```
% Föreläsning 1 31/1

% Kommentarer efter %-tecken

clear % Vi nollställer allting

1/2+1/3 % Matlab räknar numeriskt. Observera punkten som decimaltecken.
sym(1/2+1/3) % Nu blev det symboliskt
pi % Vissa konstanter finns färdiga
format long % Om vi vill se fler värdesiffror
pi
format % Återställer
sin(pi) % Vissa funktioner finns också färdiga
sin(sym(pi)) % Nu blev det exakt

x=3 % Vi ger variabeln x värdet 3
x % Finns nu i minnet
y=x+2
x+2 % Om vi inte har något vänsterled hamnar svaret i variabeln ans
y=ans+1 % y får ett nytt värde
x=x+1 % Detta ser kanske lite kontigt ut, alltså ingen ekvation
z=y-7; % Med ett ;-tecken efter så skrivs inte resultatet ut
z % Men z finns såklart i minnet

U=[ 1 2 3 ] % Vi skapar en vektor (radmatris)
A=[ 1 2 3 ; 4 5 6 ; 7 8 9 ] % Detta ger en 3*3-matris. Vi byter rad med ;
A=[ 1 2 3
    4 5 6
    7 8 9 ] % Eller helt enkelt genom att byta rad
V=[ 4 ; 5 ; 6 ] % Detta ger en kolonnvektor (kolonnmatris)

B=A*A % Matrismultiplikation
U*V % Blir ett tal (1*1-matris)
V*U % Blir en 3*3-matris
U*U % Funkar inte
A*V % Kolonnvektor
U*A % (Rad)vektor
B=A+A % Matrisaddition
U+U % Funkar
U+V % Borde inte funka, så Matlab gör något annat än vanlig addition.
B=A.*A % Med en punkt framför operatorn verkar denna elementvis
U.*U % Nu funkar detta
U.*V % Undrar vad Matlab gör här?
A^3 % Samma som A*A*A
A.^3 % Samma som A.*A.*A
U.+U % Detta finns inte med, eftersom det är samma som U+U
sin(A) % Jodå, vi kan ta sinus av en hel matris (elementvis)

At=A' % ' transponerar en matris, dvs. kastar om rader och kolonner
U' % Radvektorn U ger en kolonnvektor

A(2,3) % Så här plockar vi ut elementet på rad 2 kolonn 3
U(2) % Andra elementet i U
A(4) % Detta funkar, men observera att den räknar kolonnvis
A(3,2)=-A(3,2) % Vi kan också byta värde på ett enskilt element
```

```
A(1,:) % Här plockar vi ut hela första raden
A(:,2) % Och andra kolonnen
A(:,3)=V % Byter ut tredje kolonnen mot V
A(1:2,2:3) % Ger delmatrisen bestående av rad 1 till 2 kolonn 2 till 3
A([ 1 3 ],:)= [ 10 11 12 ; 13 14 15 ] % Byter ut första och tredje raden

X=1:9 % Ger en vektor med element från 1 till 9
Y=1:2:9 % Detta ger en med steglängd 2
Z=40:-3:10 % Eller varför inte så här
length(Z) % Ger längden av, dvs. antal element i, Z
size(Z) % Eller som storlek (typ) tolkad som matris
size(A(1:3,2:3)) % Storleken av 3*2-matrisen vi plockar ut ur A
Z(3:2:11) % Här plockar vi ut element 3 till 11 i steg om 2

A=ones(2,2) % Ger en 2*2-matris med bara 1:or
B=zeros(2,2) % Med bara 0:or
C=eye(2,2) % Enhetsmatrisen, som vi skall återkomma till
D=rand(2,2) % Ger slumptal (likformigt fördelade mellan 0 och 1)
[ A B ] % Vi kan sätta ihop hela matriser
[ A ; B ]
[ A B ; C D ]

% Vi tittar nu på några linjära ekvationssystem.
% Uppgift 1.3 i boken kan vi lösa så här
A=[ 1 -2 1 ; 2 -6 6 ; -3 5 1 ]; % Matar in systemmatrisen
Y=[ 1 ; 2 ; 3 ]; % Och högerledet
A\Y % Detta ger lösningen
linsolve(A,Y) % Detta går också bra
% Vi provar nu med uppgift 1.4
A=[ 1 -2 1 ; 2 -6 6 ; -3 5 -1 ]; % Systemmatrisen
Y=[ 1 ; 2 ; 3 ]; % Högerledet
A\Y % Detta saknade lösning. Skall förklara texten vid senare tillfälle.
% Och till sist 1.11
A=[ 1 -2 1 ; 2 -6 6 ; -3 5 -1 ]; % Systemmatrisen
Y=[ 1 ; 4 ; -2 ]; % Högerledet
A\Y % Här skulle vi ju ha parameterlösning. Förklarar återigen senare.
% Vi får iallafall en lösning.

%Vi tittar nu på några system som inte är kvadratiska.
% I uppgift 1.14 har vi fler obekanta än ekvationer,
% och förväntar oss parameterlösning
A=[ 2 3 4 ; 4 -3 2 ] % Systemmatrisen
Y=[ 5 1 ]' % Högerledet
A\Y % Matlab ger en lösning, men återigen bara en.
% I uppgift 1.9 har vi fler ekvationer än obekanta,
% och förväntar oss att lösning saknas.
A=[ 1 1 ; 1 -2 ; 3 4 ]
Y=[ -4 2 1]'
% Matlab ger en lösning, trots att någon sådan inte finns!
% Vi tittar nästa gång på vad Matlab gör.

% Några välkända operationer på vektorer
u=[1,2,3] % Detta är samma som radmatrisen [ 1 2 3 ]
v=[4,5,6]
dot(u,v) % Ger skalärprodukten (i ON-bas)
cross(u,v) % Ger vektorprodukten
```

```
norm(u) % Ger längden av vektorn u

% Nu ritar vi kurvor och ytor.

