

Notat 1, del 3 (av totalt 9 deler)

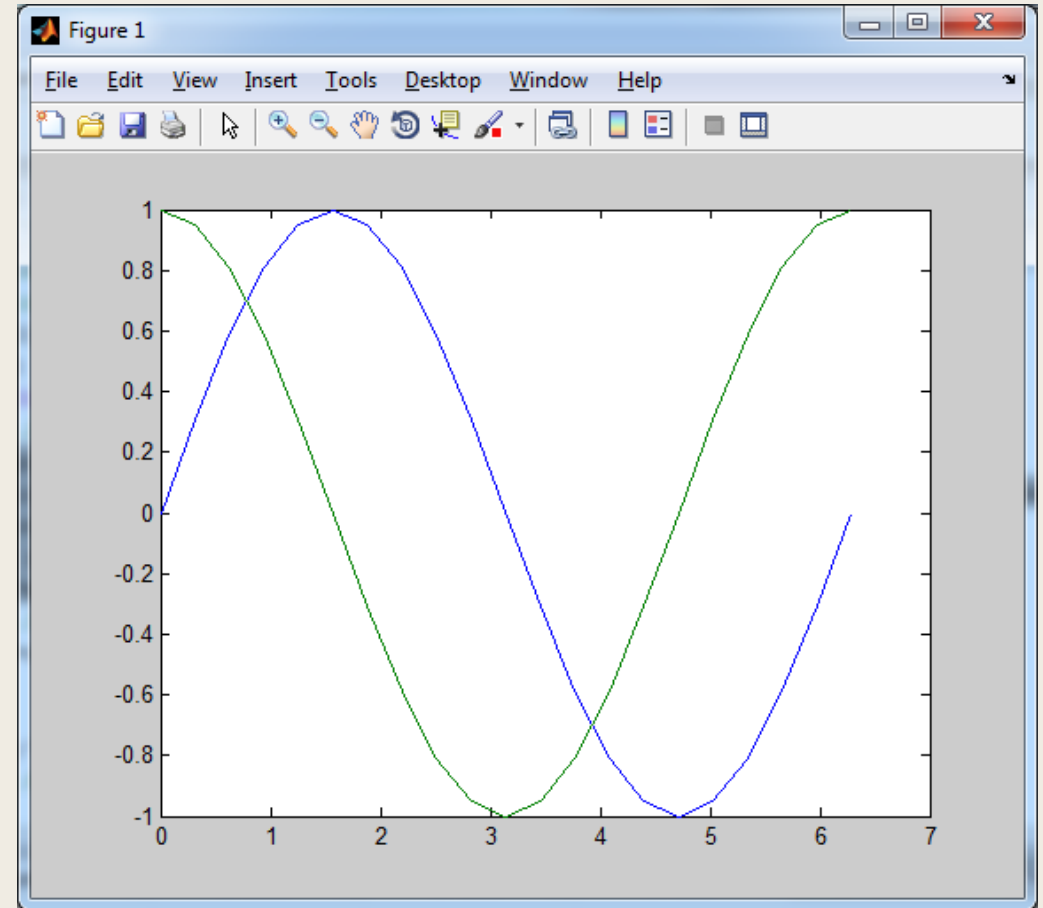
Plotting

Ulike plottealternativer

- 2D-plott:
 - Lineære plott
 - logaritmiske plott
 - polarplott
 - en eller flere kurver i ett plott
 - punktkurver
 - stolper
 - histogram
- 3D-plott:
 - kurve i tredimensjonalt rom
 - «netting»-plott (mesh plot)
 - overflateplott (surface plot)

Plottet kommer i eget vindu

- Vi kan editere på plottet i dette vinduet
- Du kan lagre plottet til .pdf-format som er det beste for bruk i rapport, se siste lysark.
- Vi kan ha flere plott i ett vindu og/eller flere grafikkvinduer



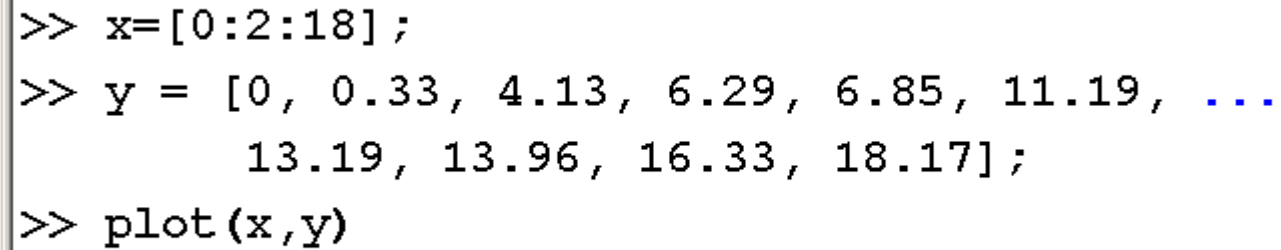
2D-plot: Eksempel på x-y data

- Tid (“time”) er fri variabel
- Avstand (“distance”) er avhengig variabel

time, sec	Distance, Ft
0	0
2	0.33
4	4.13
6	6.29
8	6.85
10	11.19
12	13.19
14	13.96
16	16.33
18	18.17

x-y og plot-funksjonen

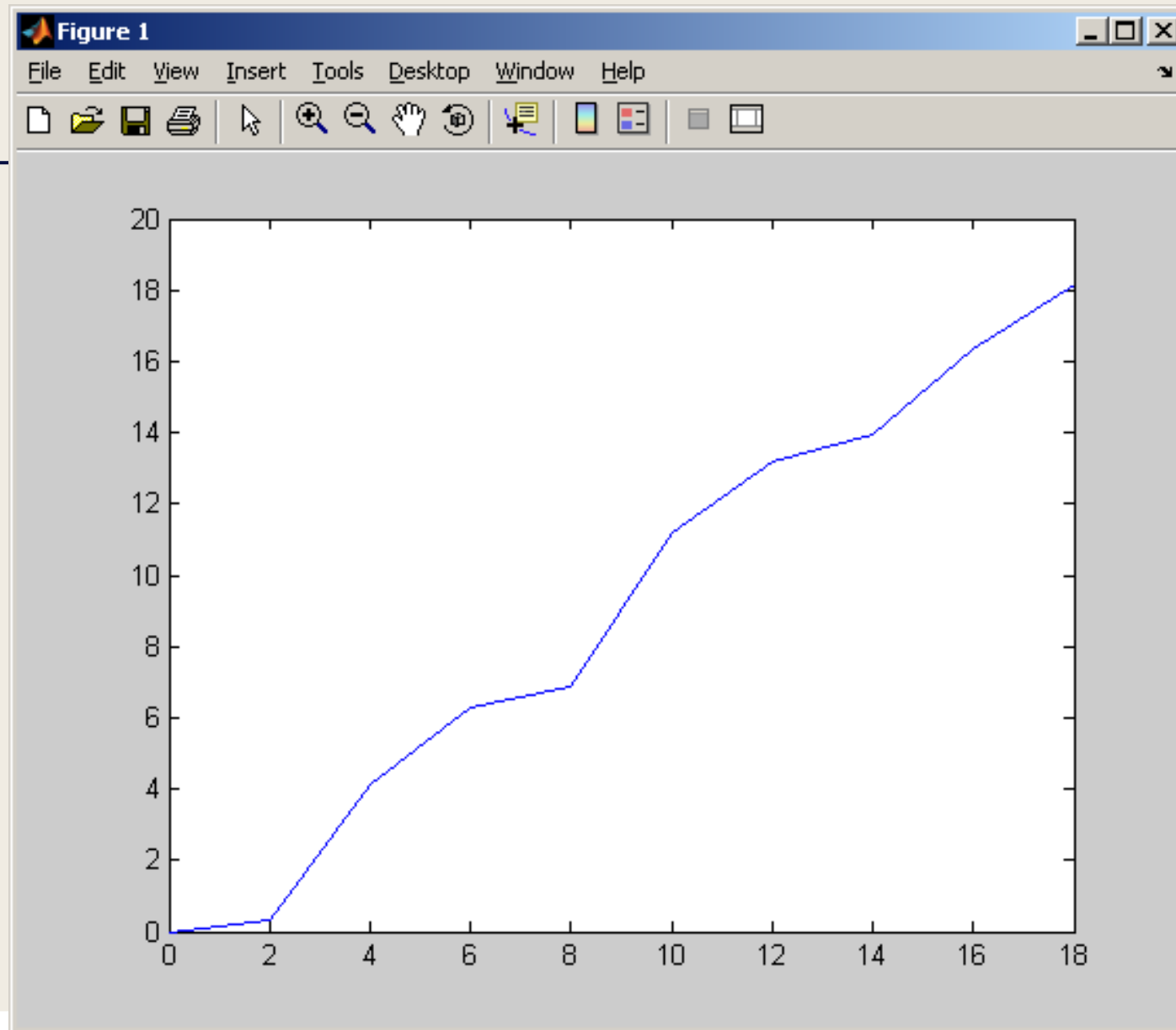
I kommandovinduet:

