Introduktion till MATLAB

Analys och Linjär Algebra, del A, K1/Kf1/Bt1

1 Inledning

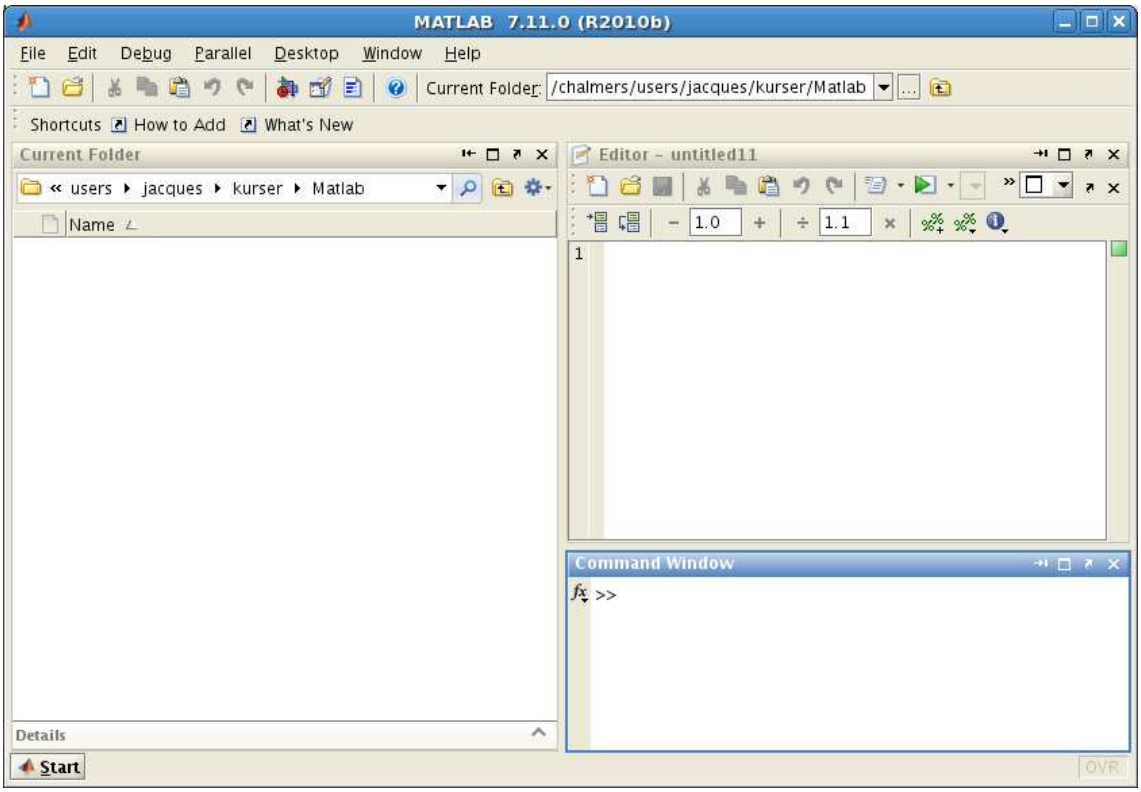
MATLAB är både en interaktiv matematikmiljö och ett programspråk, som används på de flesta tekniska högskolor runt om i världen, och har stor användning även inom industrin.

En av styrkorna med MATLAB är att systemet är utbyggbart med bibliotek eller verktygslådor, toolboxes, för olika tillämpningsområden.

Ni kommer använda MATLAB i nästan alla kurser i utbildningen. I matematikkurserna kommer vi ha studio-övningar varje vecka under läsperioderna 1, 2 och 3. Vi kommer även, i läsperiod 2 och 3, göra projektuppgifter tillsammans med kemikursen som går samtidigt.

2 Starta MATLAB

Sitter man vid en WINDOWS-dator så startar man MATLAB genom att gå in under Start-symbolen och välja All Programs och därunder välja MATLAB.



Man avslutar MATLAB genom att gå in under File och välja Exit MATLAB (längst ned i menyn).

