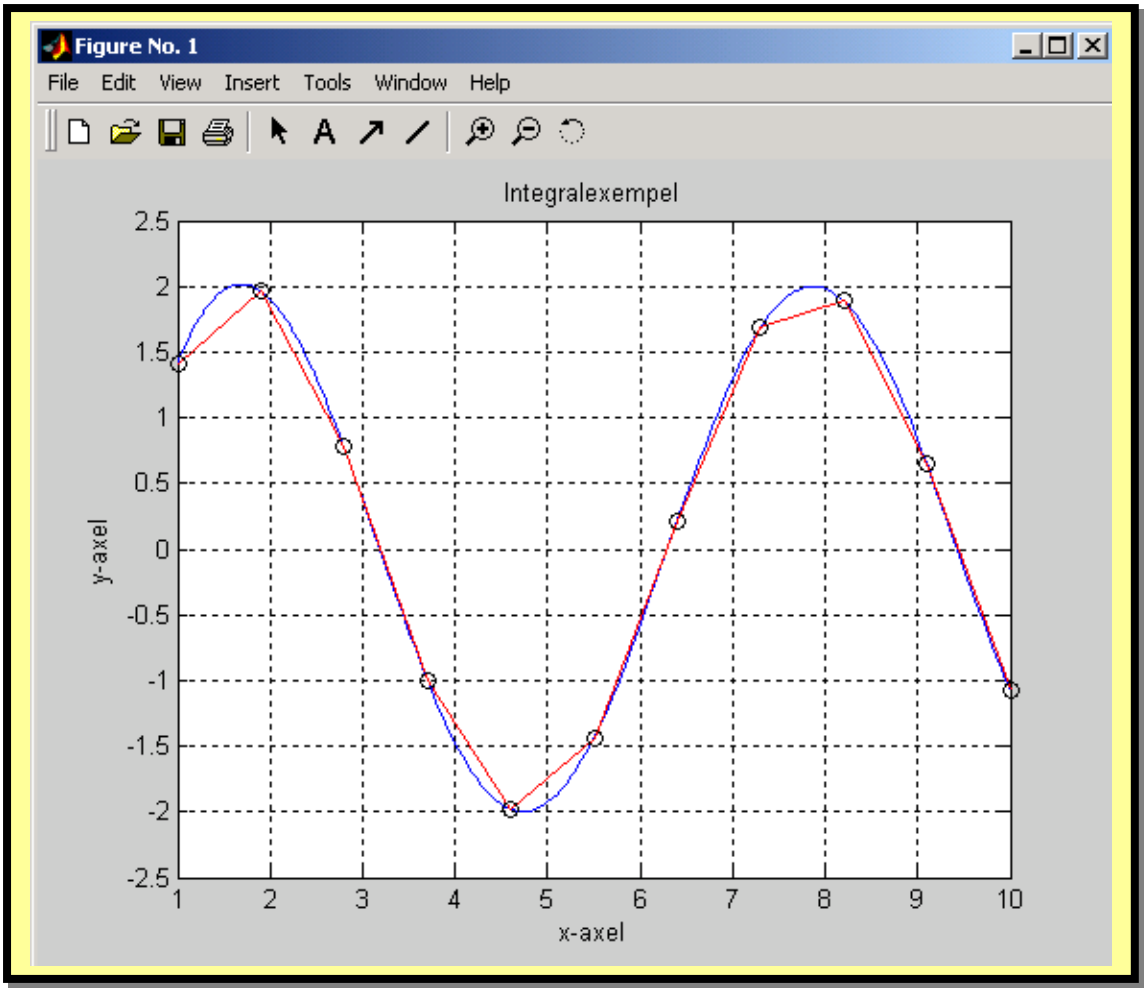
Scriptfilen: A:\Foerelaesning1\intex.m

```
% integralexempel
clear all
clf
disp( 'Integralen av funktionen: 2sin(x)-cos(x)/(1+x^2)' );
a=input( 'Ange undre gräns: ' );
b=input( 'Ange övre gräns: ' );
int=quadl( @fkn , a , b );
disp( [ 'Integralvärdet blir: ' , num2str(int) ] );
x=linspace( a , b , 11 );
y=fkn(x);
int2=trapz( x , y )
disp( [ 'Med trapets får vi: ' , num2str(int2) ] );
fplot( @fkn , [ a , b ] );
hold on
plot( x , y , 'r' );
plot( x , y , 'ko' );
grid;
xlabel( 'x-axel' );
ylabel( 'y-axel' );
title( 'Integralexempel' );
```