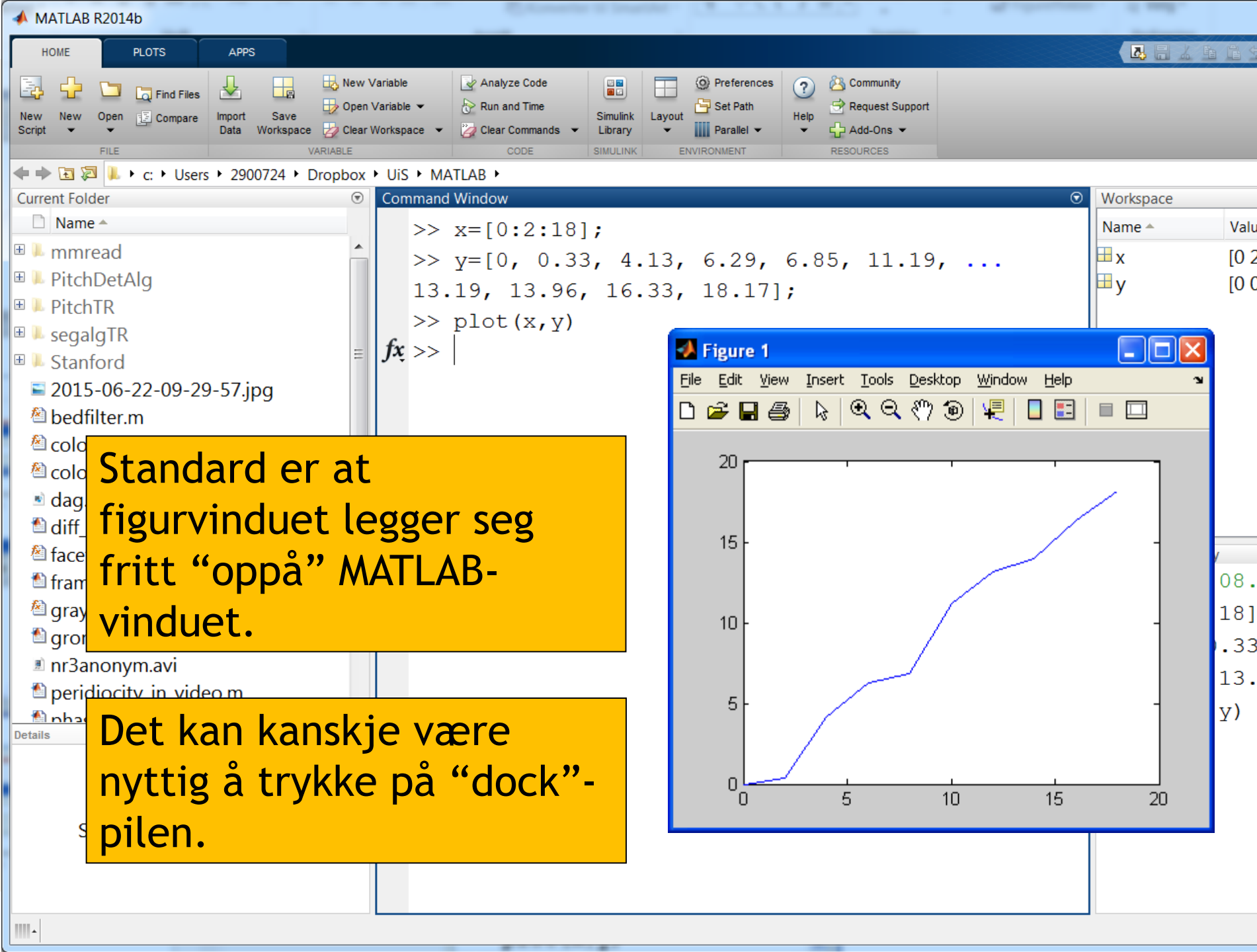


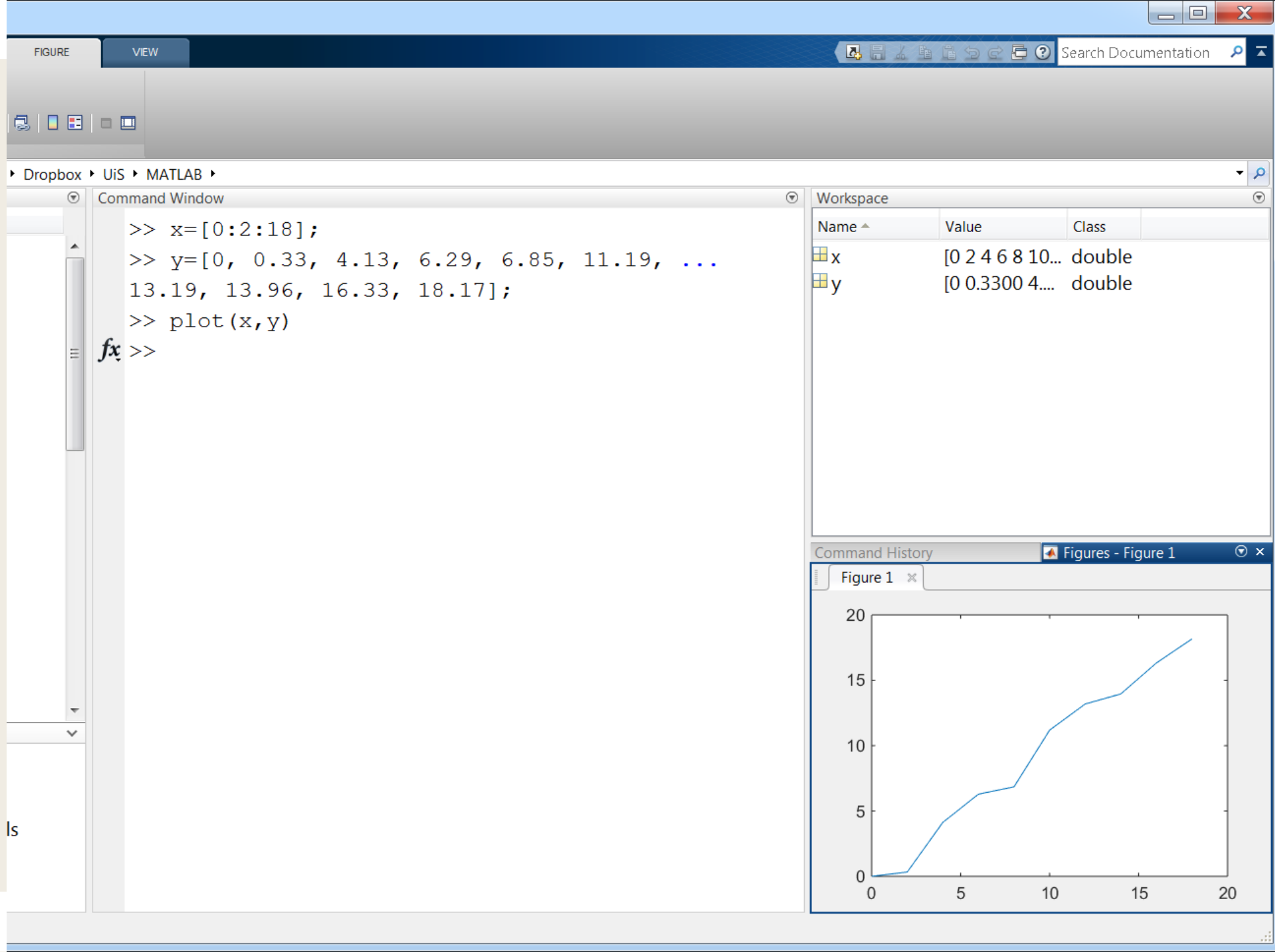
```
>> x=[0:2:18];  
>> y = [0, 0.33, 4.13, 6.29, 6.85, 11.19, ...  
        13.19, 13.96, 16.33, 18.17];  
>> plot(x,y)
```

The image shows a MATLAB command window with a scroll bar on the right and a status bar at the bottom right containing the text 'OVR' and a double-slash icon. The code entered is as follows:

PS: Variabelnavnene trenger ikke være x og y. Velg det som passer!





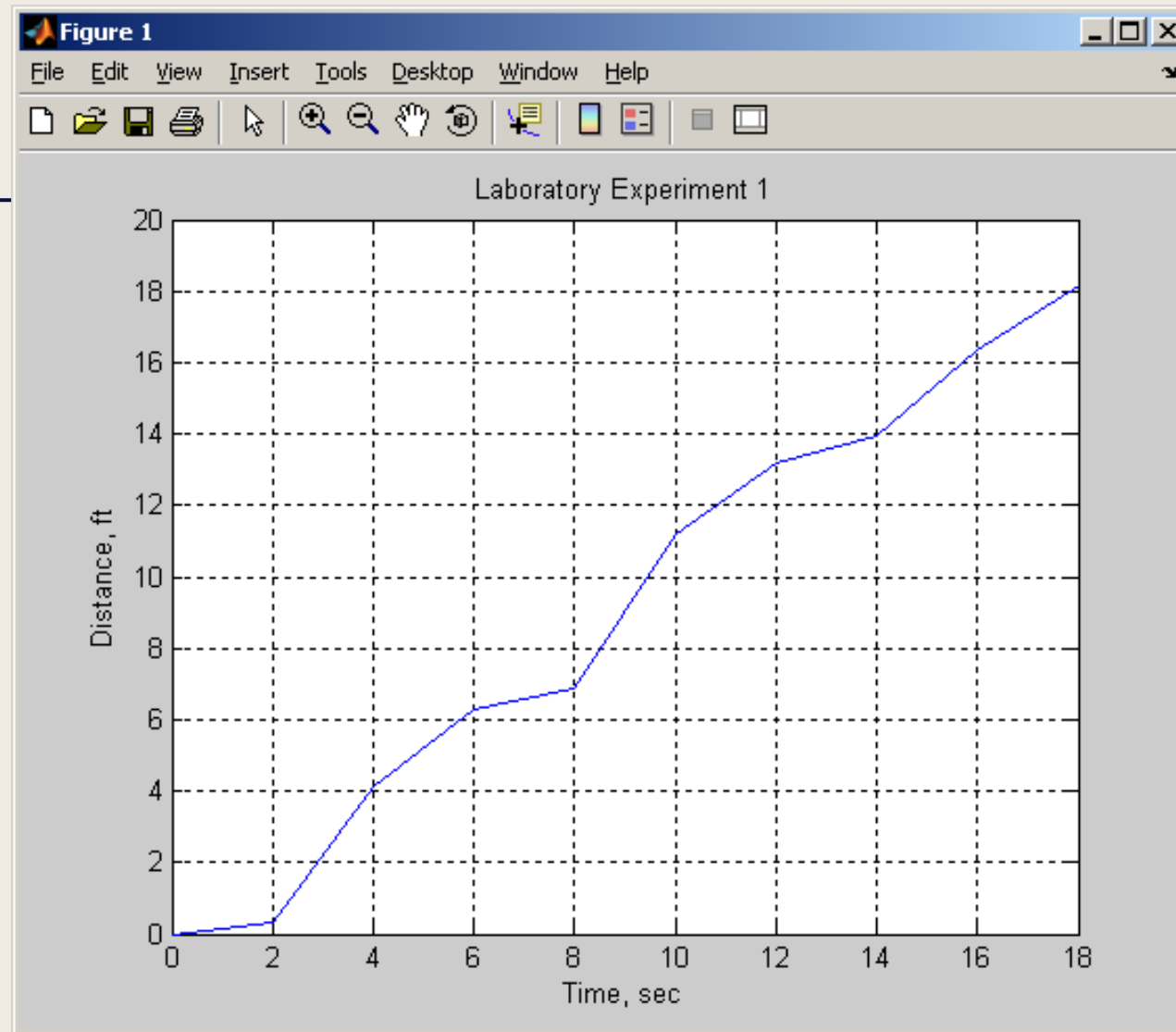


Ingeniører legger alltid til:

- Tittel
- Forklaring og enhet for x-aksen
- Forklaring og enhet for y-aksen
- Og av og til en “grid”

```
>> title('Laboratory Experiment 1')  
>> xlabel('Time, sec')  
>> ylabel('Distance, ft')  
>> grid on
```

OVR



Linjefarge, linjetype og markører

Table 5.2 Line, Mark and Color Options

Line Type	Indicator	Point Type	Indicator	Color	Indicator
solid	-	point	.	blue	b
dotted	:	circle	o	green	g
dash-dot	-.	x-mark	x	red	r
dashed	--	plus	+	cyan	c
		star	*	magenta	m
		square	s	yellow	y
		diamond	d	black	k
		triangle down	v		
		triangle up	^		
		triangle left	<		
		triangle right	>		
		pentagram	p		
		hexagram	h		

Tips:

>> help plot

plot(x,y,':ok')

- Her plotter vi x mot y
- : betyr prikket linje
- o betyr at vi vil vise en sirkel på hvert punkt
- k betyr at linjen skal plottes i svart farge
 - ('b' brukes for blå)

Tegne flere figurer

- MATLAB tegner et helt nytt plott i aktiv figur hver gang plot-funksjonen brukes.
- Dersom ny figur ønskes, bruk figure-funksjonen

Eks.:

`figure(2)`

Tegne flere linjer

- hold on
 - “Fryser” aktivt plott, slik at flere linjer kan legges inn i samme plott. For å ta av denne funksjonen, bruk
- hold off