MATLAB-fönstret man får upp kallas Desktop och dess utseende eller uppdelning kallas Desktop Layout. Den standard Desktop Layout man får då man startar MATLAB första gången ser lite annorlunda ut än på bilden.

Vi kommer under nästa studio-övning att göra en layout som ser ut ungefär som på bilden och som är lämplig för det fortsatta arbetet med MATLAB i matematikkurserna.

3 En enkel beräkning och några grafer

Här följer några exempel så att vi snabbt kommer igång och ser lite resultat. Följ gärna med vid datorn och knappa in efter hand i Command Window och se vad som händer.

Exempel 1. Beräkna volymen av ett klot med radien $r = 3$ cm. Volymen ges av $V = \frac{4}{3}\pi r^3$.

Först inför vi en variabel `r`, för radien, som vi ger värdet 3.

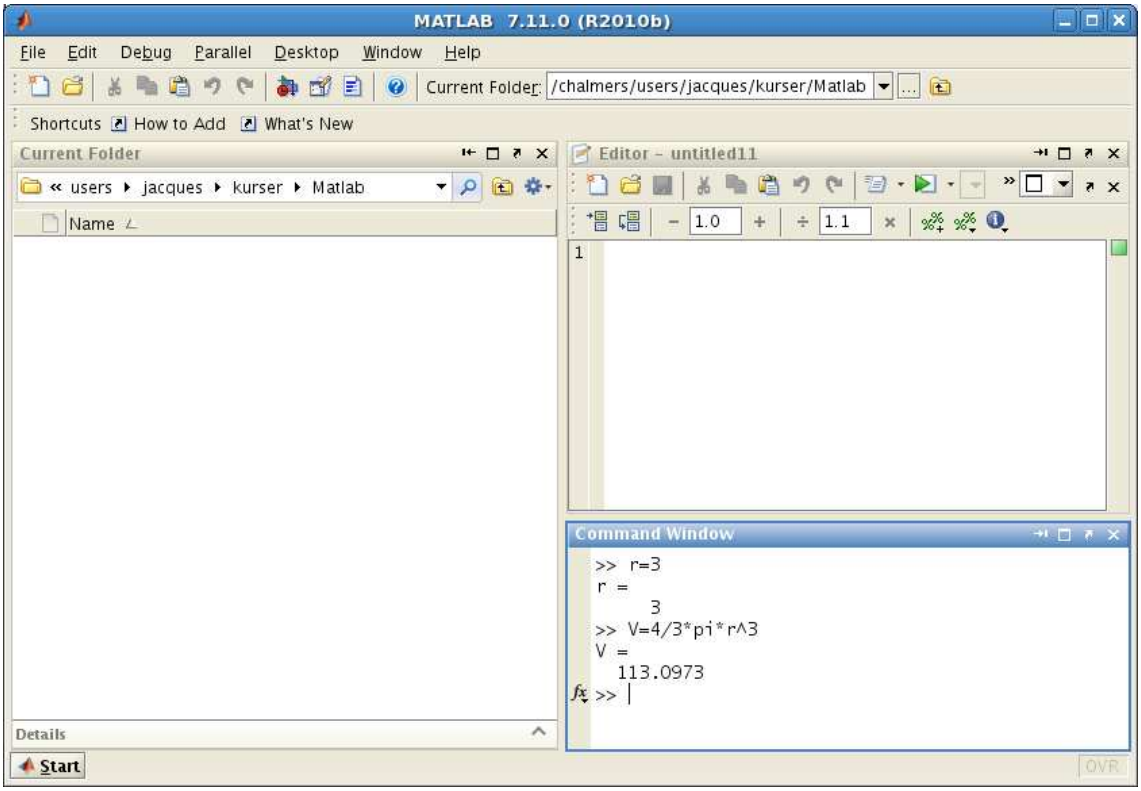
```
>> r=3
```

Den s.k. promptern (`>>`) i Command Window visar att MATLAB är redo. Ett variabelnamn skall börja med en bokstav (`a-z`, `A-Z`), därefter får vi ha bokstäver (`a-z`, `A-Z`), siffror (`0-9`) och understrykningstecken (`_`). Vidare skiljer MATLAB på stora och små bokstäver.

