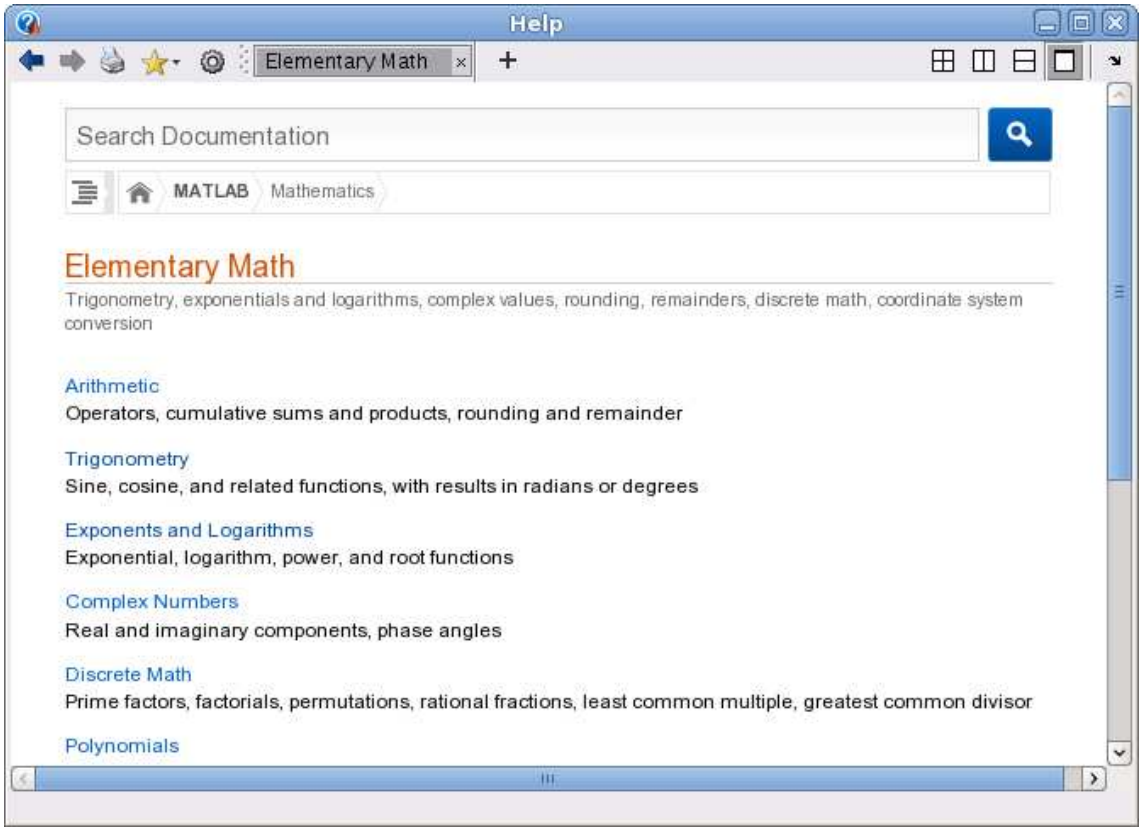
Funktioner och grafritning i MATLAB

1 Inledning

Först skall vi se lite på (elementära) matematiska funktioner i MATLAB, som sinus och cosinus. Sedan ser vi på grafritning och hur vi definerar egna funktioner. Avslutningsvis ser vi lite på ritning av allmännare kurvor.

2 Elementära funktioner

Vi letar upp hjäptexterna för elementära eller matematiska funktioner i Help genom att successivt öppna MATLAB, Mathematics och sedan Elementary Math.



Vi ser att funktionerna är grupperade, t.ex. en grupp med trigonometriska funktioner och en grupp med exponent- och logaritmfunktioner.

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 radmatrisen (radvektorn) $\mathbf{x} = (0, 0.1, 0.2, 0.3, 0.4, 0.5)$ som vi skriver in i MATLAB enligt