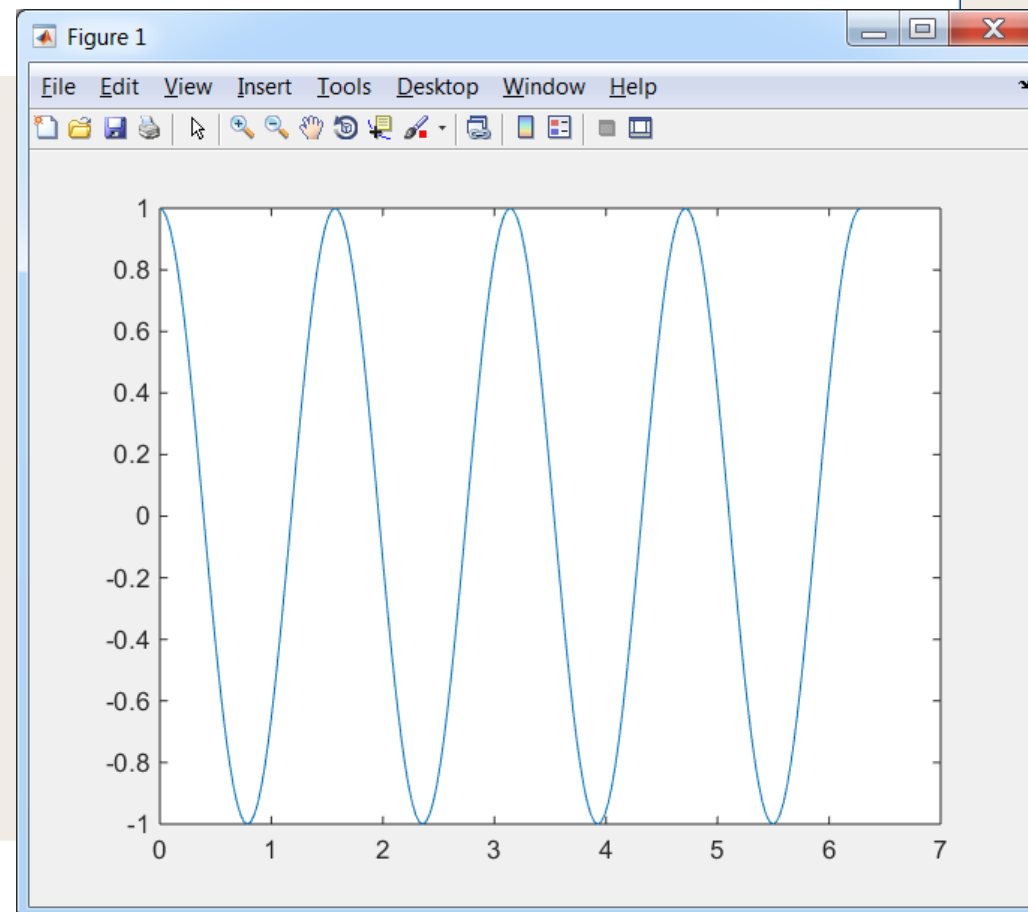
Command Window

```
>> x = 0:pi/100:2*pi;  
>> y1 = cos(x*4);  
>> plot(x,y1)  
fx >>
```

Første linje tegnes i
lyseblått, svak farge.

Alternativ:

```
>> plot(x,y1,'b')
```



Command Window

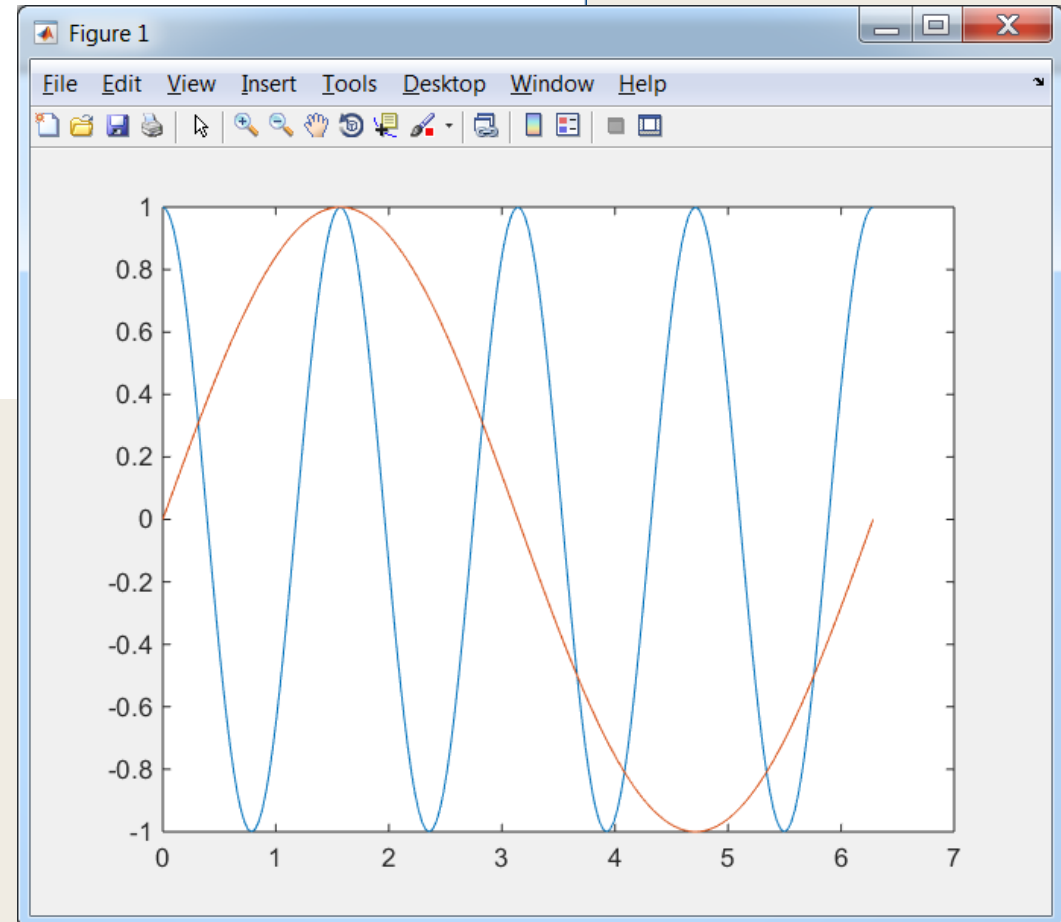
```
>> x = 0:pi/100:2*pi;  
>> y1 = cos(x*4);  
>> plot(x,y1)  
>> hold on  
>> y2 = sin(x);  
>> plot(x, y2)  
>> hold off  
fx>> |
```

Plottet "fryses"

Linje nr. 2 tegnes i lyserødt,
svak farge

Alternativ:

```
>> plot(x,y1,'b')  
>> plot(x,y2,'r')
```



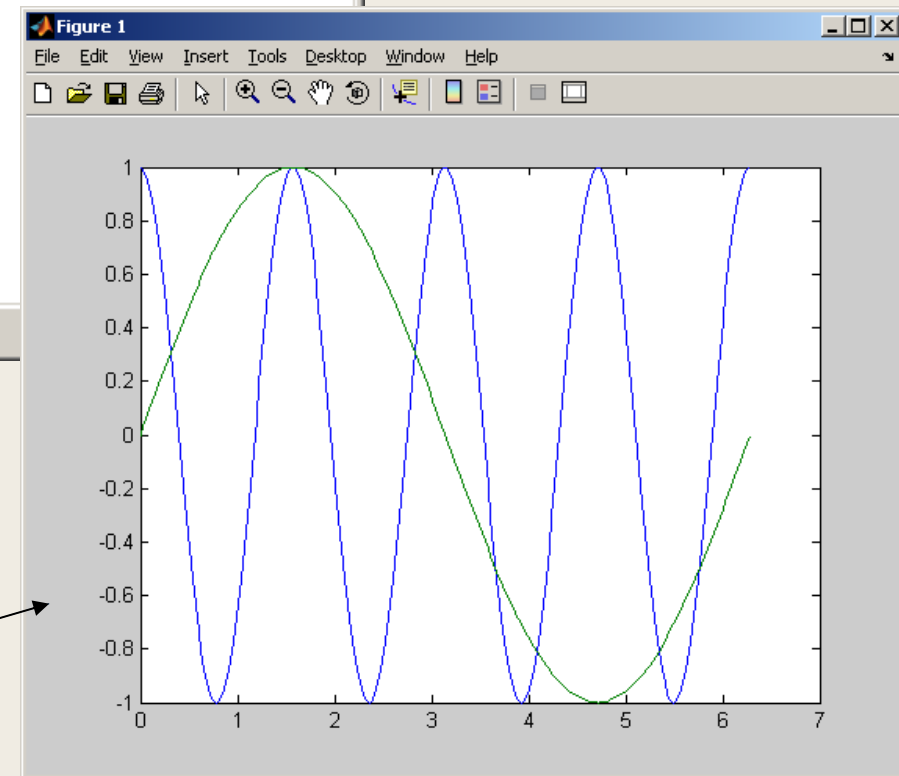
Flere linjer vha én plot-kommando

```
>> x = 0:pi/100:2*pi;  
>> y1 = cos(x*4);  
>> y2 = sin(x);  
>> plot(x,y1,x,y2)  
>>
```

Her: To linjer. Krever to vektorpar

Ny farge for hver linje, igjen svake farge. Alternativ

```
>> plot(x,y1,'b',x,y2,'r')
```



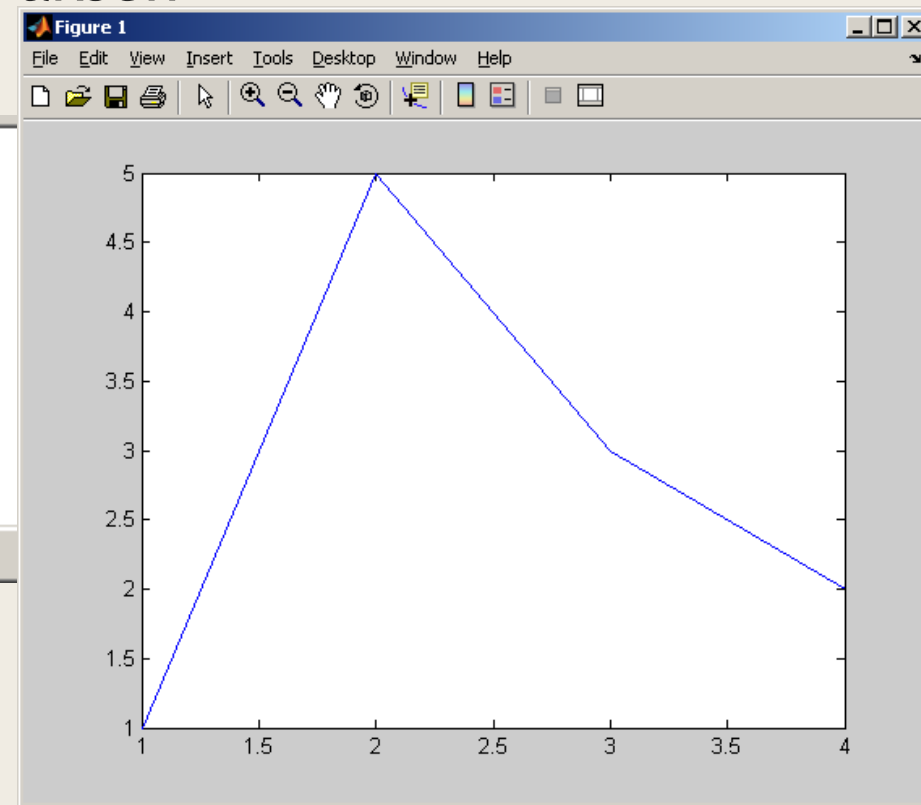
Bare én vektor går også an

- Indeksverdiene er da langs x-aksen

```
>> z=[1,5,3,2];  
>> plot(z)  
>>
```

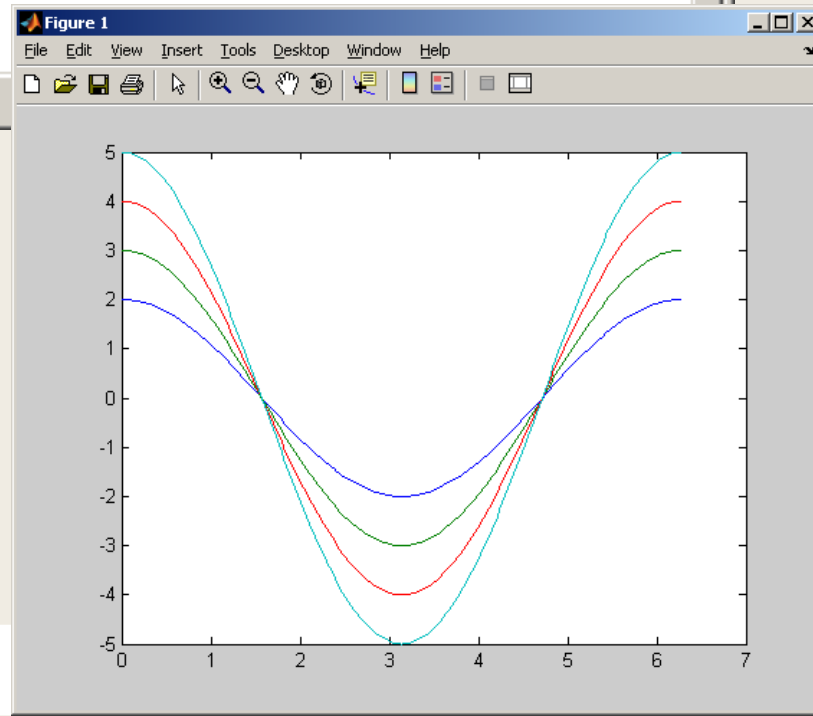
For å vise hvor verdiene er

```
>> plot(z, '-*')
```