Därefter beräknar vi volymen enligt formeln och låter variabeln `V` få detta värde.

```
>> V=4/3*pi*r^3
```

Konstanten `pi` är en approximation av den matematiska konstanten π.



Uppgift 1. Beräkna arean av en cirkelskiva med radien $r = 4$ cm. Arean ges av $A = \pi r^2$.

Exempel 2. Rita grafen av $f(x) = \sin(x) + 0.3\,\sin(4x)$ för $0 \leq x \leq 4\pi$.

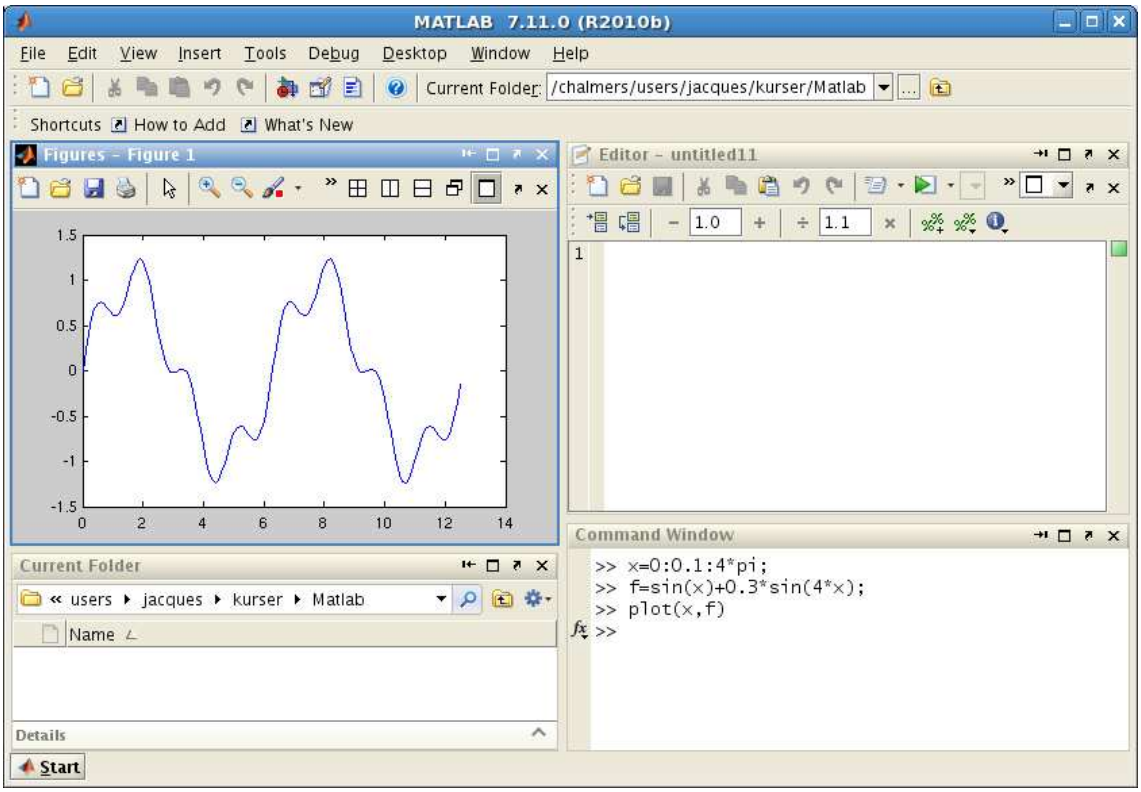
Först gör vi en lista eller radvektor `x` av x-värden mellan 0 och 4π, med

```
>> x=0:0.1:4*pi;
```

Närmare bestämt får vi värdena 0, 0.1, 0.2, 0.3, $\cdots$, 12.5, dvs. värden med start i 0, steget 0.1 och slut så nära upp mot 4π som möjligt.