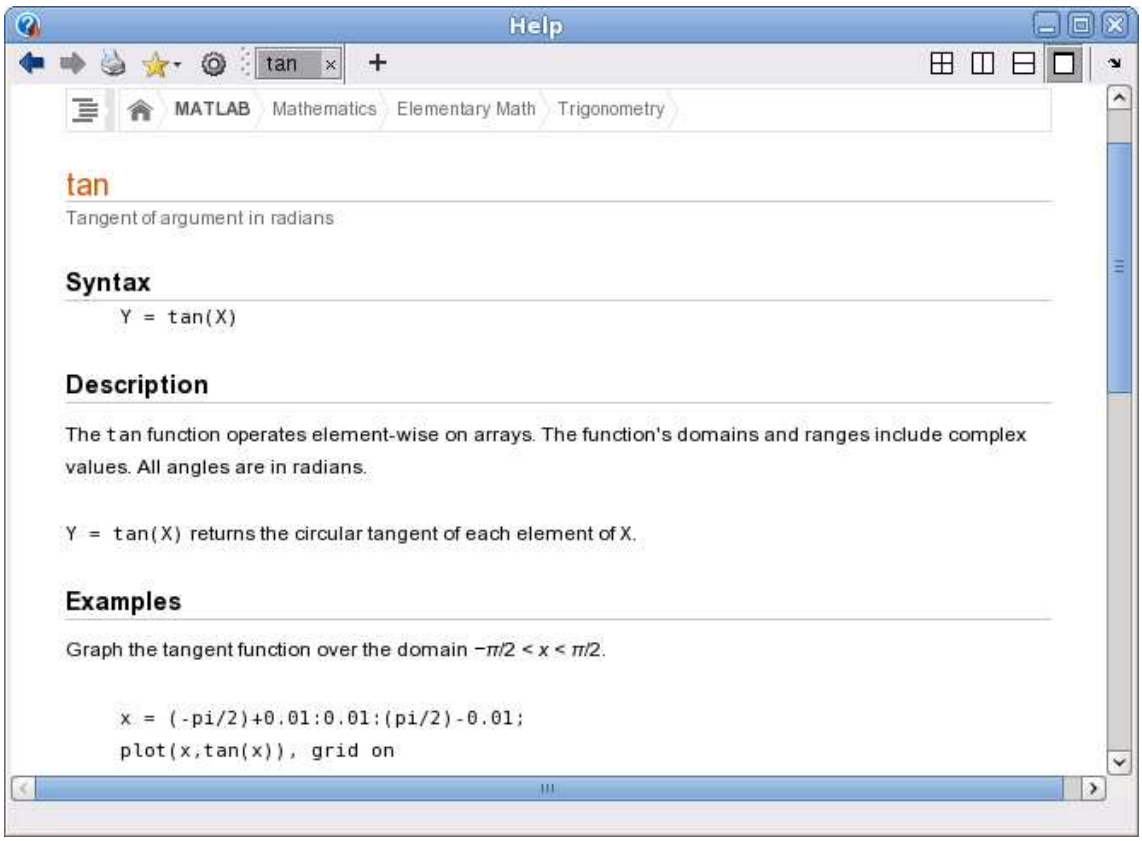
```
>> x=0:0.1:0.5
x =
         0    0.1000    0.2000    0.3000    0.4000    0.5000
```

Nu beräknar vi **y** som är sinus av radvektorn **x** med

```
>> y=sin(x)
y =
         0    0.0998    0.1987    0.2955    0.3894    0.4794
```

Här blir `sin(x)` en radvektor eftersom `x` var en radvektor.

Vi skall se på ytterligare en funktion, tangens, som ju är kvoten mellan sinus och cosinus. Den heter `tan` i MATLAB och vi söker upp dess hjälptext.



Uppgift 1. Leta upp hjälptexten du ser i figuren och rita upp tangensfunktionen enligt exemplet (i hjälptexten). Repetera hur du startar hjälpverktyget (laboration 1, avsnitt 7), om du har glömt. Varför ritar man grafen över intervallet $-\frac{\pi}{2} + s \leq x \leq \frac{\pi}{2} - s$, där s är ett litet positivt tal?