Flere linjer kan også plottes samtidig

```
>> Y1 = cos(X)*2;  
>> Y2 = cos(X)*3;  
>> Y3 = cos(X)*4;  
>> Y4 = cos(X)*5;  
>> plot(X, Y1, X, Y2, X, Y3, X, Y4)  
>> |
```



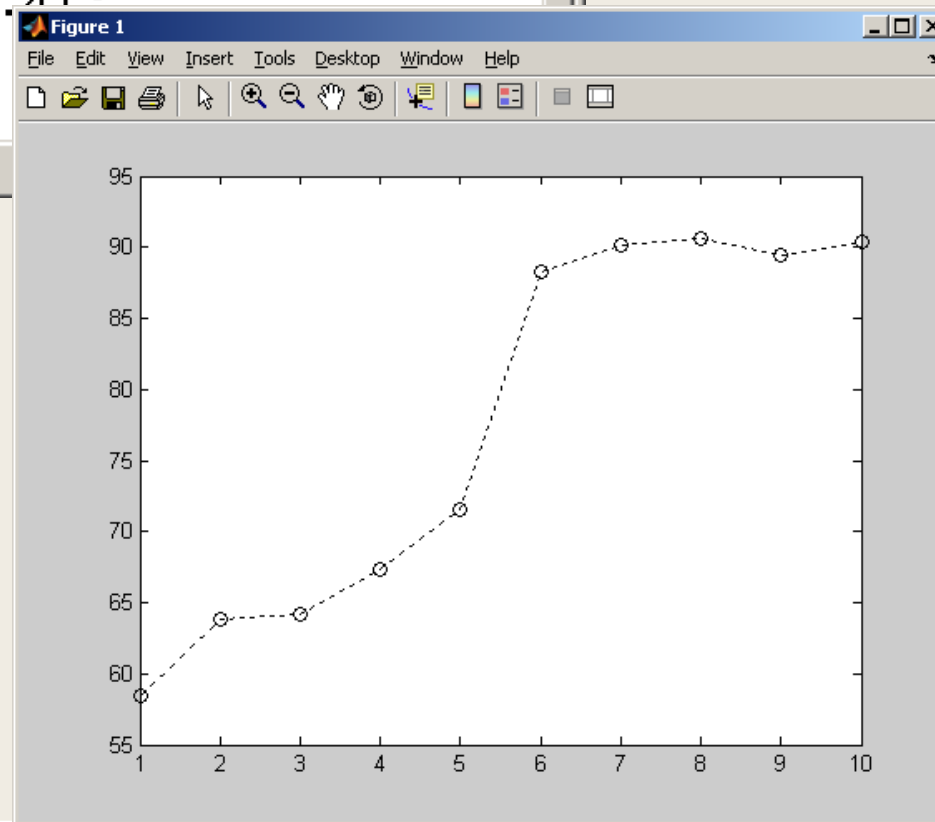
Eksempel på endring av linjetype

```
>> x = [1:10];  
>> y = [ 58.5, 63.8, 64.2, 67.3, 71.5, 88.3, ...  
        90.1, 90.6, 89.5, 90.4];  
>> plot(x,y, ':ok')  
>>
```

: = stiplet linje

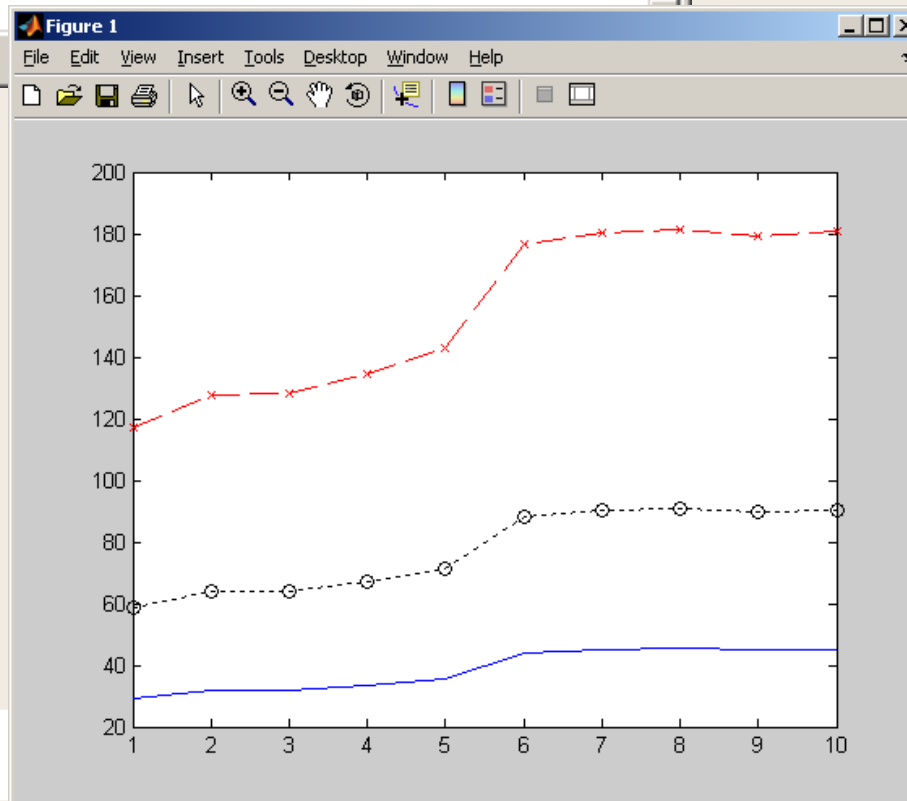
o = sirkler som punktmarkører

k = svart



Eksempel på flere linjetyper

```
>> x = [1:10];  
>> y = [ 58.5, 63.8, 64.2, 67.3, 71.5, 88.3,...  
        90.1, 90.6, 89.5, 90.4];  
>> plot(x,y,':ok',x,y*2,'--xr',x,y/2,'-b')  
>>
```

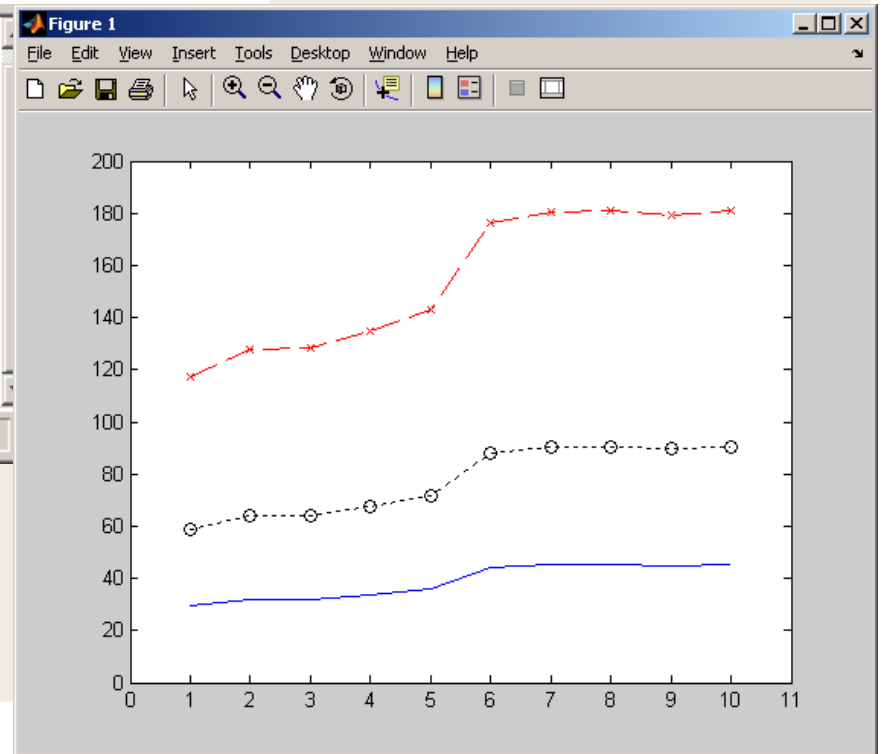


Skalering av aksene

- Dersom man ønsker noe annet enn standard skalering, brukes **axis**-funksjonen

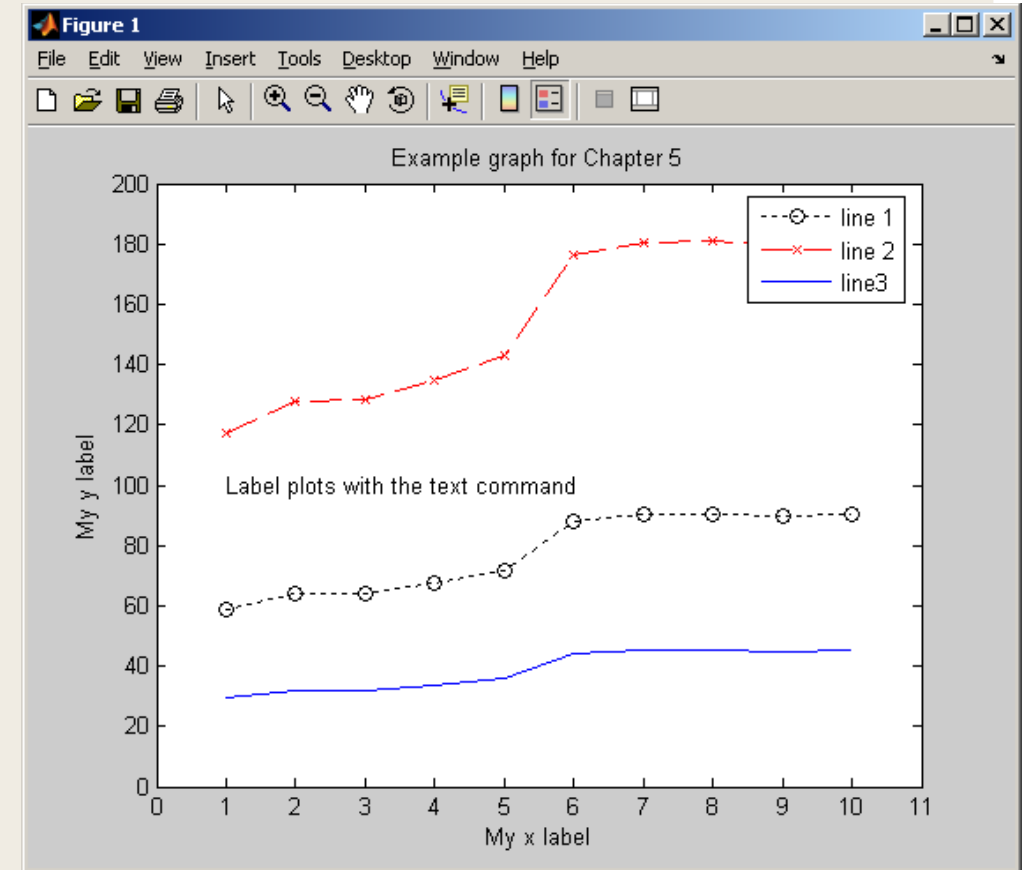
`axis([xmin,xmax,ymin,ymax])`

```
>> x = [1:10];  
>> y = [ 58.5, 63.8, 64.2, 67.3, 71.5, 88.3, ...  
        90.1, 90.6, 89.5, 90.4];  
>> plot(x,y,':ok',x,y*2,'--xr',x,y/2,'-b')  
>> axis([0,11,0,200])  
>>
```