Därefter gör vi en lista eller radvektor `f` med $f(x)$-värden för varje x-värde i `x` och ritar upp grafen med `plot`.

```
>> f=sin(x)+0.3*sin(4*x);
>> plot(x,f)
```



Om vi hade inte skrivit ett semikolon (`;`) sist i uttrycket för `x` och `f`, hade alla värden skrivits ut på skärmen och det vill vi nog inte.

Uppgift 2. Rita grafen av $f(x) = \sin(x) + 0.3\,\sin(5x)$ för $0 \leq x \leq 4\pi$.

Vi kan använda uppåtpil ($\uparrow$) för att komma till ett kommando vi givit tidigare. Om vi vill kan vi gå längs raden med vänster- och högerpilarna ($\leftarrow$), ($\rightarrow$) och redigera kommandot. När kommandot ser ut som vi vill trycker vi på enter ($\hookleftarrow$).

Vill vi rensa Command Window så ger vi kommandot `clc` och vill vi rensa Figure 1 ger vi kommandot `clf`. Vill vi ta bort variabler gör vi det med `clear`.

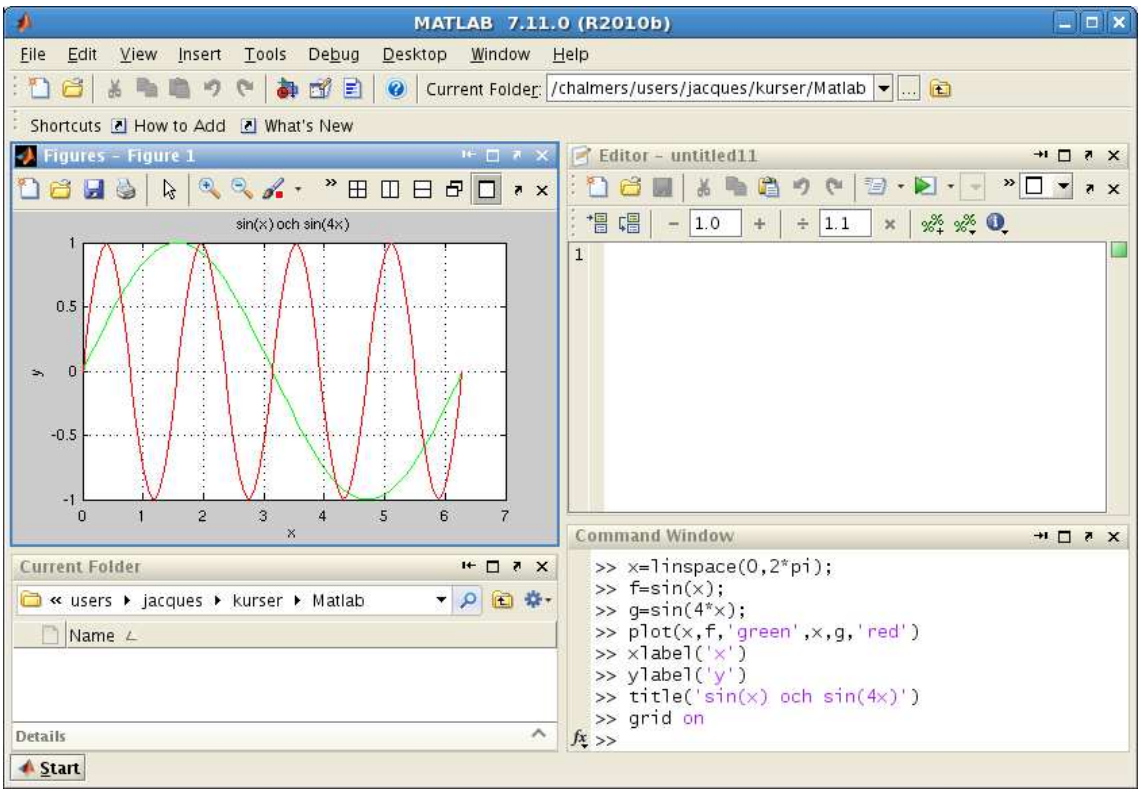
Exempel 3. Rita graferna av $f(x) = \sin(x)$ och $g(x) = \sin(4x)$ för $0 \leq x \leq 2\pi$. Sätt rubrik och text på axlarna.

