Mer om funktioner och grafik i OCTAVE

1 Inledning

Först skall vi se lite på matriser och därefter på funktioner som redan finns i OCTAVE, (elementära) matematiska funktioner som sinus och cosinus samt funktioner för att bilda och operera på vektorer och matriser. Sedan ser vi lite på grafritning och annan grafik för att avslutningvis se hur man definerar egna funktioner.

2 Något om matriser

En matris är ett rektangulärt talschema

$$\mathbf{A} = \begin{bmatrix} a_{11} & \dots & a_{1n} \\ \vdots & & \vdots \\ a_{m1} & \dots & a_{mn} \end{bmatrix}$$

Matrisen ovan har m rader och n kolonner, vi säger att den är av typ $m \times n$. Ett matriselement i rad nr i, kolonn nr j tecknas a_{ij}, där i är radindex och j är kolonnindex. I OCTAVE skrivs detta `A(i,j)` och `size(A)` ger matrisens typ.

En matris av typ $m \times 1$ kallas kolonnmatris (kolonnvektor) och en matris av typ $1 \times n$ kallas radmatris (radvektor):

$$\mathbf{b} = \begin{bmatrix} b_1 \\ \vdots \\ b_m \end{bmatrix}, \quad \mathbf{c} = \begin{bmatrix} c_1 & \dots & c_n \end{bmatrix}$$

Element nr i ges i OCTAVE av `b(i), c(i)` och antalet element ges av `length(b), length(c)`. Som exempel tar vi

$$\mathbf{A} = \begin{bmatrix} 1 & 4 & 7 & 10 \\ 2 & 5 & 8 & 11 \\ 3 & 6 & 9 & 12 \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} 1 \\ 3 \\ 5 \end{bmatrix}, \quad \mathbf{c} = \begin{bmatrix} 0 & 2 & 4 & 6 & 8 \end{bmatrix}$$

Vi skriver in detta i OCTAVE enligt

```
>> A=[1 4 7 10; 2 5 8 11; 3 6 9 12]
>> b=[1; 3; 5]
>> c=[0 2 4 6 8]
```

Man använder hakparanteser (`[ ]`) för att bygga upp matriserna. Semikolon (`;`) innanför hakparanteserna betyder radbyte. Indexeringen i matriser och vektorer i OCTAVE börjar alltid på `1`, vi kan inte påverka detta. T.ex. `b(1)` är första elementet i vektorn `b`. Sista elementet i t.ex. vektorn `b` får vi med `b(end)`.

Uppgift 1. Skriv in matriserna i OCTAVE och skriv sedan ut matriselementen a_{23}, b_2, c_3. Prova `size` och `length`. Ändra a_{23} genom att skriva `A(2,3)=15`.

Ibland vill vi se en tabell som en matris. Som exempel tar vi: Värmeförlusten hos den som vistas i kyla beror inte enbart på temperaturen, utan även på hur mycket det blåser. Tabellen visar vilken *effektiv temperatur* det blir vid olika temperaturer T (°C) och vindhastigheter v (m/s).

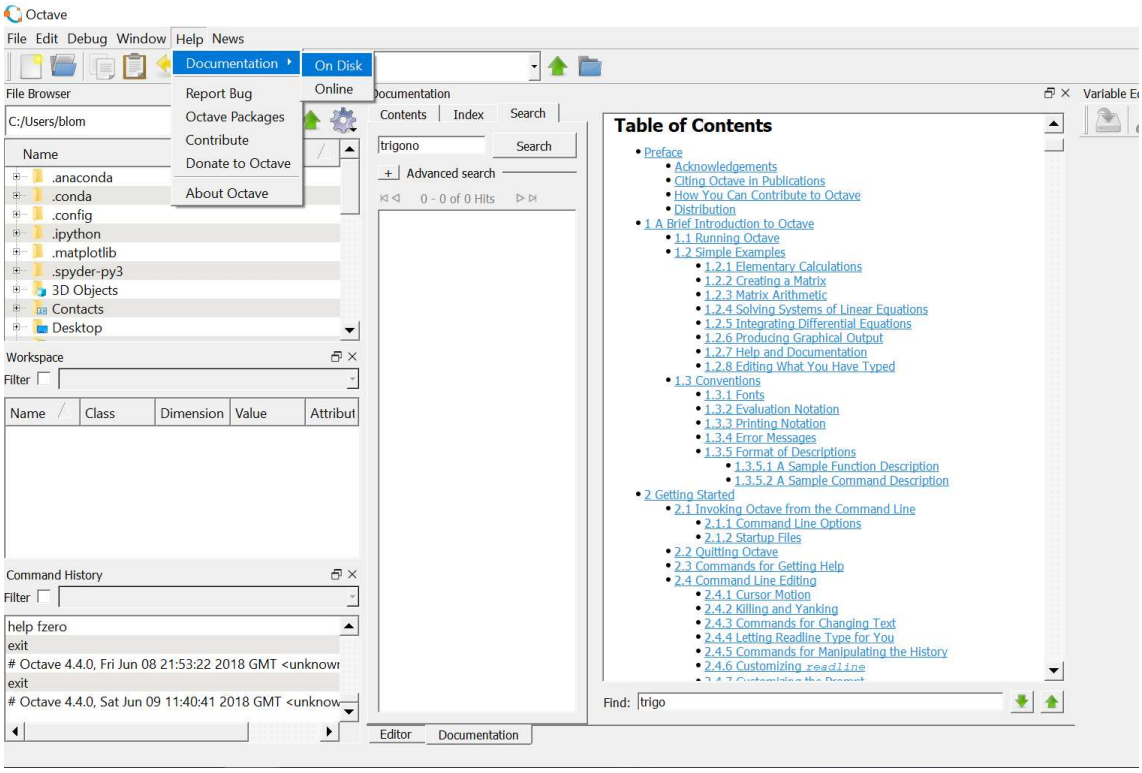
v \ T	10	6	0	-6	-10	-16	-26	-30	-36
2	9	5	-2	-9	-14	-21	-33	-37	-44
6	7	2	-5	-13	-18	-26	-38	-44	-51
10	6	1	-7	-15	-20	-28	-41	-47	-55
14	6	0	-8	-16	-22	-30	-44	-49	-57
18	5	-1	-9	-17	-23	-31	-45	-51	-59