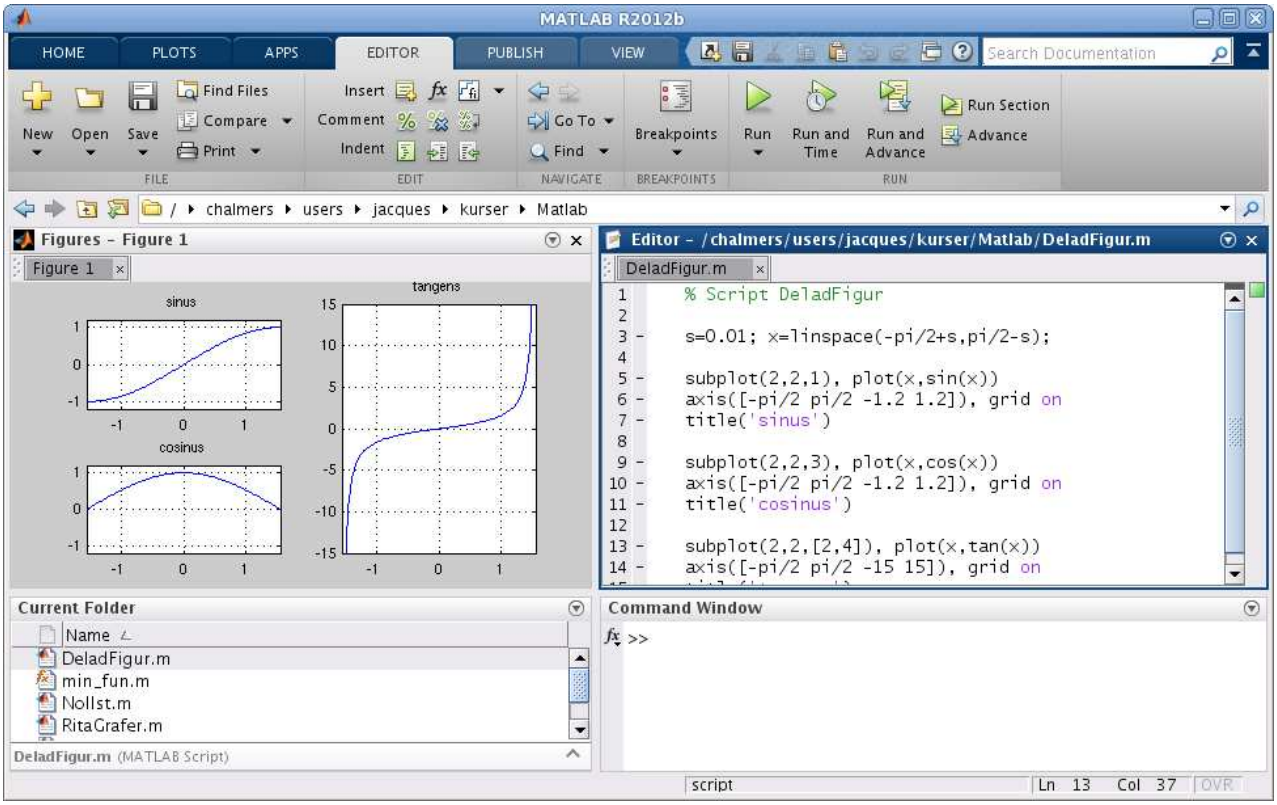
3 Grafer

Ibland vill man rita flera grafer i samma koordinatsystem. Efter att ha ritat första grafen ger man kommandot `hold on` för att bevara den, sedan kan man rita fler grafer ovanpå tills man tar bort skyddet med `hold off`. Vi påminner oss att vi kan lägga på ett rutnät med `grid on` och ta bort det igen med `grid off`, om vi vill det. Med `xlabel` och `ylabel` kan vi sätta texter på axlarna och med `title` kan vi sätta rubrik på koordinatsystemet. Allt detta har vi redan gjort, kommer du inte ihåg det är det kanske läge att kort repetera (laboration 1, avsnitt 3).

Ibland vill man ha flera koordinatsystem i samma figur-fönster (Figure). Då använder man kommandot `subplot`. Vi ser på ett exempel.

Exempel 1. Vi skall i samma figur göra tre olika koordinatsystem. I dessa skall vi rita graferna av $\sin(x)$, $\cos(x)$ respektive $\tan(x)$ över intervallet $-\frac{\pi}{2} \leq x \leq \frac{\pi}{2}$.

Så här kommer det se ut



Vi ser lite på programkoden

```
>> 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')
```