quadl har som första argument namnet på funktionsfilen fkn.m . *(Förväxla inte filnamn!!!)*

Funktionsfilen: A:\Foerelaesning1\fkn.m

```
function y= fkn(x)
%Här beräknas funktionsvärdena
y=2*sin(x)-cos(x)./(1+x.^2);
```



>> Integralen av funktionen: 2sin(x)-cos(x)/(1+x^2)

Ange undre gräns: 1

Ange övre gräns: 10

Integralvärdet blir: 2.8674

int2 =

2.6319

Med trapets får vi: 2.6319
>>

Ett exempel på kurvanpassning och tvådimensionell grafik:

Anpassa polynom av graderna 1, 3 och 5 till 6 givna punkter. Rita in punkter och polynom i samma grafikfönster.

Vi använder MATLAB's:

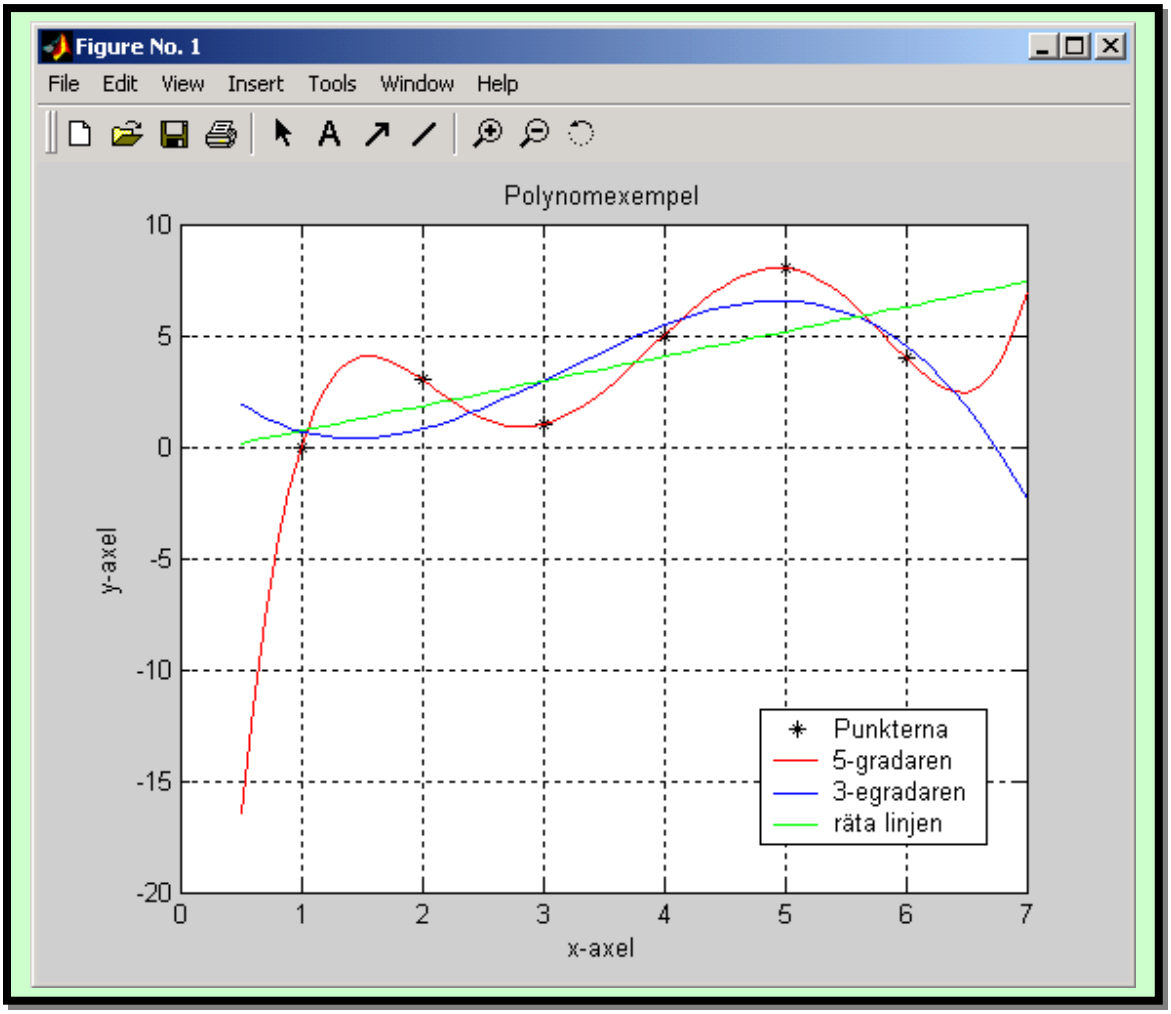
polyfit	**för att bestämma polynomen**
polyval	**för att beräkna polynomens värden**
plot	**för att rita polynom och punkter**
hold on	**för att få allt i samma bild**
legend	**för att få en förklarande text**
title, xlabel, ylabel, grid	