Om vi ville göra något med dessa data i OCTAVE så skulle vi lagra temperaturer och vindhastigheter i rad- eller kolonnvektorer och effektiva temperaturerna i en matris. (Vill du läsa mer om värmeförlust gå till SMHI:s hemsida och sök på vindavkylning.)

3 Inbyggda funktioner

3.1 Matematiska funktioner

Vi letar upp några matematiska funktioner i OCTAVEs hjälptexter genom att successivt välja Documentation, On Disk från Help-menyn högst upp på desktopen. Sedan scrollar vi ner i innehållsförteckningen till kapitel 17.



Vi ser att funktionerna är grupperade, t.ex. en grupp med trigonometriska funktioner (Trigonometry) och en grupp med exponent- och logaritmfunktioner (Exponents and Logarithms).

Funktioner som exempelvis sinus och cosinus, kan operera både på enskilda tal och på matriser. Man får som resultat en matris av samma storlek, vars element är funktionsvärdet av respektive element i argumentet.

Som exempel tar vi återigen

$$\mathbf{A} = \begin{bmatrix} 1 & 4 & 7 & 10 \\ 2 & 5 & 8 & 11 \\ 3 & 6 & 9 & 12 \end{bmatrix}, \quad \mathbf{c} = \begin{bmatrix} 0 & 2 & 4 & 6 & 8 \end{bmatrix}$$

som vi skriver in i OCTAVE enligt

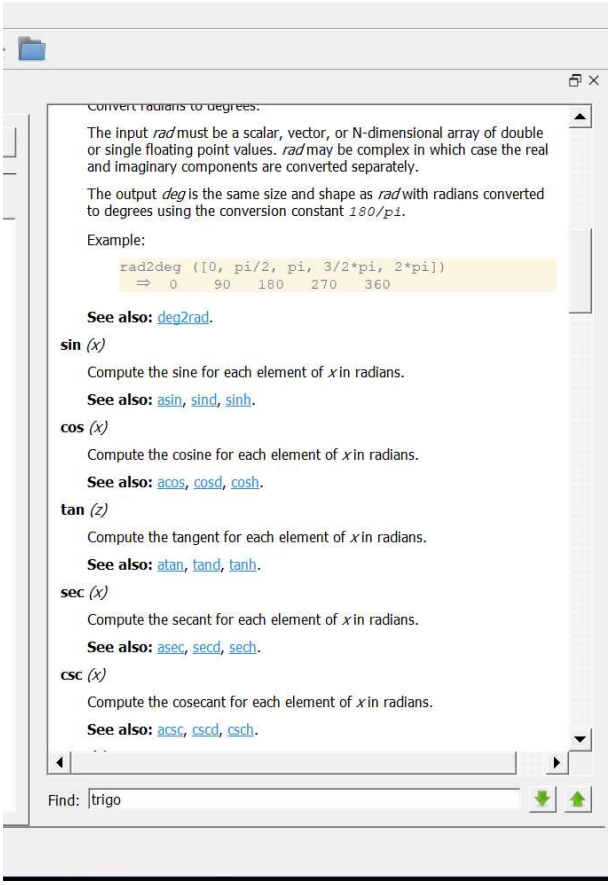
```
>> A=[1 4 7 10; 2 5 8 11; 3 6 9 12]
>> c=[0 2 4 6 8]
```

Nu beräknar vi sinus av vektorn $\mathbf{c}$ och matrisen $\mathbf{A}$ med

```
>> v=sin(c)
v =
         0    0.9093   -0.7568   -0.2794    0.9894

>> V=sin(A)
V =
    0.8415   -0.7568    0.6570   -0.5440
    0.9093   -0.9589    0.9894   -1.0000
    0.1411   -0.2794    0.4121   -0.5366
```

Vi söker upp `tan`, dvs. tangensfunktionen, genom att först välja Trigonometry.



Vi ser en allmän beskrivning av de trigonometriska funktionerna, lite längre ner på sidan följer en uppräkning av själva funktionerna. Vi scrollar ner till tan.

Uppgift 2. Leta upp hjälptexten du ser i figuren och rita upp tangensfunktionen på intervallet $-\pi/2 < x < \pi/2$. Vad händer med tangens när $x = \pi/2$?

3.2 Matris- och vektorfunktioner

Vi har redan sett de inbyggda funktionerna `length` och `size`. Som exempel använder vi vektorn **c** och matrisen **A** från förra avsnittet. Antal element i vektorn **c** ges av

```
>> l=length(c)
l =
    5
```