Den första 2:an i `subplot` förbereder för två rader av koordinatsystem och den andra förbereder för två kolonner av koordinatsystem. Dessa numreras stigande vänster till höger, uppifrån och nedåt. Vi anger att det 1:a systemet skall vara aktivt och där hamnar grafen av 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')
```

I det 3:e systemet ritade vi grafen av cosinus. Eftersom grafen av tangens behöver få sträcka sig ganska mycket vertikalt, fogar vi samman det 2:a och 4:e systemet, genom att bilda vektorn `[2,4]`, och där ritar vi sedan grafen.

Kommandot `axis` använder vi när vi inte nöjer oss med de skalor på axlarna som vi får automatiskt. För t.ex. tangens vill vi ha intervallet $-\frac{\pi}{2} \leq x \leq \frac{\pi}{2}$ horisontellt och vertikalt blir intervallet $-15 \leq y \leq 15$ rätt lagom. Vi har alltså vertikalt skurit bort en bra bit av grafen för att få en snygg bild. Mer om `axis` i uppföljningen.

Exempel 2. 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

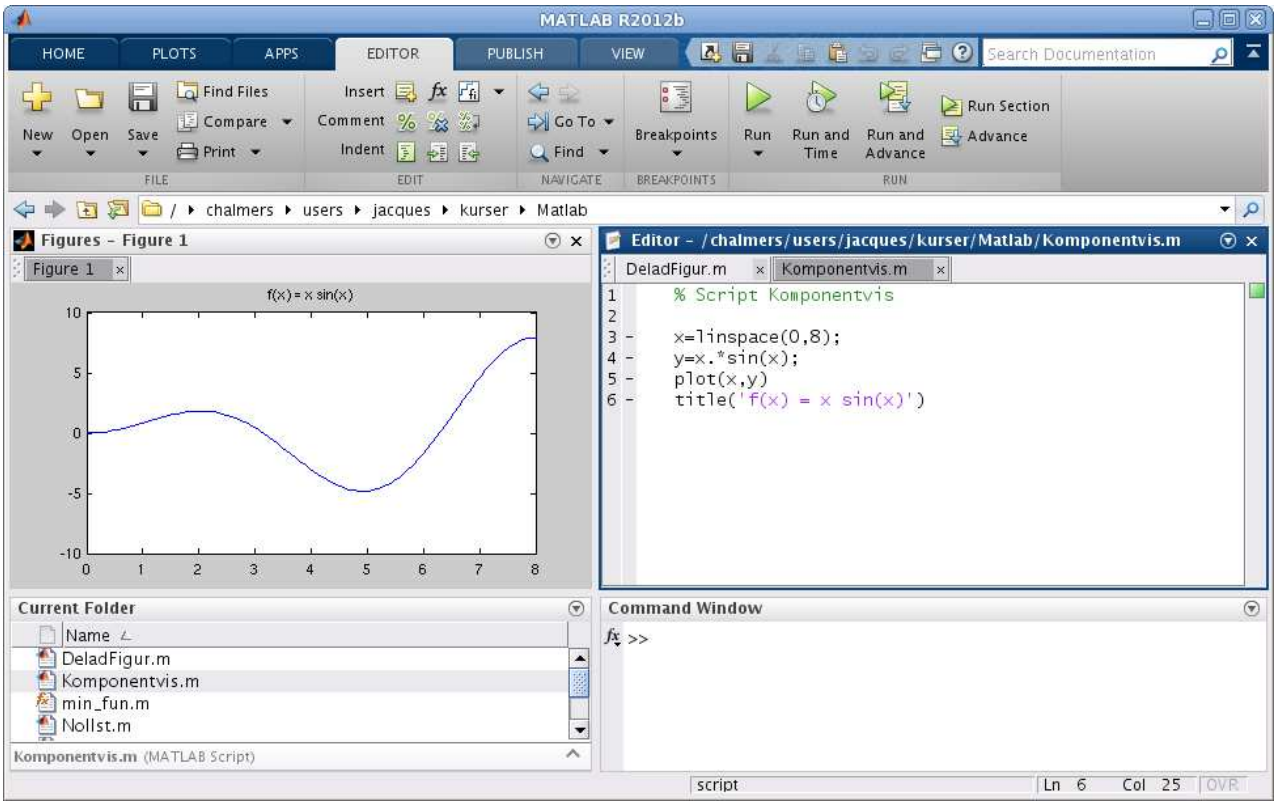
$$\mathbf{y} = (f(x_1), f(x_2), \cdots, f(x_n)) = (x_1\sin(x_1), x_2\sin(x_2), \cdots, x_n\sin(x_n))$$

och ritar upp grafen. För att bilda vektorn $\mathbf{y}$ behövs den komponentvisa multiplikationen, vi vill ju att $y_i = f(x_i) = x_i\sin(x_i)$ för alla $i = 1, 2, \cdots, n$. Har du glömt komponentvisa operationer så repetera lite (uppföljning av laboration 1).

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

4 Egna funktioner

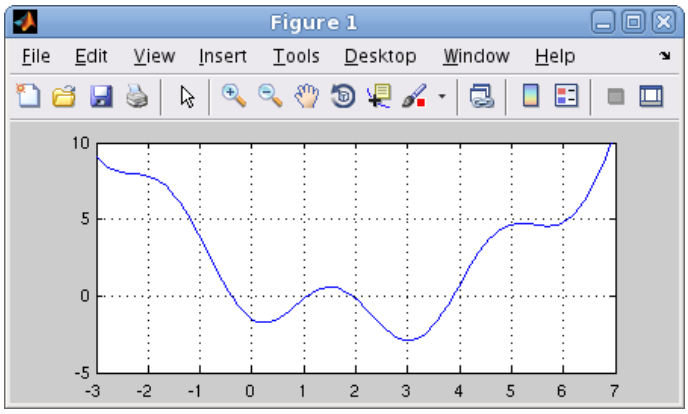
I senare laborationer skall vi se på beräkningsmetoder för att lösa ekvationer av typen $f(x) = 0$, dvs. söka nollställen till en funktion f.

Som exempel kan vi ta

$$f(x) = 0.5\,(x-2)^2 - 2\,\cos(2x) - 1.5 = 0$$

Det vi alltid kommer börja med är att rita grafen till f för att få en uppfattning om hur många nollställen vi har och ungefär var de ligger.