Scriptfilen: **A:\Foerelaesning1\polyex.m**

```
%Exempel på polynomhantering:
clear all
clf
xpunkter=[1,2,3,4,5,6];
ypunkter=[0,3,1,5,8,4];

koeff6=polyfit(xpunkter,ypunkter,5);
koeff4=polyfit(xpunkter,ypunkter,3);
koeff2=polyfit(xpunkter,ypunkter,1);

xfin=0.5:0.1:7;
y6=polyval(koeff6,xfin);
y4=polyval(koeff4,xfin);
y2=polyval(koeff2,xfin);

plot(xpunkter,ypunkter,'k*');
hold on;
plot(xfin,y6,'r');
plot(xfin,y4);
plot(xfin,y2,'g');
grid
xlabel('x-axel');
ylabel('y-axel');
title('Polynomexempel');
legend('Punkterna','5-gradaren','3-egradaren','räta',... '
linjen');
roetter_till_5egradaren=roots(koeff6)
```

Fin indelning för snygga kurvor.

***legend*, där rubrikerna skall komma i den ordning kurvorna ritats i filen.**




```
>>
roetter_till_5egradaren =

   6.5369 + 0.4520i
   6.5369 - 0.4520i
   2.8315 + 0.4691i
   2.8315 - 0.4691i
   1.0000
>>
```

Ett exempel på 3-dimensionell grafik:

Dra grafer över funktionen:

$$\sin(x)+\cos(y)\ ,\ 0\le x,y\le 2\pi$$

Vi använder 4 MATLAB funktioner:

mesh, contour, waterfall, surf

väljer en av dem med hjälp av

menu

delar upp grafikfönstret i fyra delar med

subplot

Scriptfilen: **A:\Foerelaesning1\grafex.m**

```
%3D-grafexempel
clear all
disp('Graf över sin(x)+cos(y)på int 0<=x,y<=2Pi');
val=menu('Välj typ av graf',…
         'mesh','contour','waterfall','surf');
xled=0:0.5:2*pi;
yled=xled;
[X,Y]=meshgrid(xled,yled);
A=sin(X)+cos(Y);
if val==1
   subplot(2,2,1);
   mesh(X,Y,A);
elseif val==2
   subplot(2,2,2);
   contour(X,Y,A);
elseif val==3
   subplot(2,2,3);
   waterfall(X,Y,A);
else
   subplot(2,2,4);
   surf(X,Y,A);
   colorbar; shading interp
end
shg
hold on
```

⇔

Här sker ett val per anrop av filen. Eftersom filen avslutas med hold on, fylls samma grafikfönster på vid ett nytt anrop.

shg visar grafik-fönstret på bild-skärmen.