och antalet rader och kolonner **A** fås med

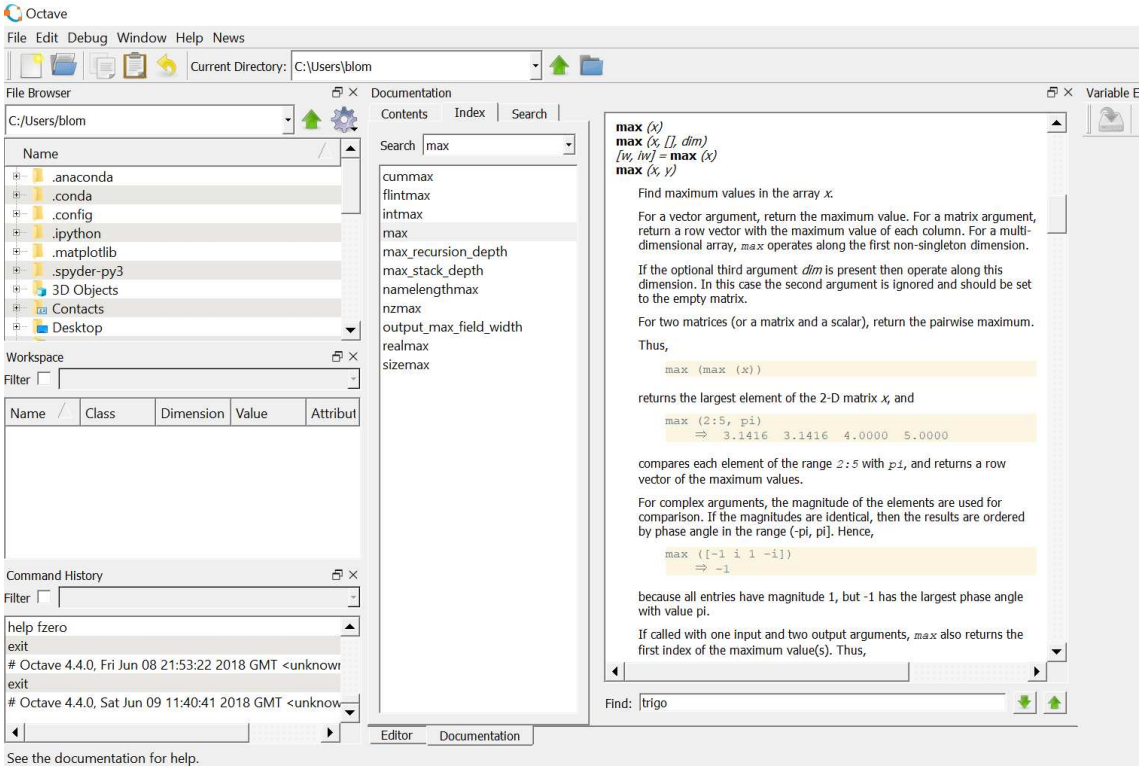
```
>> [m,n]=size(A)
m =
     3
n =
     4
```

Största och minsta elementet i en vektor fås med funktionerna `max` och `min`. För en matris blir det de största elementen i varje kolonn.

```
>> v=max(c)
v =
    8

>> v=max(A)
v =
     3     6     9    12
```

Vi ser på hjälptexten för `max` som finns i avsnittet 17.5 - Utility Functions. Observera att man kan söka på kommandonamnet i sökrutan till vänster om fönstret med informationstexten. Välj fliken Index och sök på kommandonamnet.



Vi ser att vi med `[v,index]=max(c)` även kan få reda på var det maximala värdet finns någonstans. Summan och produkten av elementen i vektorn fås med `sum` och `prod`. För en matris blir det summan eller produkten av varje kolonn.

```
>> s=sum(c)
s =
    20

>> s=sum(A)
s =
     6     15     24     33
```

Vill vi sortera en vektor i stigande ordning gör vi det med `sort`. För en matris blir det varje kolonn som sorteras om i stigande ordning.

4 Hantering av matriser

Vektorn $\mathbf{c} = (0, 2, 4, 6, 8)$ från förra avsnittet kan bildas på två sätt

```
>> c=[0 2 4 6 8]
>> c=0:2:8
```

det senare lite enklare sättet kallas *kolon-notation*.

Med kolon-notationen bygger vi upp en vektor enligt `variabel=startvärde:steg:slutvärde`

Vi låter s få tredje värdet c_3 med `s=c(3)` och bildar vektorn $\mathbf{v}$ av andra och femte värdet, dvs. $\mathbf{v} = (c_2, c_5)$, med

```
>> v=c([2,5])
v =
     2     8
```

Vi kan ändra ett element i $\mathbf{v}$, t.ex. låta $v_2 = 0$, med

```
>>  v(2)=0
v =
     2     0
```

Vi låter s få värdet av elementet på rad 2, kolonn 3 i matrisen $\mathbf{A}$ från inledningen med `s=A(2,3)` och vi bildar en radvektor $\mathbf{v}$ av rad 3, alla kolonner med

```
>> v=A(3,:)
v =
    3     6     9    12
```

samt en kolonnvektor $\mathbf{u}$ av rad 2-3, kolonn 2 med

```
>> u=A(2:3,2)
u =
    5
    6
```

Vi bildar en matris $\mathbf{V}$ av blocket rad 1-2, kolonn 2-3

```
>> V=A(1:2,2:3)
V =
    4     7
    5     8
```

Om vi har två vektorer $\mathbf{u} = (2, 3, 5)$ och $\mathbf{v} = (1, 2, 3)$ av samma typ och vill bilda summan $\mathbf{a} = \mathbf{u} + \mathbf{v}$ och skillnaden $\mathbf{b} = \mathbf{u} - \mathbf{v}$, så gör vi det med `a=u+v` respektive `b=u-v`. Operationerna sker komponentvis

$$\mathbf{a} = \mathbf{u} + \mathbf{v} = (2, 3, 5) + (1, 2, 3) = (2 + 1, 3 + 2, 5 + 3) = (3, 5, 8)$$
$$\mathbf{b} = \mathbf{u} - \mathbf{v} = (2, 3, 5) - (1, 2, 3) = (2 - 1, 3 - 2, 5 - 3) = (1, 1, 2)$$

eller med andra ord $a_i = u_i + v_i$ och $b_i = u_i - v_i$.