```
>> f=@(x)0.5*(x-2).^2-2*cos(2*x)-1.5;
>> x=linspace(-3,7);
>> plot(x,f(x))
>> axis([-3 7 -5 10]), grid on
```



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 men 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), som vi nu skall ge ett exempel på.

Exempel 3. 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.

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

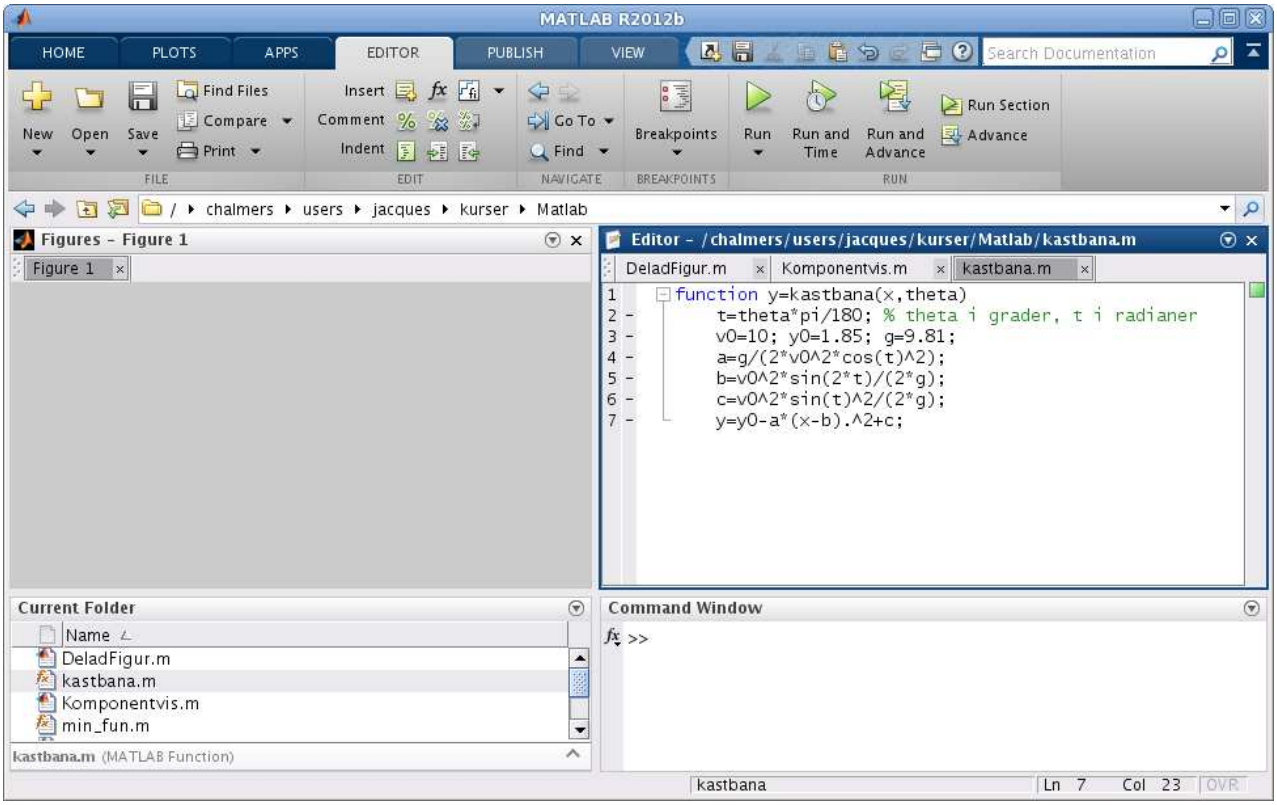
```
function y=kastbana(x,theta)
    t=theta*pi/180;     % theta i grader, t i radianer
    v0=10; y0=1.85; g=9.81;
```