Vi använder funktionen `linspace` för att göra en radvektor `x` av x-värden jämnt fördelade mellan 0 och 2π, så att graferna blir jämna och snygga. Vi gör också två radvektorer `f` och `g` med motsvarande funktionsvärden.

```
>> x=linspace(0,2*pi);
>> f=sin(x);
>> g=sin(4*x);
>> plot(x,f,'green',x,g,'red')
```

Vi ritar båda graferna samtidigt med `plot`, både paret `x`, `f` och paret `x`, `g`. För att skilja graferna åt gör vi $\sin(x)$-grafen grön `'green'` och $\sin(4x)$-grafen röd `'red'`.

Så här set det ut.



Vi sätter text på axlarna och rubrik samt lägger på ett rutnät med

```
>> xlabel('x')
>> ylabel('y')
>> title('sin(x) och sin(4x)')
>> grid on
```

Vill vi ta bort rutnätet, gör vi det med `grid off`. Texterna inom apostrofer (`' '`) är s.k. textsträngar. Exempelvis är `'green'`, `'x'` och `'sin(x) och sin(4x)'` textsträngar.

4 Script

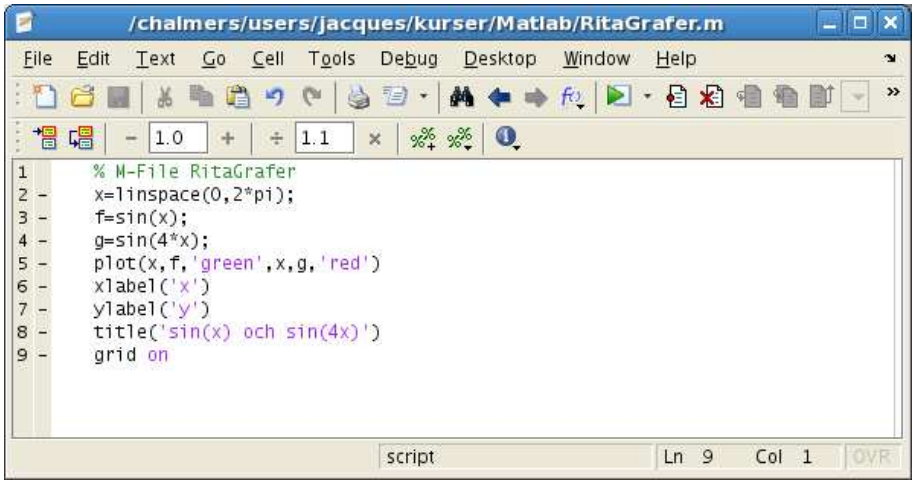
För att slippa skriva om sina kommandon, eller bläddra med uppåt- och nedåtpilar (↑), (↓) så brukar man oftast skriva sin kod i ett script eller en *skriptfil*. Detta är en textfil som innehåller det man skulle kunna skriva direkt vid promptern (>>) i Command Window, och som utförs i MATLAB genom att man ger textfilens namn som kommando.

MATLAB har en inbyggd editor som är det bästa verktyget att göra ett script med. Om man inte redan har editorn uppe på Desktop så startas den genom att man går till File, sedan New och väljer

Script. Editorn markerar koden med olika färger för att visa vad som är kommentarer, nyckelord, textsträngar, etc. (Kommentarer inleds med procenttecken.)