T.ex. vid grafritning behövs de komponentvisa motsvarigheterna till multiplikation och division

$$\mathbf{u} \texttt{ .* } \mathbf{v} = (2, 3, 5) \texttt{ .* } (1, 2, 3) = (2 \cdot 1, 3 \cdot 2, 5 \cdot 3) = (2, 6, 15)$$
$$\mathbf{u} \texttt{ ./ } \mathbf{v} = (2, 3, 5) \texttt{ ./ } (1, 2, 3) = (2/1, 3/2, 5/3) = (2, 1.5, 1.666\ldots)$$

Här har vi lånat beteckningar från OCTAVE där vi skriver `u.*v` respektive `u./v` för att utföra beräkningarna.

Vi behöver även komponentvis upphöjt till, t.ex. kvadrering

$$\mathbf{u} \texttt{ .\^{} } 2 = (2, 3, 5) \texttt{ .\^{} } 2 = (2^2, 3^2, 5^2) = (4, 9, 25)$$

Även här har vi lånat beteckningen från OCTAVE.

Det finns en funktion `linspace` som är bra för att bygga upp vektorer och funktionerna `zeros` och `ones` för att bygga upp matriser fylla med nollor respektive ettor samt funktionen `eye` för att göra enhetsmatriser.

Uppgift 3. Använd `linspace`, läs hjälptexten, för att bilda vektorn $\mathbf{d} = (2, 5, 8, 11, 14)$.

5 Grafik

5.1 Grafritning

Vi har redan ritat några grafer av funktioner $f(x)$ över ett intervall $a \leq x \leq b$. Nu ser vi på ett exempel där vi behöver komponentvisa operationer.

Exempel 1. Rita grafen till $f(x) = x\sin(x)$ över intervallet $0 \leq x \leq 8$.

Vi bildar en vektor $\mathbf{x} = (x_1, x_2, \cdots, x_n)$ med värden jämnt fördelande över intervallet $0 \leq x \leq 8$. Sedan bildar vi vektorn

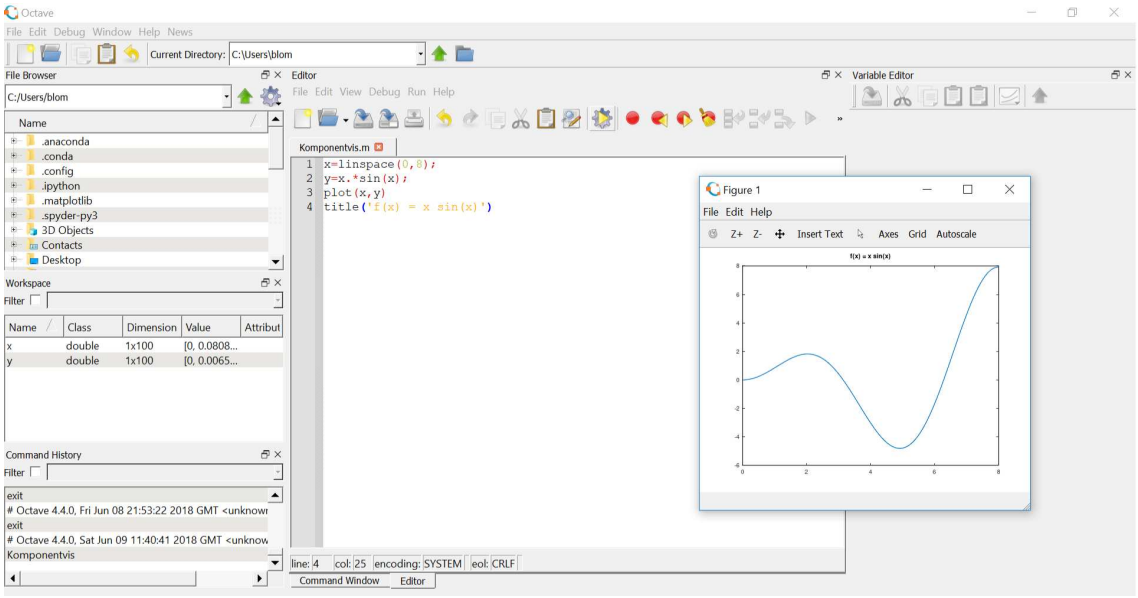
$$\mathbf{y} = (f(x_1), f(x_2), \cdots, f(x_n)) = (x_1\sin(x_1), x_2\sin(x_2), \cdots, x_n\sin(x_n)) = \mathbf{x}.*\sin(\mathbf{x})$$

och ritar upp grafen. För att bilda vektorn $\mathbf{y}$ behövs den komponentvisa multiplikationen.

Vi ritar grafen med

```
>> x=linspace(0,8);
>> y=x.*sin(x);
>> plot(x,y)
>> title('f(x) = x sin(x)')
```

och så här ser resultatet ut



Uppgift 4. Rita grafen till $f(x) = x - x\cos(7x)$ över intervallet $0 \leq x \leq 8$.

5.2 Dela figuren

Ibland vill man ha flera koordinatsystem i samma figur-fönster (Figure).