```
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;
```

Första raden inleds med `function`, för att tala om att det just är en funktion vi beskriver, och `kastbana` är namnet på funktionen.

Funktionens värde kommer ges till variabeln `y` (utdata) och funktionens argument (indata) är `x`, så klart, samt utkastvinkeln `theta` (lämpligt i vårt fall då vi skall rita flera grafer). Lägg märke till omvandlingen från grader till radianer.

Vi skriver in funktionen i editorn och `kastbana.m` ges som namn till textfilen.



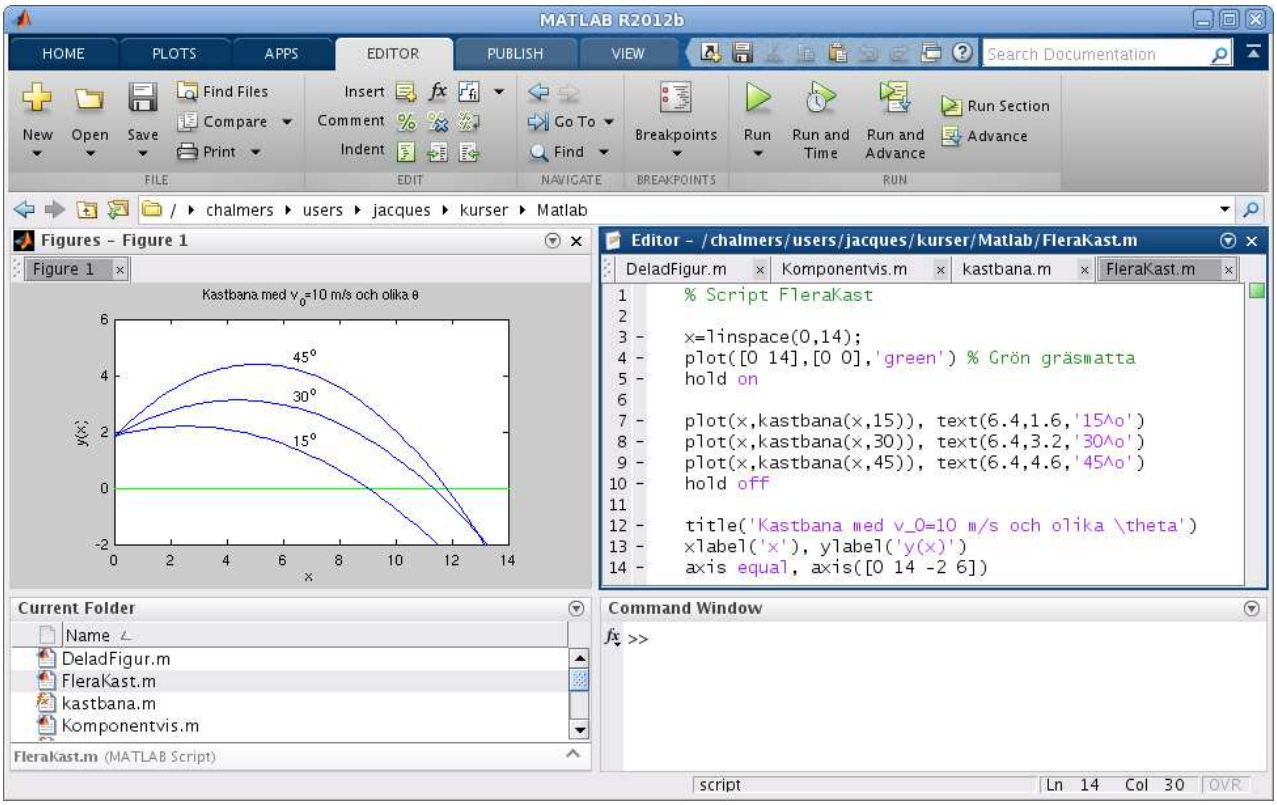
En funktion (function) är alltså en textfil med följande struktur

```
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.

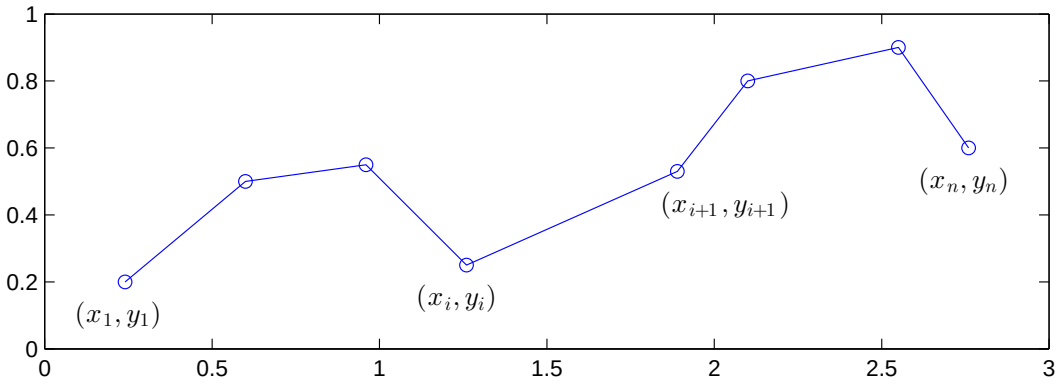
Så här ser det ut när vi ritat graferna. Vi har även placerat ut lite förklarande text vid graferna med kommandot `text`.