Legend og text

```
>> x = [1:10];  
>> y = [ 58.5, 63.8, 64.2, 67.3, 71.5, 88.3,...  
        90.1, 90.6, 89.5, 90.4];  
>> plot(x,y,':ok',x,y*2,'--xr',x,y/2,'-b')  
>> axis([0,11,0,200])  
>> legend('line 1', 'line 2', 'line3')  
>> text(1,100,'Label plots with the text command')  
>> xlabel('My x label'), ylabel('My y label')  
>> title('Example graph for Chapter 5')
```



Flere plott i et figurvindu

- subplot-funksjonen

subplot(m,n,p)

rader

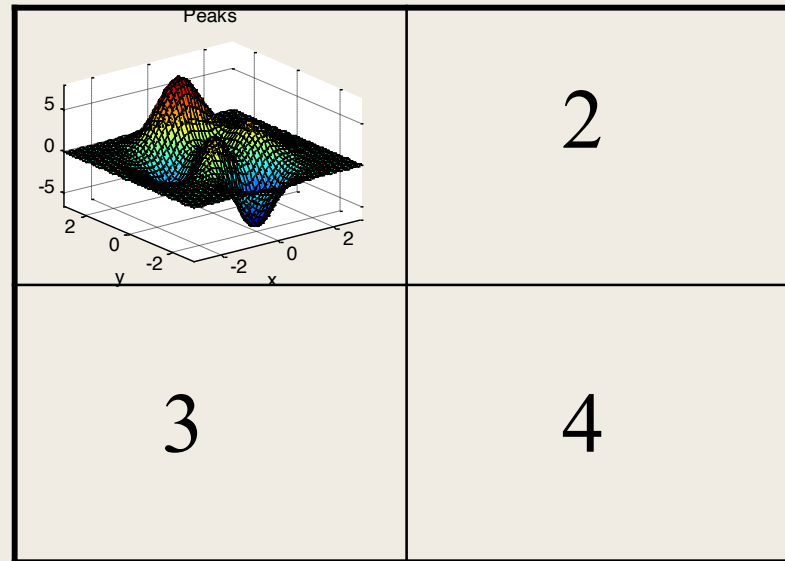
kolonner

plottnummer

Subplot(2,2,1)

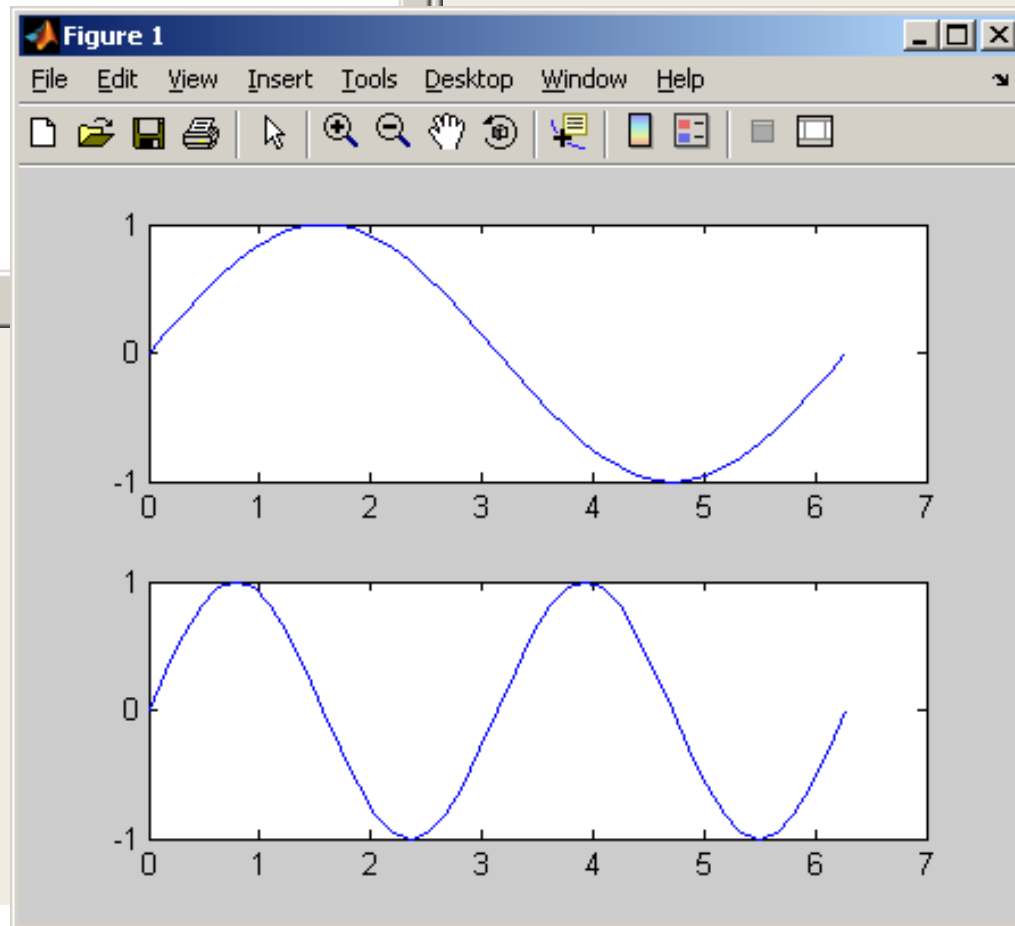
2 columns

2
rows



Eksempel på bruk av subplot

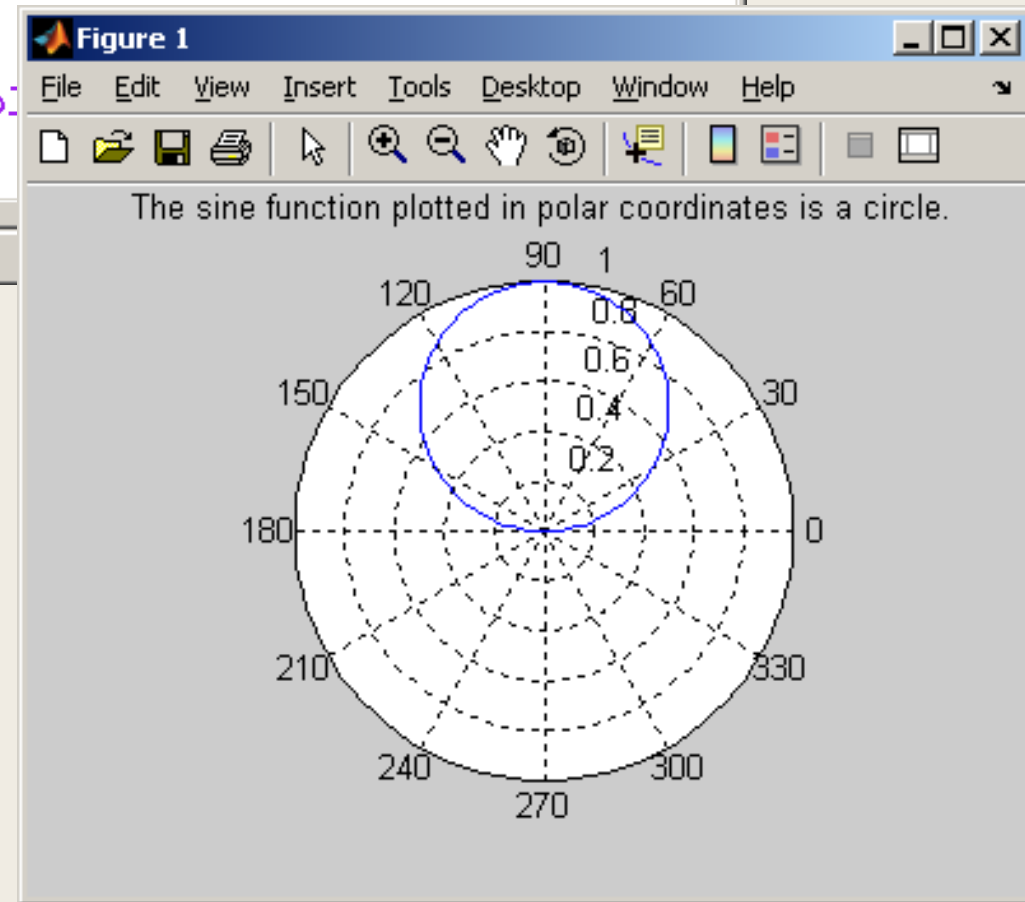
```
>> x=0:pi/20:2*pi;  
>> subplot(2,1,1)  
>> plot(x,sin(x))  
>> subplot(2,1,2)  
>> plot(x,sin(2*x))  
>>
```



Polarplot – et eksempel

```
>> x=0:pi/100:pi;  
>> y=sin(x);  
>> polar(x,y)  
>> title('The sine function plotted in polar  
>> |
```

Fin å bruke til å vise omrisset av objektet som scannes med ultralydsensoren i Legoprosjektet "Mål areal av eske".