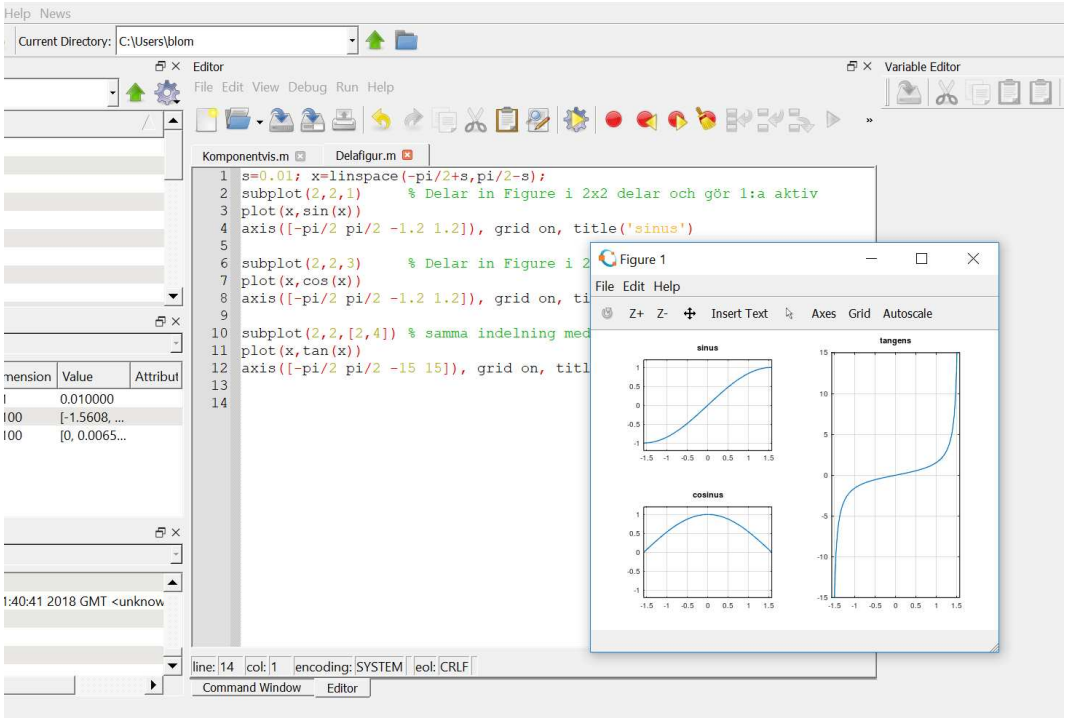
Exempel 2. Vi skall i samma figur göra tre olika koordinatsystem. I dessa skall vi rita grafen av $\sin(x)$, $\cos(x)$ respektive $\tan(x)$ över intervallet $-\frac{\pi}{2} \leq x \leq \frac{\pi}{2}$.

```
>> s=0.01; x=linspace(-pi/2+s,pi/2-s);
>> subplot(2,2,1)            % delar in Figure i 2x2 delar och gör 1:a aktiv
>> plot(x,sin(x))
>> axis([-pi/2 pi/2 -1.2 1.2]), grid on, title('sinus')
```

```
>> subplot(2,2,3)           % delar in Figure i 2x2 delar och gör 3:e aktiv
>> plot(x,cos(x))
>> axis([-pi/2 pi/2 -1.2 1.2]), grid on, title('cosinus')


>> subplot(2,2,[2,4])       % samma indelning men gör 2:a och 4:e aktiva
>> plot(x,tan(x))
>> axis([-pi/2 pi/2 -15 15]), grid on, title('tangens')
```

Så här kommer det se ut



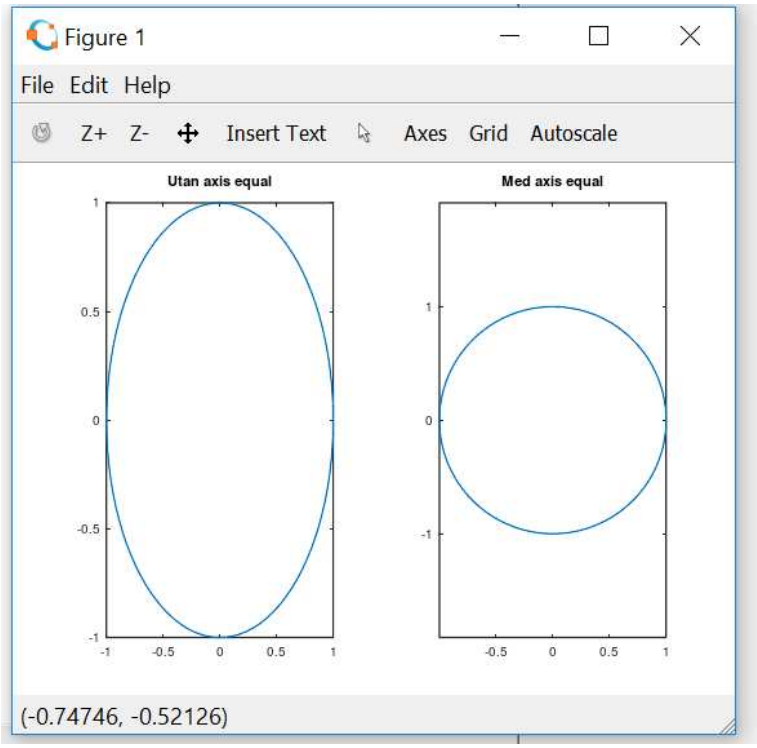
5.3 Kurvritning

Nu skall vi rita s.k. parameterframställda kurvor. Som exempel tar vi enhetscirkeln

$$t \mapsto (x(t), y(t)) = (\cos(t), \sin(t)), \quad 0 \leq t \leq 2\pi$$

När man ritar sådana kurvor ritar man inte ut parametern t utan enbart x- och y-värdena.

```
>> t=linspace(0,2*pi);
>> x=cos(t); y=sin(t);
>> subplot(1,2,1)
>> plot(x,y)
>> title('Utan axis equal')
```



För att cirkeln inte skall bli tillplattad måste vi använda `axis equal`.

```
>> subplot(1,2,2)
>> plot(x,y)
>> axis equal    % annars blir cirkeln tillplattad
>> title('Med axis equal')
```

Uppgift 5. Rita kurvorna $t \mapsto (x(t), y(t)) = (\cos(t) + \cos(3t), \sin(2t))$ och $t \mapsto (x(t), y(t)) = (\cos(t) + \cos(4t), \sin(2t))$, för $0 \leq t \leq 2\pi$. Använd `subplot`.