Uppgift 3. 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?

5 Kurvritning

Ett polygontåg är en följ av punkter $(x_1, y_1), (x_2, y_2), \cdots, (x_n, y_n)$, som vi successivt förbinder med räta linjer.



Polygontåget kan ritas upp i MATLAB 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 ger kommandot `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)`.

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

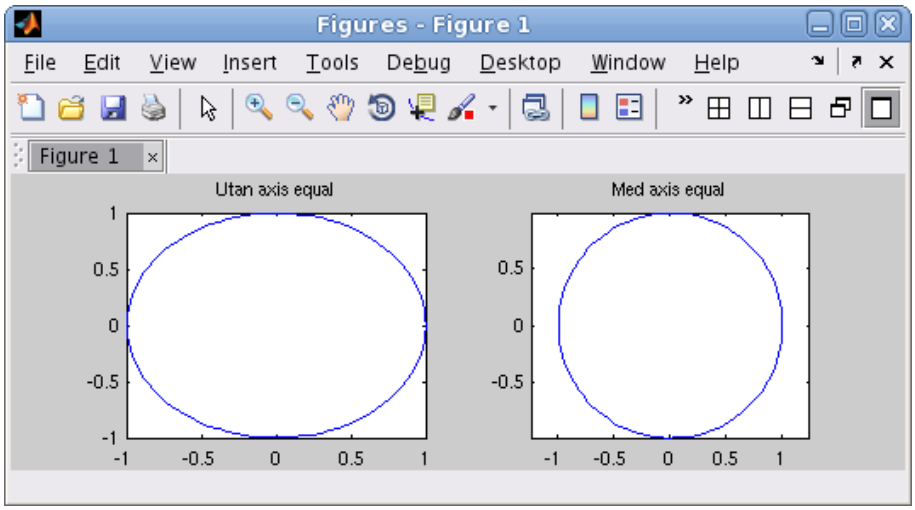
$$(x(t), y(t)) = (\cos(t), \sin(t)), \quad 0 \leq t \leq 2\pi$$

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

Vi bildar polygontåget och ritar upp enligt

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

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



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

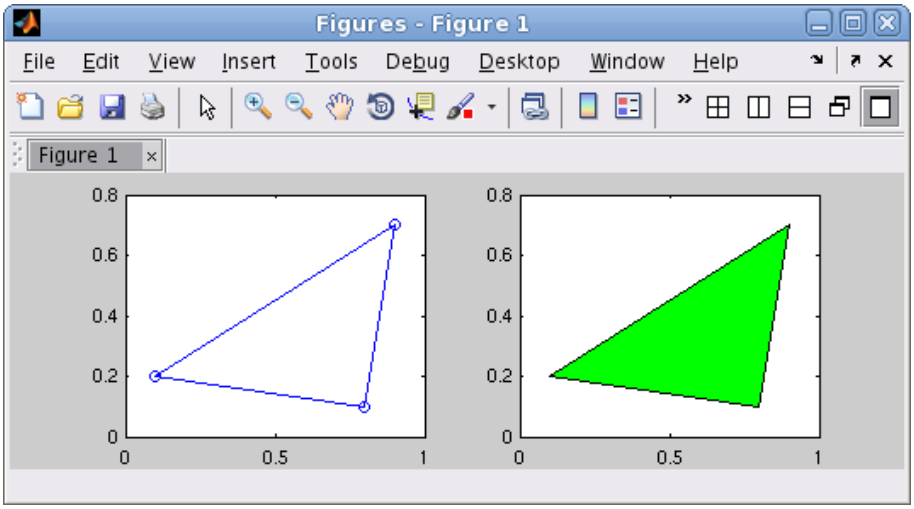
Om polygontåget är slutet, dvs. $x_n = x_1$ och $y_n = y_1$, och om det inte korsar sig självt så omsluter det ett område i planet, ett s.k. polygonområde. Trianglar, rektanglar och cirkelskivor är exempel på polygonområden.