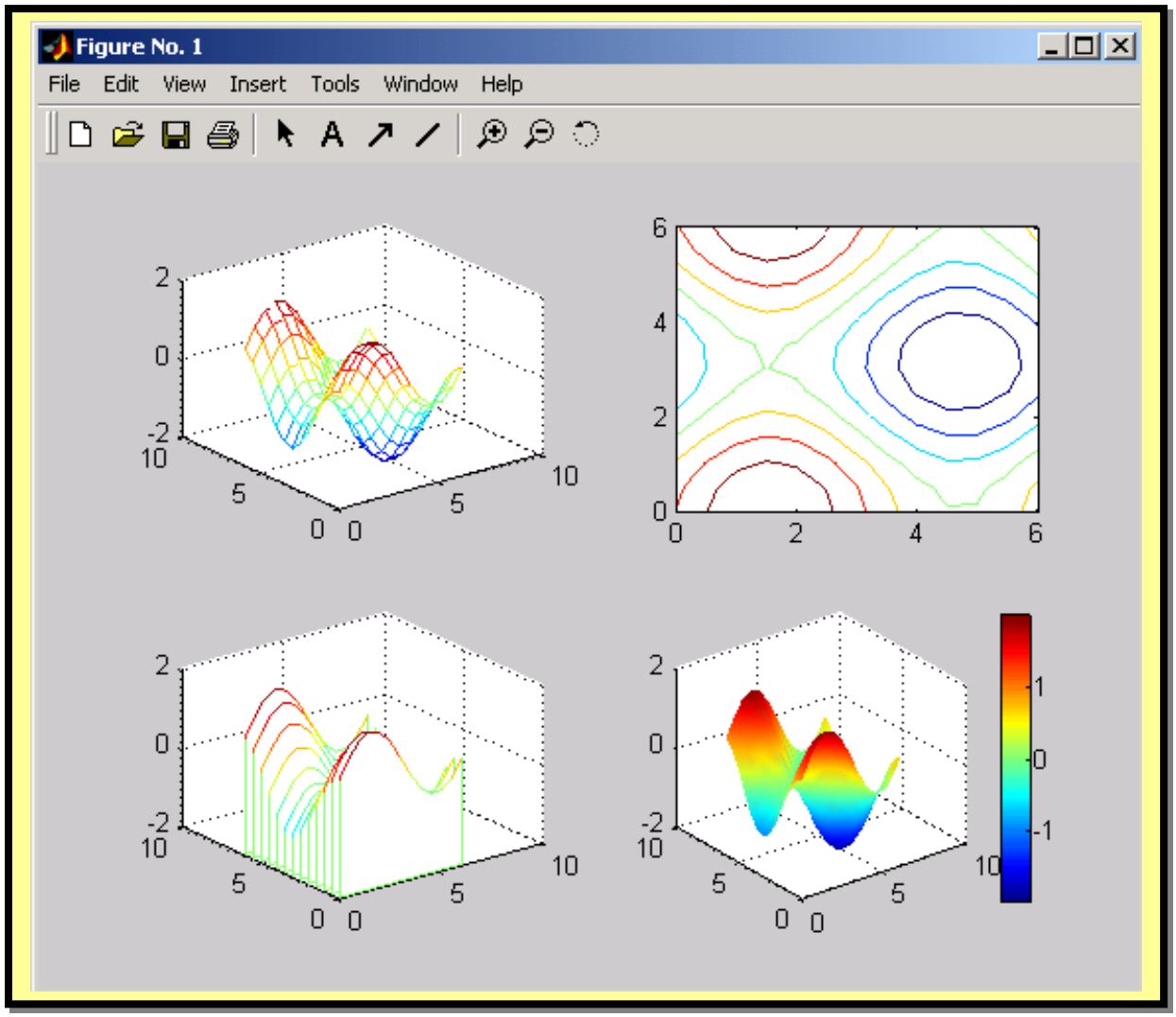


Figure No. 1
File Edit View Insert Tools Window Help
2
0
-2
10
5
0
0
5
10
6
4
2
0
0
2
4
6
2
0
-2
10
5
0
0
5
10
2
0
-2
10
5
0
0
5
10
1
0
-1

Ett exempel på lösning av en ordinär differentialekvation:

$$y' = \frac{\sin(5x)}{1+x} \quad , \quad y(0) = 1 \quad : y(3) = ?$$

Vi använder MATLAB'S: **ode45**

Scriptfilen: **A:\Foerelaesning1\diffex.m**

```
%Exempel på ordinära diffekvationer
clear all
clf
[x,y]=ode45(@der,[0,3],1);
plot(x,y);
hold on;
grid;
xlabel('x-axel');
ylabel('y-axel');
title('ODE-ex');
```

ode45 har som första argument namnet på funktionsfilen der.m . ***(Förväxla inte filnamn!!)***

Funktionsfilen: **A:\Foerelaesning1\der.m**

```
function yprim = der(x,y)
yprim = sin(5*x)./(1+x);
```

Talrepresentation

I basen 10: $(\quad)_{10}$

$(\quad)_{10} = (\pm d_n \ldots d_0 . d_{-1} d_{-2} \ldots)_{10}$ betyder

$$\pm d_n 10^n + \ldots d_1 \cdot 10^1 + d_0 \cdot 10^0 + d_{-1} 10^{-1} + \ldots$$

decimalsystemet.

Exempel:

$$123.27 = \quad 1 \cdot 10^2 + 2 \cdot 10^1 + 3 \cdot 10^0 + 2 \cdot 10^{-1} + 7 \cdot 10^{-2}$$

I basen 2: $(\quad)_2$

$(\quad)_2 = (\pm d_n \ldots d_0 . d_{-1} d_{-2} \ldots)_2$ betyder