5.4 Polygontåg

Ett polygontåg som ges av $(x_1, y_1), (x_2, y_2), \cdots, (x_n, y_n)$, ritas upp i OCTAVE genom att man bildar vektorerna $\mathbf{x} = (x_1, x_2, \cdots, x_n)$ och $\mathbf{y} = (y_1, y_2, \cdots, y_n)$ och sedan gör `plot(x,y)`.

Grafritning är ett polygontåg vi ritar upp. Tag t.ex. grafen till $f(x) = \sin(x)$ för $0 \leq x \leq 2\pi$. Vi har då $\mathbf{x} = (x_1, x_2, \cdots, x_n)$ med $0 = x_1 < x_2 < \cdots < x_n = 2\pi$ och $\mathbf{y} = (y_1, y_2, \cdots, y_n)$ med $y_i = \sin(x_i)$. Sedan ritar vi upp med `plot(x,y)`.

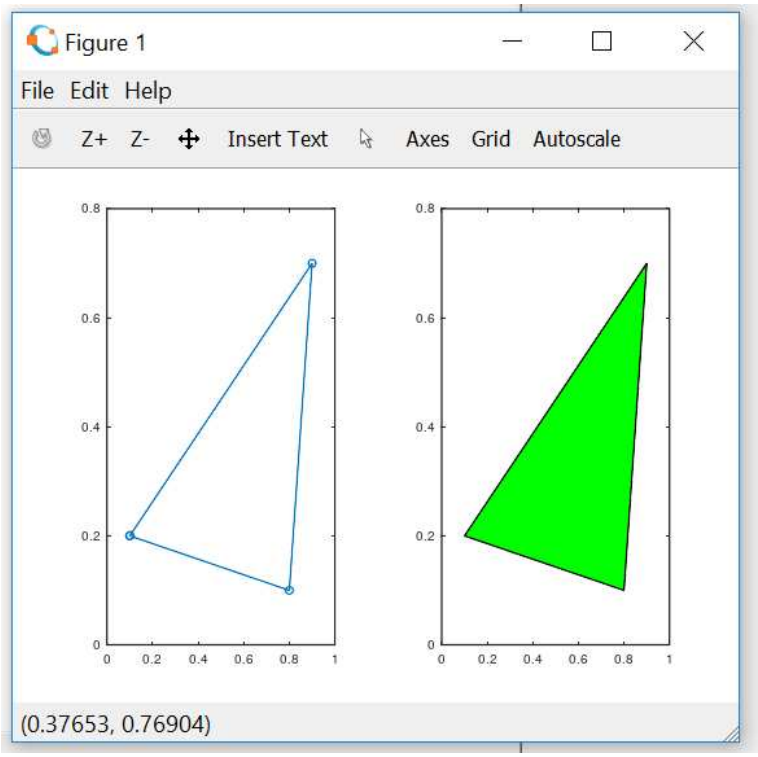
Om polygontåget är slutet, dvs. $x_n = x_1$ och $y_n = y_1$, och om det inte korsar sig självt så omsluter det ett område i planet, ett s.k. polygonområde. Vi kan använda `fill` för att färglägga ett sådant område.

Vi ritar upp polygontåg som ges av $(0.1, 0.2), (0.8, 0.1), (0.9, 0.7), (0.1, 0.2)$, dvs. en triangel.

```
>> x=[0.1 0.8 0.9 0.1];
>> y=[0.2 0.1 0.7 0.2];
>> subplot(1,2,1)
>> plot(x,y,'-o')
>> axis([0 1 0 0.8])
```

Med `'-o'` anger vi att punkterna både skall förbindas med räta linjer och markeras med små ringar. Vi fyller området med grön färg och vi använder `axis` för att få lite "luft" runt triangeln.

```
>> subplot(1,2,2)
>> fill(x,y,'g')
>> axis([0 1 0 0.8])
```



Uppgift 6. Rita en cirkel fylld med grön färg, rita sedan en kvadrat inskriven i cirkeln och fyll kvadraten med gul färg. Använd `hold on`.

6 Egna funktioner

Exempel 3. Vi skall rita grafen av

$$f(x) = \frac{\sin(ax)}{x}$$

över intervallet $-5 \leq x \leq 5$, för $a = 1, 2,$ och 3.

Vi måste tänka på att $f(x)$ inte definierad i $x = 0$. Finns gränsvärdet och vad blir det i så fall?