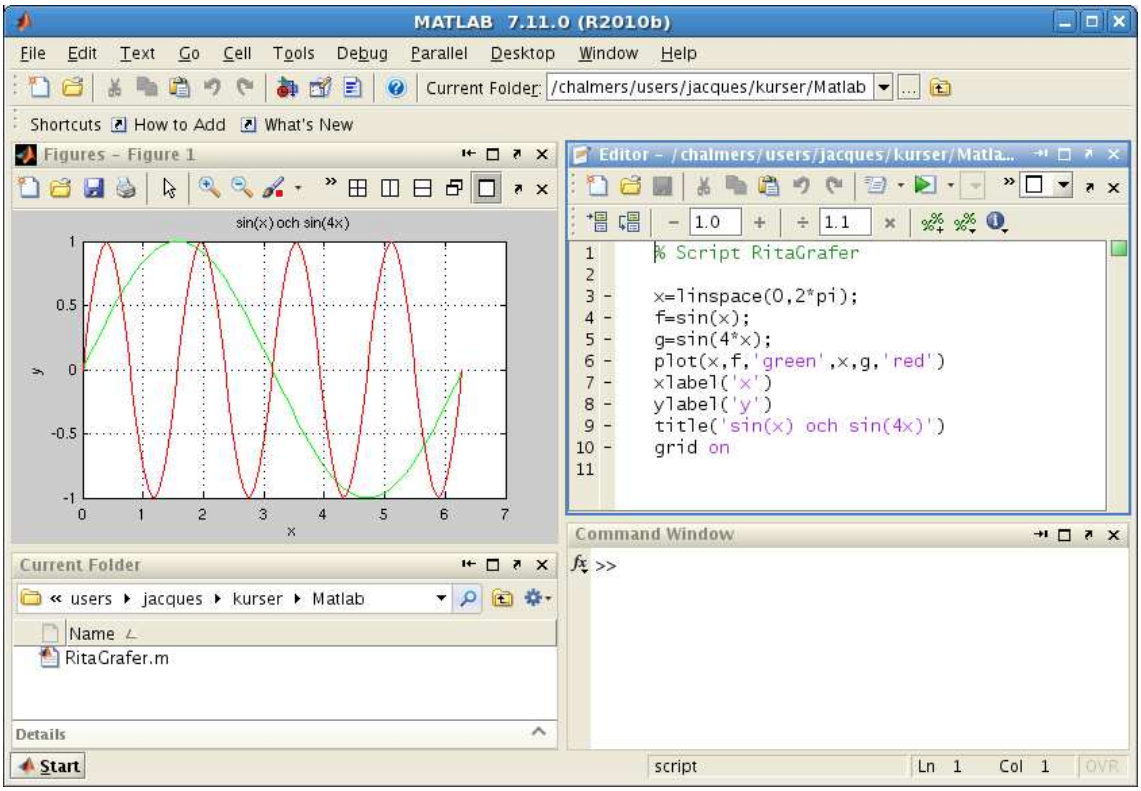
För att MATLAB skall hitta filen, förutsätter det att katalogen där filen ligger är aktuell katalog. Man kan byta katalog med kommandot cd i Command Window, klicka sig fram i Current Folder eller använda Browse for folder i verktygsfältet på Desktop. Utanför MATLAB får namnet på ett script tillägget .m för att skilja den från andra filer.

Vi gör ett script som ritar graferna från exempel 3.

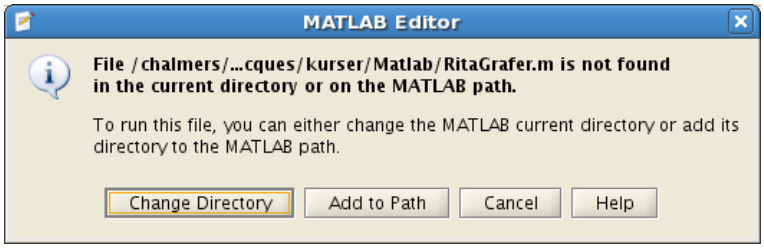


Spara kan man göra under File och köra under Debug. Enklast är dock att trycka på  som finns i verktygsfältet. Då sparas vårt script och utförs som om vi gav den som ett kommando.

Alla eventuella utskrifter från programmet skrivs i Command Window, liksom alla eventuella felmeddelanden.



Om filen ligger i en annan katalog än den aktuella, så får man upp en fråga om att byta katalog:



Välj Change Directory så byter MATLAB katalog och vårt script körs.

5 Lite programmering

I MATLAB finns repetitions- och villkorssatser som påminner om motsvarande i programspråk som C och Java. Vi nöjer oss för tillfället med att se på en repetitionssats, en `for`-sats, som vi använder för att beräkna en summa i följande exempel.