Vi ritar upp den triangel som ges av punkterna $(0.1, 0.2), (0.8, 0.1), (0.9, 0.7), (0.1, 0.2)$.

```
>> 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 använder `axis` för att få lite "luft" runt triangeln.

Resultatet ser vi här nedan till vänster.



Vi kan använda `fill` för att färglägga ett polygonområde.

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

Vi fyllde triangeln med grön färg. Resultatet ser vi ovan till höger.

Uppgift 5. Rita en cirkel fylld med grön färg, rita sedan en kvadrat inskriven i cirkeln och fyll kvadraten med gul färg. Använd `hold on`.

1 Målsättning

Avsikten med laborationen är att lära sig att leta upp vad de vanliga elementära funktionerna, som sinus och liknade, heter i MATLAB och hur de används. Vidare skall vi stärka färdigheter i att göra snygga och ändamålsenliga grafer. Viktigast är dock att öka kunnadet om att skapa och använda egendefinierade funktioner (function). Avslutningsvis vill vi lyfta grafritning till något allmännare, nämligen kurvritning, t.ex. rita en cirkel.

2 Kommentarer och förklaringar

Här följer kommentarer till några avsnitt i laborationstexten.

I avsnitt 3 ”Grafer” försöker vi höja nivån på grafritningen och då får vi jobba lite mer. Se på exempel 1, där vi använder `subplot` för att sätta ihop en balanserad bild av tre grafer som har koppling till varandra. Sinus och cosinus är ju nära släkt och tangens är ju kvoten mellan dem. I bilden är det naturligt att sinus och cosinus får lika stora koordinatsystem, medan tangens som har ett stort vertikalt spann får ett ”högre” koordinatsystem. Vi skär av tangensgrafen med `axis` genom att ge gränserna -15 och 15 i y-led.

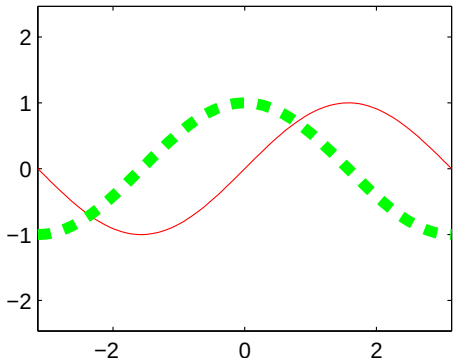
Vi väljer skalor i aktuellt koordinatsystem med

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

där vi ger en vektor till `axis` med gränserna horisontellt och vertikalt.

Vi kan också ändra tjocklek och stil på linjerna i en graf. Som exempel tar vi: Rita graferna av sinus och cosinus över intervallet $-\pi \leq x \leq \pi$. Sinusgrafen skall vara röd heldragen och cosinusgrafen skall vara grön streckad och lite tjockare.

```
>> x=linspace(-pi,pi);
>> plot(x,sin(x),'r')
>> hold on
>> plot(x,cos(x),'--g','linewidth',5)
>> hold off
>> axis equal
```



Avsnitt 4 ”Egna funktioner” får man repetera flera gångar, gradvis kommer vi vänja oss och börja förstå. Arbeta med det, ge det tid! Försök tänka efter vad ni gör, undvik ”Copy and Paste”, det finns inga magiska genvägar.

I avsnitt 5 ”Kurvritning” ser vi en annan användning av `axis`. När vi vill ha ett koordinatsystem där t.ex. en cirkel inte ser tillplattad ut ger vi `axis equal`, vill vi återställa till standard ger vi `axis normal`. Även här valde vi linjestil när vi ritade den första triangeln, `'-o'` anger att punkterna skall markeras med ringar och förbindas med heldragna linjer och att färgen skall vara standard blå (eftersom vi inte angav något annat). Vill vi ha samma linjestil men med gröna linjer ger vi `'-go'`.

Linjetyper: `'-'`(heldragen), `'--'`(streckad), `':'`(prickad), `'-.'`(prick-streckad)

Standardfärger: `'r'`(röd), `'g'`(grön), `'b'`(blå), `'c'`(cyan), `'m'`(magenta), `'y'`(gul), `'k'`(svart), `'w'`(vit)

Markörer: `'+'`(plus), `'o'`(ring), `'*'`(stjärna), `'x'`(kryss), `'s'`(kvadrat), `'d'`(diamant)

3 Fördjupning

Gå till Help och sök efter `linespec`. Där ser ni att vi kan, förutom de fördefinierade färgerna, även välja att använda egna färger som vi anger med RGB-tripplar. Ni ser vidare vilka linjestilar som finns och vilka markörer för punkter vi kan välja på. Ni kan också se hur vi kan välja storlek på markörer (`'markersize'`), deras färg (`'markerfacecolor'`) och kantfärg (`'markeredgecolor'`).

4 Lärandemål

Efter denna laboration skall du i MATLAB

- kunna leta upp vad elementära funktioner heter och hur de används
- kunna rita grafer och välja lämpliga skalor på axlar med `axis`
- kunna skapa flera koordinatsystem i ett figurfönster med `subplot`
- kunna skapa och använda egna funktioner
- kunna rita polygontåg och allmänna kurvor