$$\pm d_n 2^n + \ldots d_1 \cdot 2^1 + d_0 \cdot 2^0 + d_{-1} 2^{-1} + \ldots$$

binära systemet.

Exempel:

$$1011.01 = \quad 1 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0 + 0 \cdot 2^{-1} + 1 \cdot 2^{-2}$$

Exempel på lagring av tal

Lagring av heltal binärt med en ordlängd på 32 bitar:

±	Talets binära representation																													
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	1	1	0

är representationen av talet:

$0 \cdot 2^0 + 1 \cdot 2^1 + 1 \cdot 2^2 + 0 \cdot 2^3 + 0 \cdot 2^4 + 1 \cdot 2^5 + 1 \cdot 2^6$ = 102 i decimalsystemet

Icke-heltal representeras binärt som "flytande tal" på formen:

$$a \cdot 2^e$$

där a kallas mantissa, och skall ha 1 som heltalsdel
och e kallas exponent.

Lagring av talet i "enkel precision" i ett ord med 32 bitar.

±	ett skiftat e								mantissans bråkdel																					
0	0	0	1	0	1	1	1	1	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0

är representation av talet:

$1.11001100110011001100110 * 2^{-80}$, som i decimalsystemet =

$(1+1/2+1/4+0/8+0/16+1/32+1/64+0/128+...) \cdot 2^{-80} \approx$
$1.4863 \cdot 10^{-24}$

MATLAB räknar standardmässigt i dubbel precision.