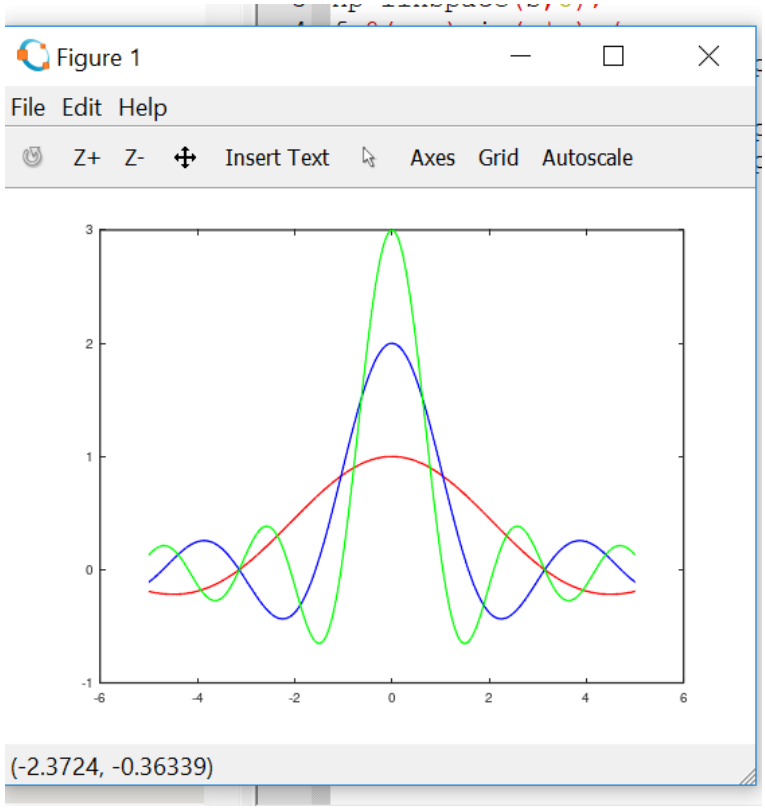
```
>> s=0.01;              % s för att separera nollan
>> xn=linspace(-5,-s);  % xn negativa x-värden
>> xp=linspace(s,5);    % xp positiva x-värden
```

Nu behöver vi komponentvis division och vi inför ett funktionshandtag (function handle). Vi ser först på fallet $a = 1$

```
>> f=@(x)sin(x)./x;
>> plot(xn,f(xn),'r',xp,f(xp),'r')
```

Nu skall vi ta fallen $a = 2$ och 3 också. Enklast är om vi gör om vårt funktionshandtag så att även a blir ett argument.

```
>> f=@(x,a)sin(a*x)./x;
>> plot(xn,f(xn,1),'r',xp,f(xp,1),'r')  % a=1
>> hold on
>> plot(xn,f(xn,2),'b',xp,f(xp,2),'b')  % a=2
>> plot(xn,f(xn,3),'g',xp,f(xp,3),'g')  % a=3
>> hold off
```



I figuren ser det ut som $f(x) \to a$ då $x \to 0$. Tänk dock på att en figur inte är något bevis!

Vi använde en anonym funktion (anonymous function) med ett funktionshandtag (function handle) enligt

```
handtagsnamn = @(parametrar) sats
```

Här är delen `@(parametrar) sats` den anonyma funktionen och `handtagsnamn` är det namn vi väljer på funktionshandtaget som kopplas till funktionen. Med `parametrar` avser vi indata till funktionen, ofta en variabel ibland flera.

I denna konstruktion är det bara tillåtet med en enda beräkningssats. En mer komplicerad funktion (som kan bestå av flera beräkningssatser) kräver att vi definierar en funktion (function).

Exempel 4. En kastbana utan luftmotstånd beskrivs av

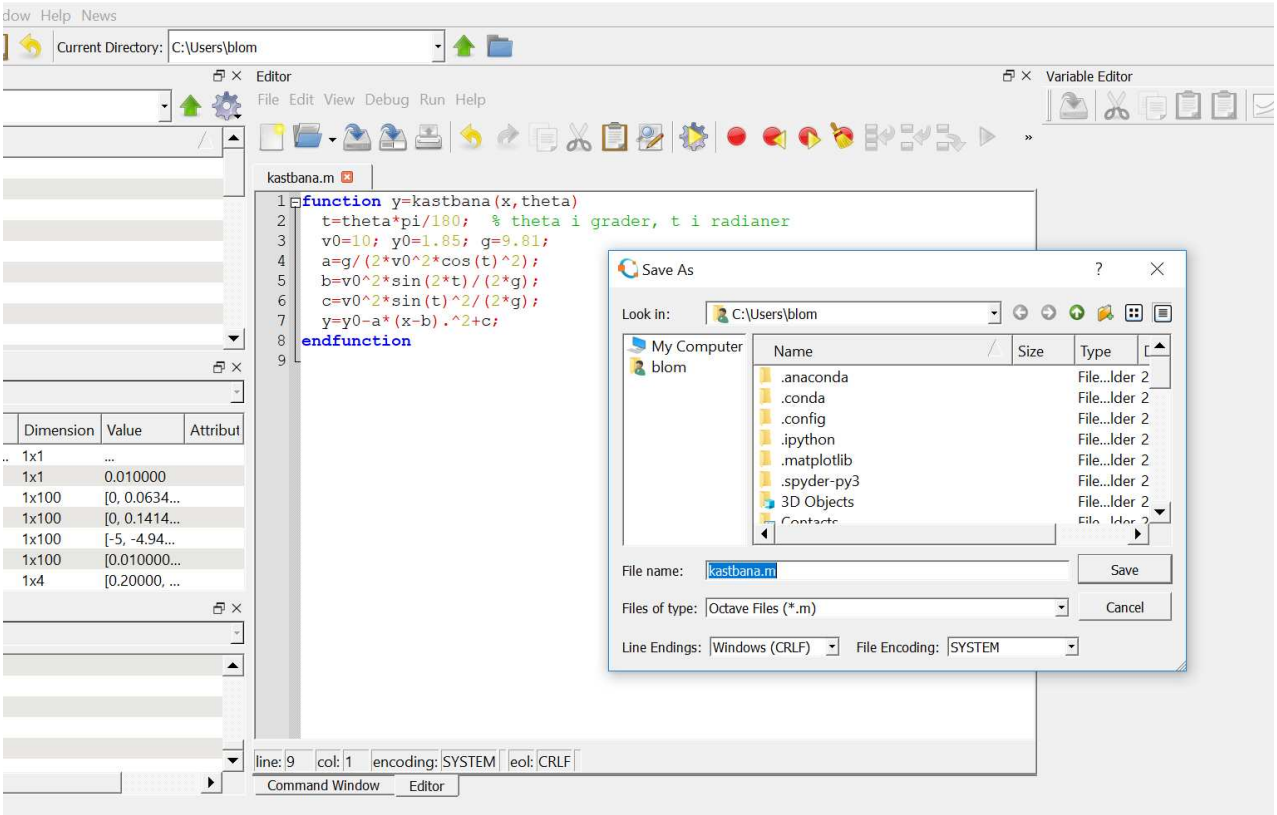
$$y(x) = y_0 - \frac{g}{2v_0^2\cos^2(\theta)}\left(x - \frac{v_0^2\sin(2\theta)}{2g}\right)^2 + \frac{v_0^2\sin^2(\theta)}{2g}$$

där v_0 är utkastfarten, y_0 är utkasthöjden, θ är utkastvinkeln och g är tyngdaccelerationen.