% För att rita exempelvis en funktionskurva y=f(x) så skapar vi först en
% vektor med en massa x-värden och sedan motsvarande vektor med y-värden,
% och plottar paren (x,y)
x=0:0.02:5; % Från 0 till 5 i steg om 0.02
y=x.^2; % Kvadraten av alla x-värden, dvs. vi har y=f(x)=x^2
plot(x,y) % Ritar alltså kurvan y=x^2 från 0 till 5
axis equal % Ger samma skala på axlarna
axis normal % Återställer
% Kurvan ritas genom att paren av punkter förbinds med räta linjestycken
x=0:10;
y=sin(x);
plot(x,y) % Nu blev det lite sniket

% För att rita flera kurvor i samma figur skriver man "hold on"
% Man kan också ge kurvorna olika utsende
x=0:0.1:3;
y1=exp(-x);
y2=exp(-x).*sin(x);
y3=x.^4.*exp(-x.^2);
plot(x,y1,'--') % Streckad kurva
hold on
plot(x,y2,'r') % Röd
plot(x,y3,'LineWidth',3) % Fetare
hold off % Återställer

% Vi kan också rita i olika fönster. "subplot(m,n,k)" ger m*n fönster, och
% ritar i fönster k (räknas radvis först).
x=0:0.1:10;
subplot(3,2,1)
y1=sin(x); plot(x,y1)
title('y=sin(x)') % Detta ger en titel på grafen
subplot(3,2,2)
y2=cos(x); plot(x,y2)
title('y=cos(x)')
subplot(3,2,3)
y3=sin(2*x); plot(x,y3)
title('y=sin(2x)')
subplot(3,2,4)
y4=cos(2*x); plot(x,y4)
title('y=cos(2x)')
subplot(3,2,5)
y5=2*sin(x); plot(x,y5)
title('y=2sin(x)')
subplot(3,2,6)
y6=2*cos(x); plot(x,y6)
title('y=2cos(x)')

% Vi kan också rita parametriserade kurvor
subplot(1,1,1) % Återställer till ett fönster
t=linspace(0,2*pi);
x=cos(t);
y=sin(t);
```

```
plot(x,y) % Ger enhetscirkeln
axis equal % Nu ser vi bättre
% Vi har här använt ett alternativt sätt att skapa t-värden. "linspace(a,b)"
% ger 100 värden från a till b, "linspace(a,b,N)" ger N stycken.
t=linspace(-3,3);
x=-1+2*t;
y= 2-3*t;
plot(x,y) % Ger linjen genom (-1,2) med riktningsvektor (2,-3)
grid on % Ritar ut rutnät
axis equal % Nu får vi rätt lutning på linjen
grid off % Tar bort rutnätet igen

% Vi kan även rita kurvor i tre dimensioner, med "plot3"
t=linspace(-3,3);
x=2-  t;
y=    t;
z=1-2*t;
plot3(x,y,z) % Linjen genom (2,0,1) med riktningsvektor (-1,1,-2)
xlabel('x')
ylabel('y')
zlabel('z')
axis equal
grid on
% Prova att vrida på figuren (tryck först på pilen där uppe som går runt).

t=linspace(0,4*pi);
x=cos(t);
y=sin(t);
z=t;
plot3(x,y,z) % Ger en spiral

% För att rita funktionsytor för funktioner z=f(x,y) av två variabler så
% behöver vi först skapa ett "rutnät" av (x,y)-punkter.
x=-3:0.01:3;
y=-3:0.01:3;
[X,Y]=meshgrid(x,y); % Skapar rutnätet
Z=sin(X).*cos(2*Y); % Här är alltså f(x,y)=sin(x)*cos(y)
mesh(X,Y,Z) % Ritar funktionsytan
xlabel('x')
ylabel('y')
zlabel('z')
axis equal

% För att se rutnätet på ytan så väljer vi betydligt längre steg i våra
% x och y-intervall.
x=-3:0.2:3;
y=-3:0.2:3;
[X,Y]=meshgrid(x,y);
Z=sin(X).*cos(2*Y);
mesh(X,Y,Z)
xlabel('x')
ylabel('y')
zlabel('z')
axis equal
% Nu går det också mycket "snabbare" att vrida på figuren.
% Prova nu själv "meshz", som ger referenslinjer till xy-planet, och
```

```
% "meshc", som även ger så kallade nivåkurvor (mer om det i kursen Flerdim).

% Vi ritar planet 2x-y+3z-5=0
x=-3:0.2:3;
y=-3:0.2:3;
[X,Y]=meshgrid(x,y);
Z=(-2*X+Y+5)/3; % Vi löser ut z ur planets ekvation
mesh(X,Y,Z)
xlabel('x')
ylabel('y')
zlabel('z')
axis equal

% Om vi bara vill rita ut en punkt i rummet gör vi såhär
plot3(1,2,3)
% Syns inte, men detta blir bättre
plot3(1,2,3,'*')
% Om vi har punkten sparat som vektor kan vi göra såhär
P=[1 2 3];
plot3(P(1),P(2),P(3),'*')


% Ta redan nu för vana att spara det ni skriver i ett så kallat "script".
% Från HOME-menyn väljer ni då "New Script". Då kommer ett fönster upp som ni
% kan skriva i. Ni kan, efter att ha namngett och sparat i en fil, köra
% alla kommando i fönstret med "Run" från EDITOR-menyn.
% Ni kan sedan också köra hela filen med ett enda kommando:
% Om ni exempelvis sparat i filen Test.m så skriver ni bara
% "Test" i kommandofönstret.
```