Logaritmiske plot

- **plot** - bruker lineær skala på både x- og y-akse
- **semilogy** - bruker \log_{10} -skala på y-aksen
- **semilogx** - bruker \log_{10} -skala på x-aksen
- **loglog** - bruker \log_{10} -skala på begge aksene

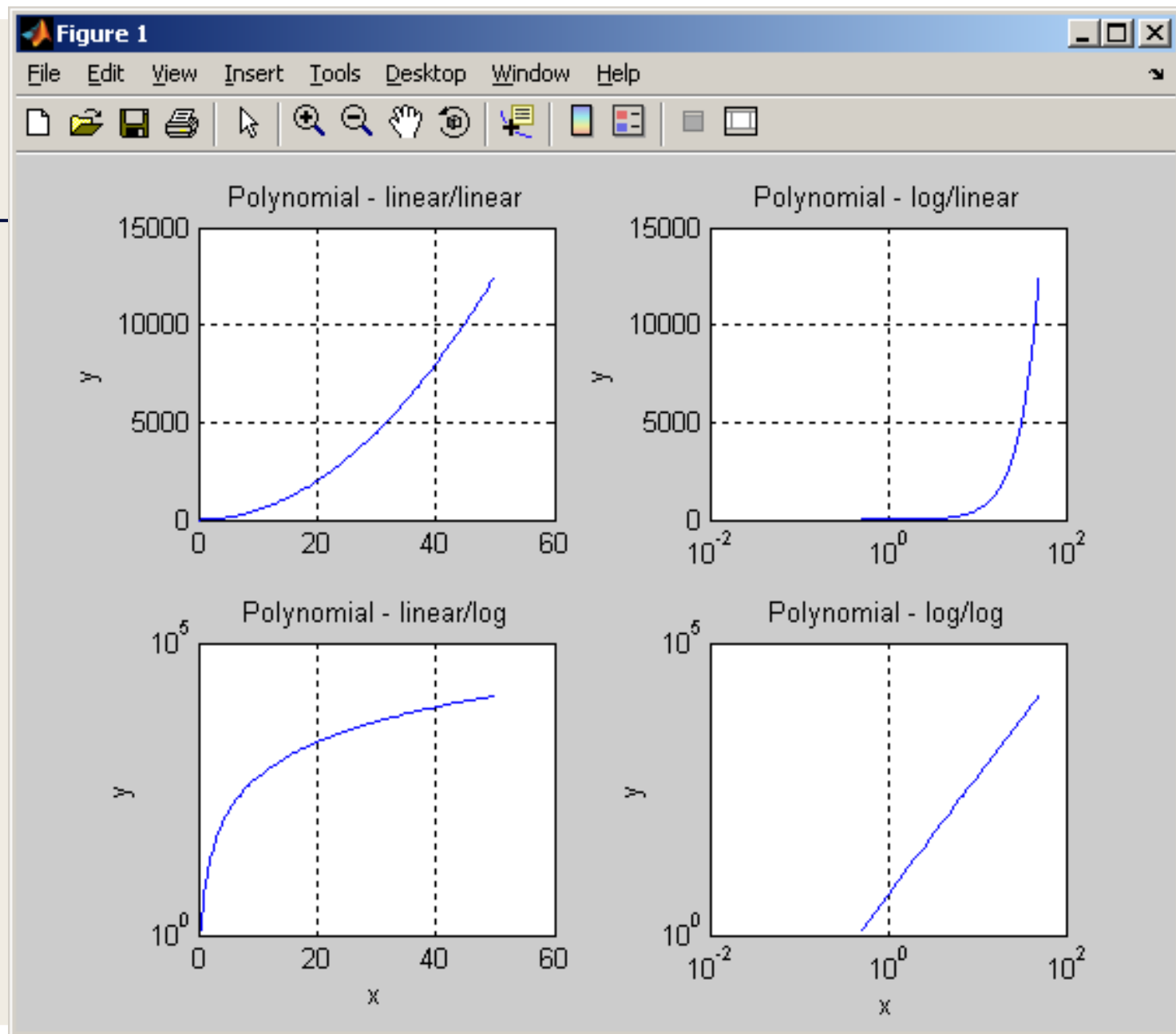
I editorvinduet

```
1 x = 0:0.5:50;
2 y = 5*x.^2;
3 subplot(2,2,1)
4 plot(x,y)
5     title('Polynomial - linear/linear')
6     ylabel('y'), grid
7 subplot(2,2,2)
8 semilogx(x,y)
9     title('Polynomial - log/linear')
10    ylabel('y'), grid
11 subplot(2,2,3)
12 semilogy(x,y)
13     title('Polynomial - linear/log')
14     xlabel('x'), ylabel('y'), grid
15 subplot(2,2,4)
16 loglog(x,y)
17     title('Polynomial - log/log')
18     xlabel('x'), ylabel('y'), grid
19
```

Dersom du har store forskjeller i y-verdier fra f.eks.

[10, 1, 0.1, 0.01, 0.001]

så vil linær y-akse ikke gi informasjon om de 3 siste elementene, mens med logaritmisk y-akse vil du få frem informasjonen også om de 3 siste



Histogram

- Et histogram viser fordelingen av en gitt mengde med tall. Tallene blir gruppert i "bokser". Y-aksen representerer "antall", men kan normaliseres til f.eks. fordelingsfunksjon som viser sannsynlighet.
- `>> x=rand(1,20)`

Columns 1 through 10

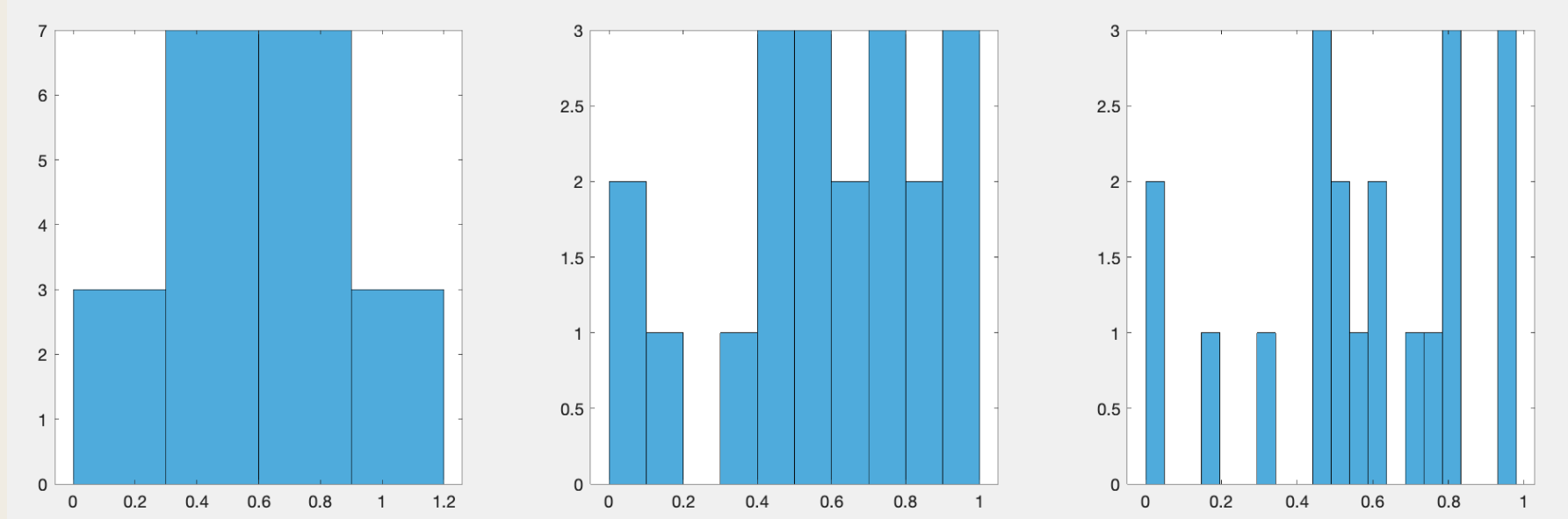
0.8497	0.4241	0.4988	0.9063	0.5742	0.0821	0.9724	0.2266	0.6519	0.2930
--------	--------	--------	--------	--------	--------	--------	--------	--------	--------

Columns 11 through 20

0.7855	0.9061	0.7622	0.5060	0.0530	0.5632	0.2259	0.9009	0.4998	0.2952
--------	--------	--------	--------	--------	--------	--------	--------	--------	--------

Histogram viser antall elementer i vektoren x som har verdi innenfor bredden av hver søyle

- `subplot(1,3,1); histogram(x);`
- `subplot(1,3,2); histogram(x,10) % tallet 10 kalles for bins`
- `subplot(1,3,3); histogram(x,20)`

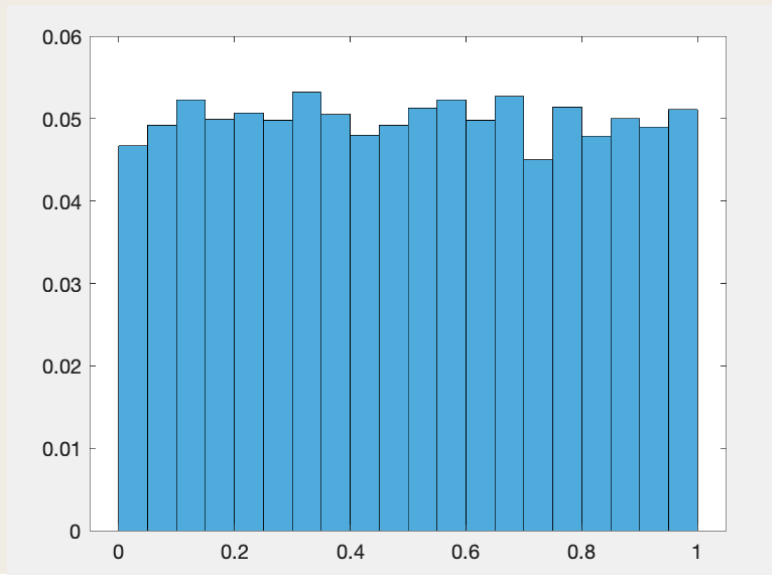


Histogram/fordelingsfunksjon av tilfeldig tall

- `rand(N)`
- Lager en NxN matrise med tilfeldige tall mellom 0.0 og 1.0 med uniform fordeling som betyr at sannsynligheten er jevnt fordelt

```
>> a=rand(1,10000);
```

```
>> histogram(a, 'Normalization', 'probability')
```

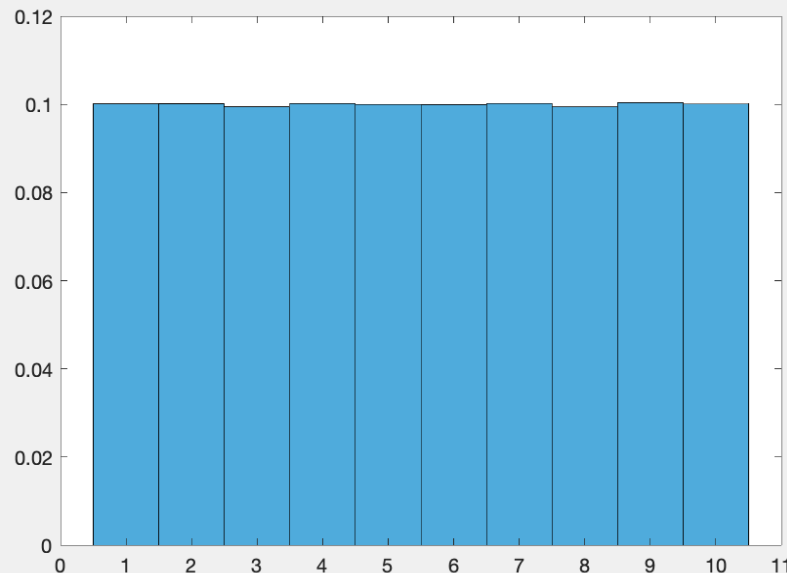


Histogram/fordelingsfunksjon av tilfeldig tall

- `randi(IMAX, N)`
- Lager en NxN matrise med tilfeldige heltall fra 1 til IMAX med uniform fordeling

```
>> a=randi(10,1000);
```

```
>> histogram(a, 'Normalization', 'probability')
```