Först gör vi en function med namnet `kastbana` som beskriver kastbanan för olika utkastvinklar.

```
function y=kastbana(x,theta)
    t=theta*pi/180;     % theta i grader, t i radianer
    v0=10; y0=1.85; g=9.81;
    a=g/(2*v0^2*cos(t)^2);
    b=v0^2*sin(2*t)/(2*g);
    c=v0^2*sin(t)^2/(2*g);
    y=y0-a*(x-b).^2+c;
```

Vi skriver in i editorn.

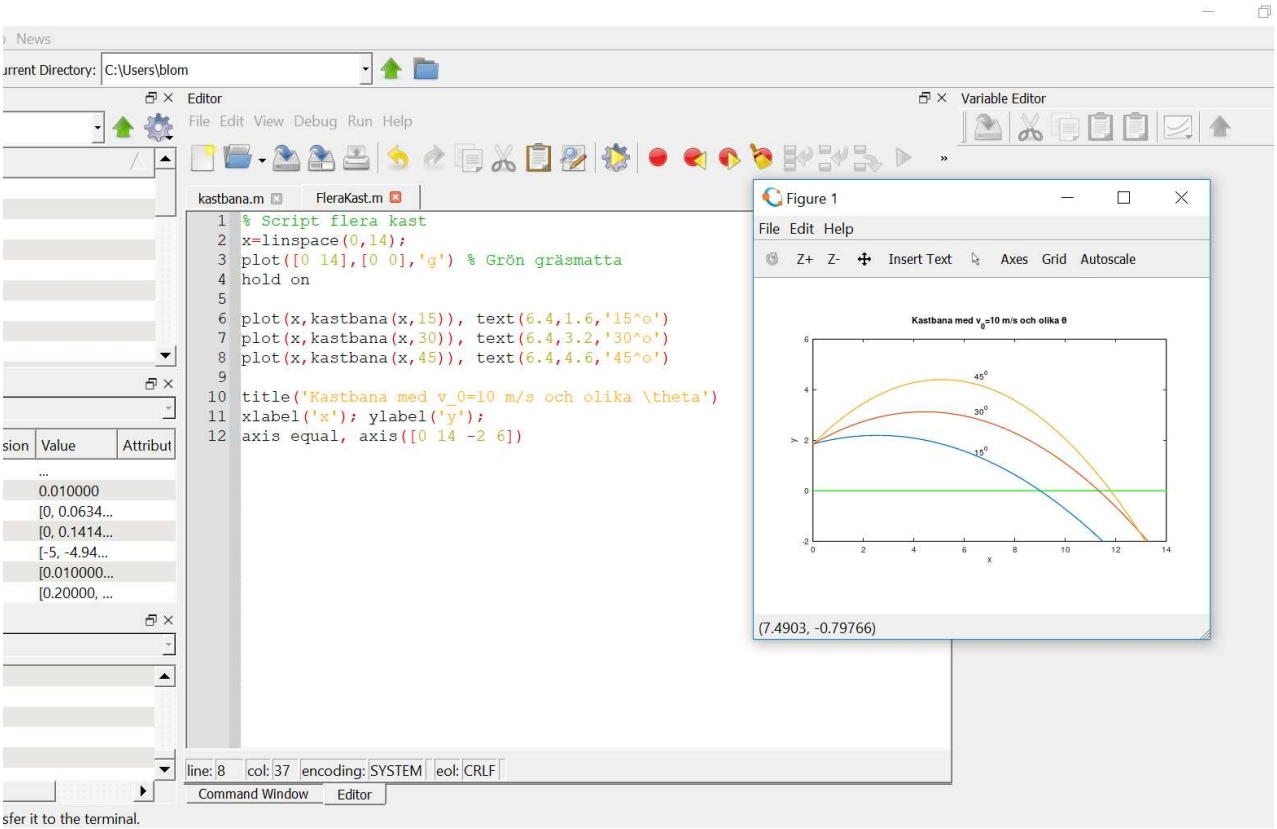


En funktion (function) är alltså en textfil med följande struktur

```
function ut = funktionsnamn(parametrar)
    satser
```

Här är `funktionsnamn` det namn vi ger funktionen och `funktionsnamn.m` är det namn vi ger textfilen där programkoden lagras. Med `parametrar` avser vi indata till funktionen, ofta en variabel ibland flera. Funktionen måste innehålla en sats där `ut`, som står för utdata eller funktionsvärdet, tilldelas ett värde.

Vi gör sedan ett script där vi tar $v_0 = 10$ m/s, $y_0 = 1.85$ m och ritar kastbanorna för några olika utkastvinklar. Så här ser det ut när vi ritat graferna. Vi har också placerat ut lite förklarande text vid graferna med kommandot `text`.



Uppgift 7. Skriv den function och det script för kastbanan som vi pratar om i exemplet. Rita graferna. Varför delar vi upp funktionsuttrycket för $y(x)$ i flera delar?