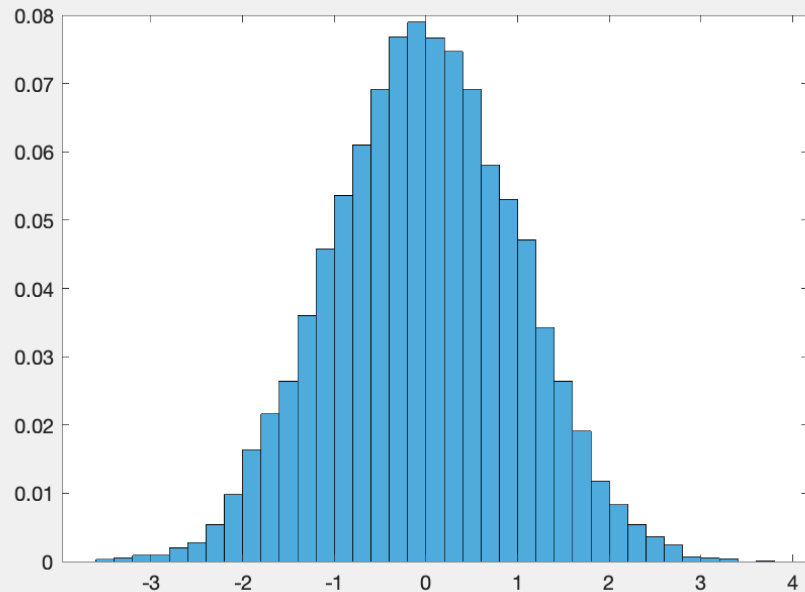


Histogram/fordelingsfunksjon av tilfeldig tall

- `randn(N)`
- Lager en NxN matrise med normalfordelte verdier med middelerverdi 0 og varians på 1

```
>> a=randn(1,10000);
```

```
>> histogram(a, 'Normalization', 'probability')
```



Tredimensjonale plot

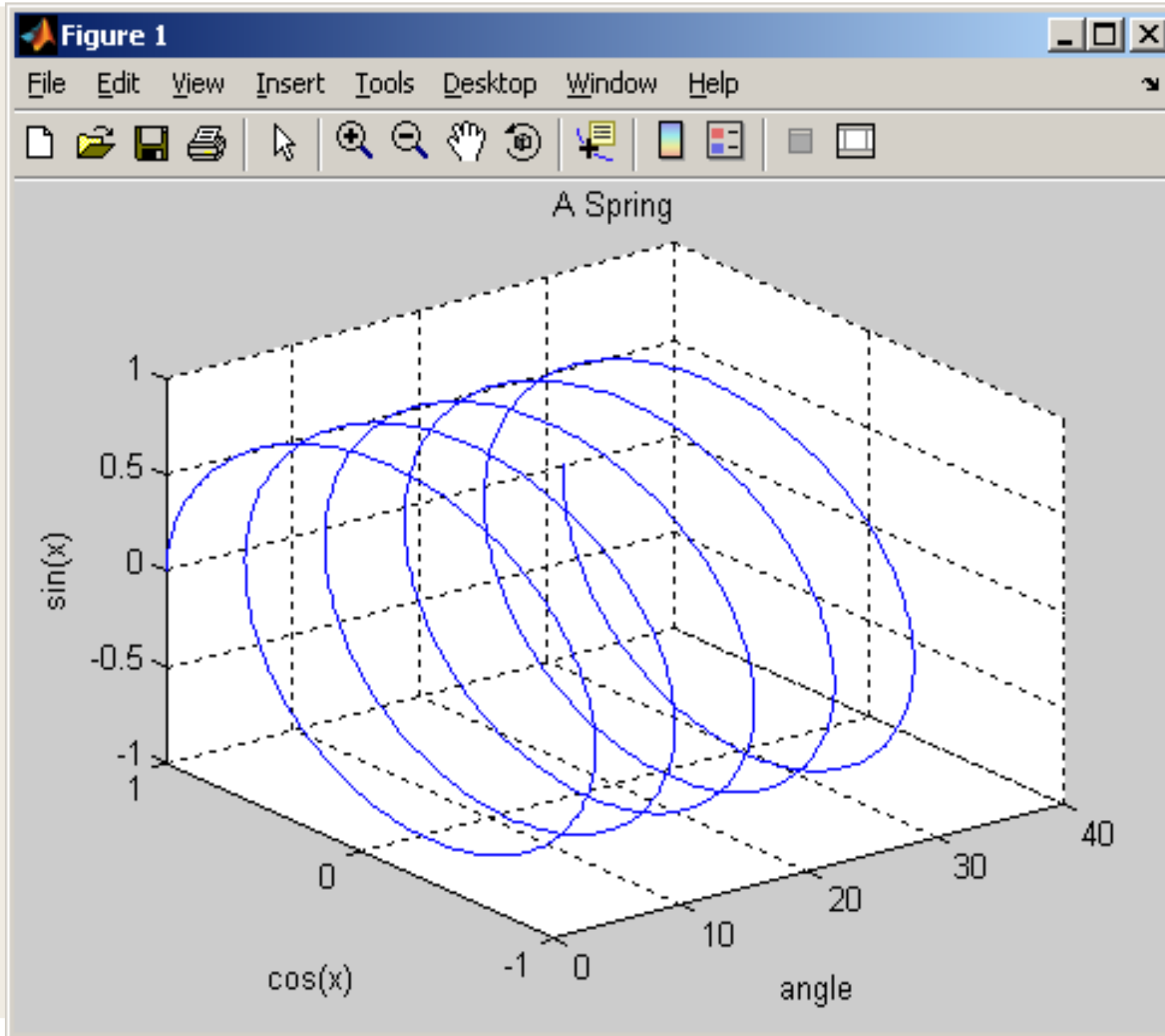
- Når vi har både x-, y- og z-verdi
- **plot3** for linjer i 3D-rommet
- **mesh** for å få fram flater i en nettingstruktur
- **surf** som mesh, men med farger på flaten
- **contour** høydekurve-plott

plot3-eksempel

- I editorvinduet:

```
1  x = linspace(0,10*pi,1000);  
2  y = cos(x);  
3  z = sin(x);  
4  plot3(x,y,z)  
5  grid  
6  xlabel('angle'), ylabel('cos(x)')  
7  zlabel('sin(x)'), title('A Spring')  
8
```

script Ln 1 Col 1 OVR

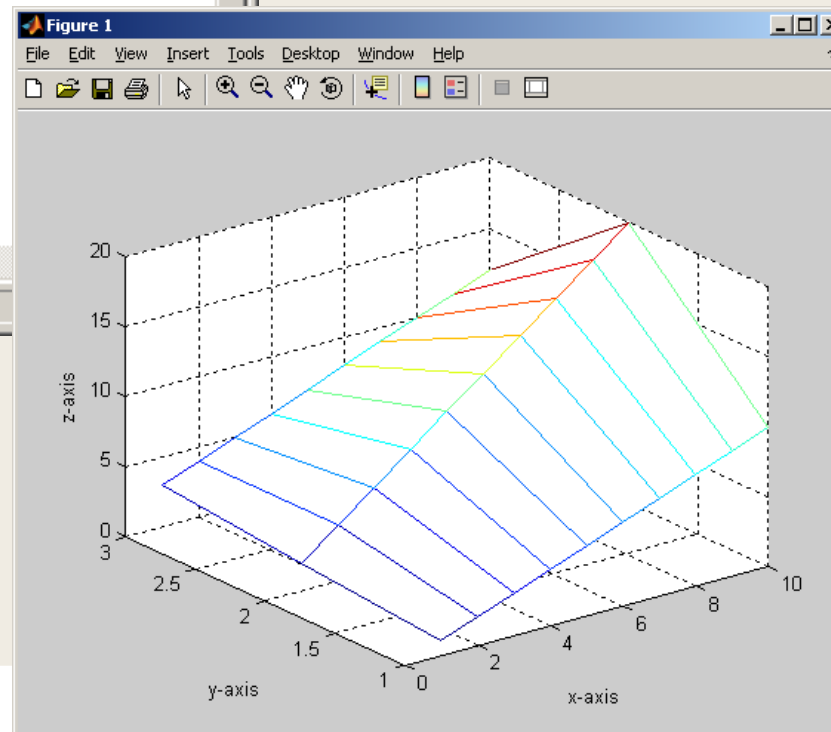


MATLAB
er konsistent
i forhold til
høyrehånds-
regelen!

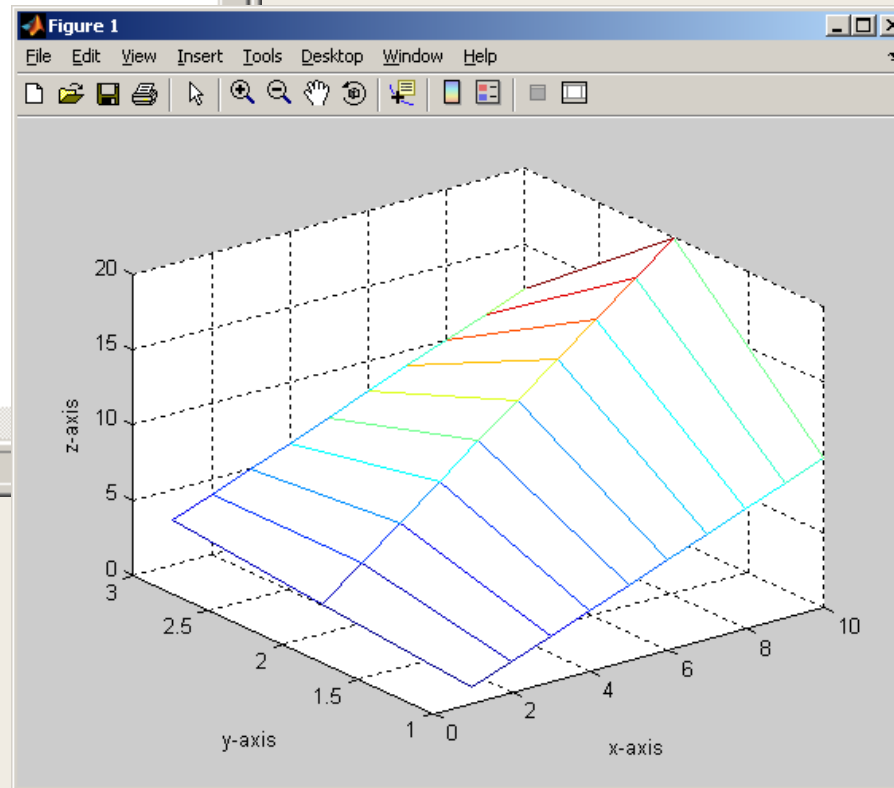
mesh-eksempel

```
1  z = [1,2,3,4,5,6,7,8,9,10;  
2      2,4,6,8,10,12,14,16,18,20  
3      3,4,5,6,7,8,9,10,11,12];  
4  mesh(z)  
5  xlabel('x-axis')  
6  ylabel('y-axis')  
7  zlabel('z-axis')  
8
```

x- og y-koordinatene
er matrisens
indeksverdier

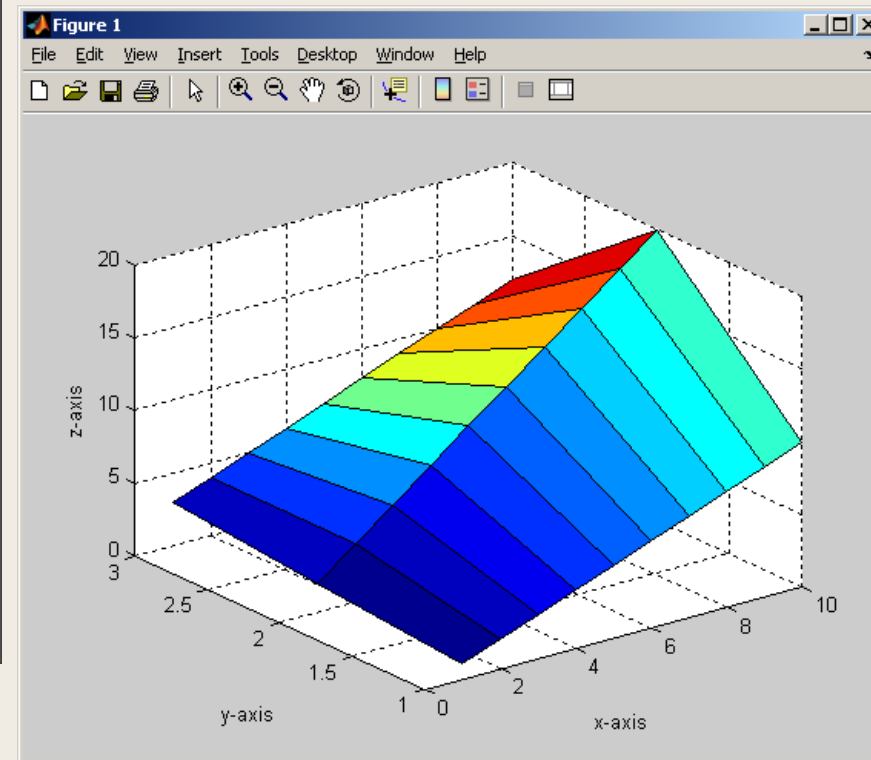


```
1  z = [1,2,3,4,5,6,7,8,9,10;  
2      2,4,6,8,10,12,14,16,18,20  
3      3,4,5,6,7,8,9,10,11,12];  
4  x = linspace(1,50,10);  
5  y = linspace(500,1000,3);  
6  
7  mesh(z)  
8  xlabel('x-axis')  
9  ylabel('y-axis')  
10 zlabel('z-axis')  
11
```



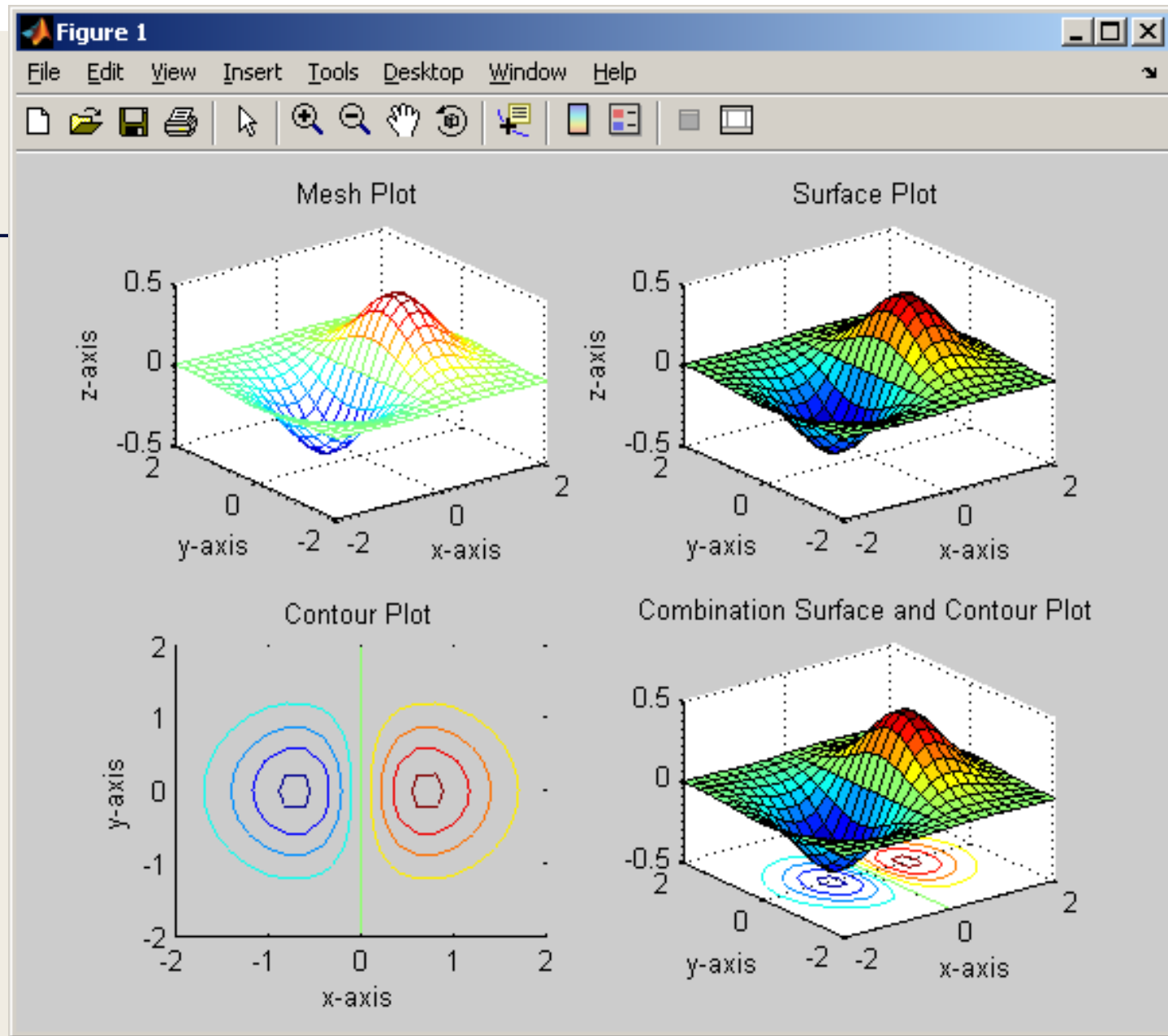
surf-eksempel

```
1  z = [1,2,3,4,5,6,7,8,9,10;  
2      2,4,6,8,10,12,14,16,18,20  
3      3,4,5,6,7,8,9,10,11,12];  
4  x = linspace(1,50,10);  
5  y = linspace(500,1000,3);  
6  
7  surf(z)  
8  xlabel('x-axis')  
9  ylabel('y-axis')  
10 zlabel('z-axis')  
11
```

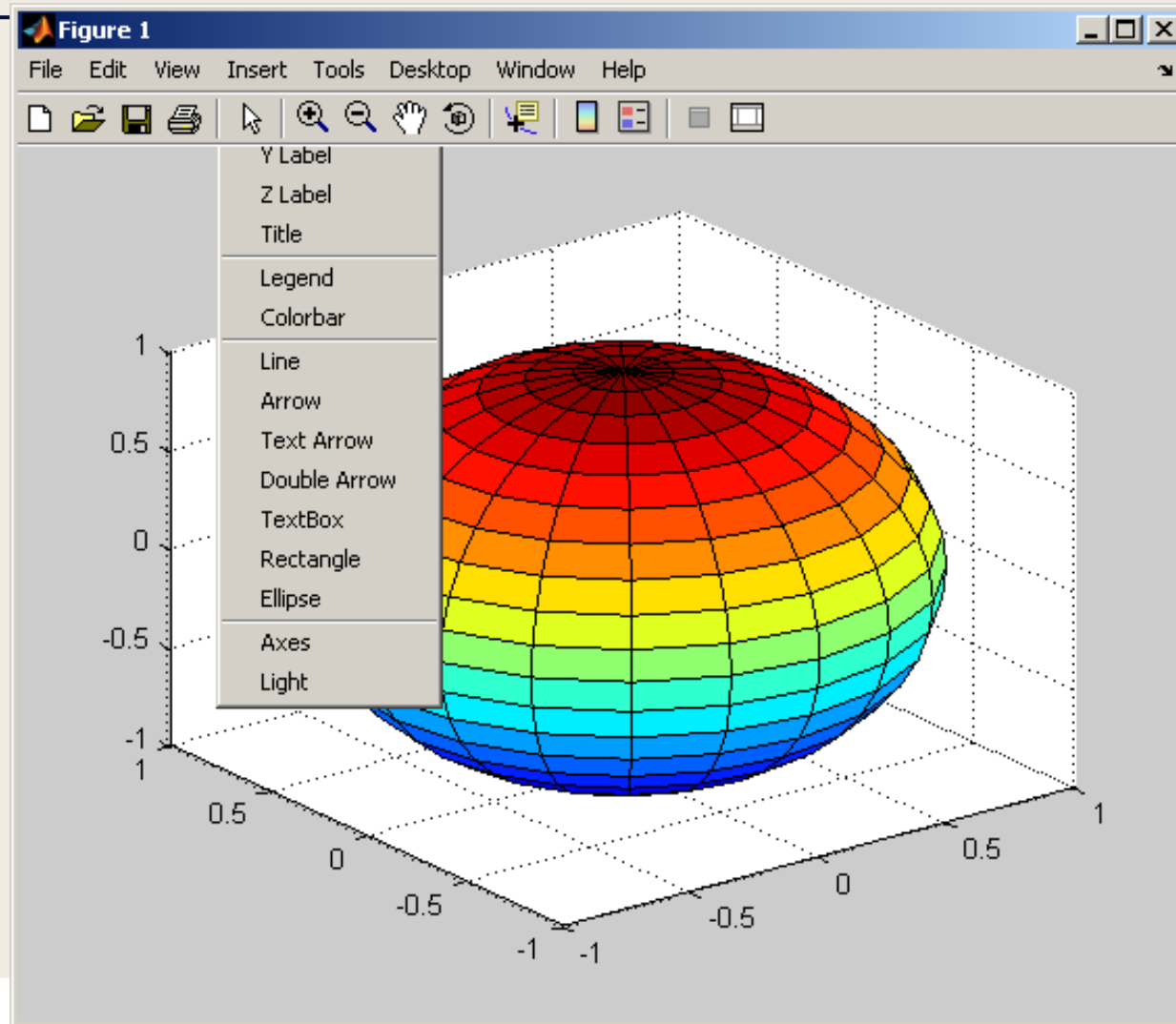


Contour-eksempel

```
1  x= [-2:0.2:2];
2  y= [-2:0.2:2];
3  [X,Y] = meshgrid(x,y);
4  Z = X.*exp(-X.^2 - Y.^2);
5
6  subplot(2,2,1)
7  mesh(X,Y,Z)
8  title('Mesh Plot'), xlabel('x-axis'), ylabel('y-axis'), zlabel('z-axis')
9  subplot(2,2,2)
10 surf(X,Y,Z)
11 title('Surface Plot'), xlabel('x-axis'), ylabel('y-axis'), zlabel('z-axis')
12 subplot(2,2,3)
13 contour(X,Y,Z)
14 xlabel('x-axis'), ylabel('y-axis'), title('Contour Plot')
15 subplot(2,2,4)
16 surfc(X,Y,Z)
17 xlabel('x-axis'), ylabel('y-axis')
18 title('Combination Surface and Contour Plot')
19
```



Redigere plot i figurvinduet



Lagre plot

- Lagre et plott fra figurvinduet ved bruk av `Save as...`
 - `.fig`, MATLAB figurfil. Absolutt alt av informasjon/data ligger i lagret i filen og du kan derfor åpne den og fortsette redigering senere.
 - `.pdf`, PDF-format. Det absolutt beste formatet for resultater i form av tidsresponser. Ulempen med å bruke `Save as...` menyen til dette er at du får en stor mye hvitt område rundt figuren (eng: *bounding box*).
 - Bruk derfor filen `SaveMyFigure.m` som fjerner dette hvite området og lagrer både en `.fig` og en `.pdf`.
- Ikke bruk `.jpg` eller `.png` til å lagre tidsresponser.
- Les deg gjerne opp på forskjellene på vektor- og rastergrafikk