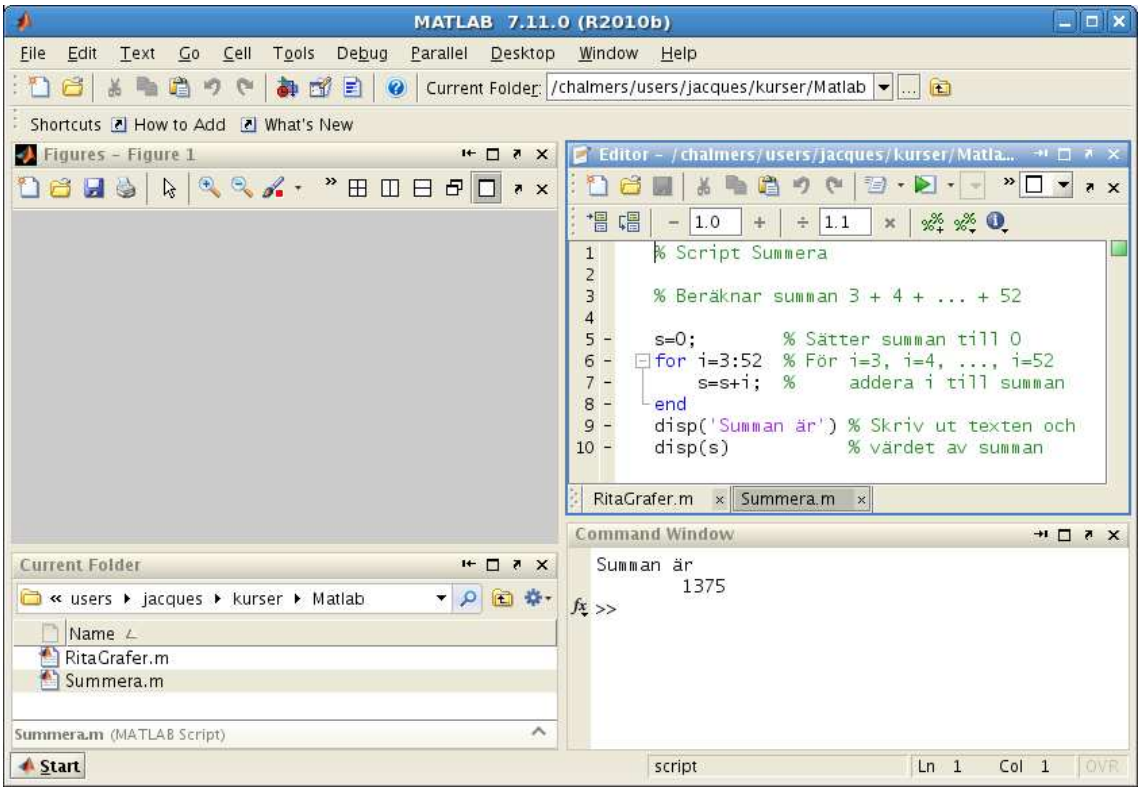
Exempel 4. Beräkna summan $s = 3 + 4 + 5 + \cdots + 52$

Vi gör ett script med programkoden

```
s=0;
for i=3:52
    s=s+i;
end
```

Vi skriver lämpliga kommentarer (grön text) i programkoden och gör lämplig utskrift med `disp`, först textsträngen `Summan är` och sedan summans värde.

Så här ser det när vi kört vårt script genom att trycka på  .



I matematik skriver man gärna summan $3+4+5+\cdots+52$ med beteckningen

$$\sum_{i=3}^{52} i$$

Uppgift 3. Skriv ett script som beräknar summan

$$\sum_{i=1}^{5} i^2 = 1^2+2^2+3^2+4^2+5^2$$

6 Function

Det finns olika sätt att göra egna funktioner i MATLAB. Om funktionen innehåller flera uttryck eller satser måste man göra en function eller *funktionsfil*, dvs. skapa en textfil med funktionsbeskrivningen. Består funktionen av ett enda uttryck så kan vi göra en s.k. anonym funktion (anonymous function).

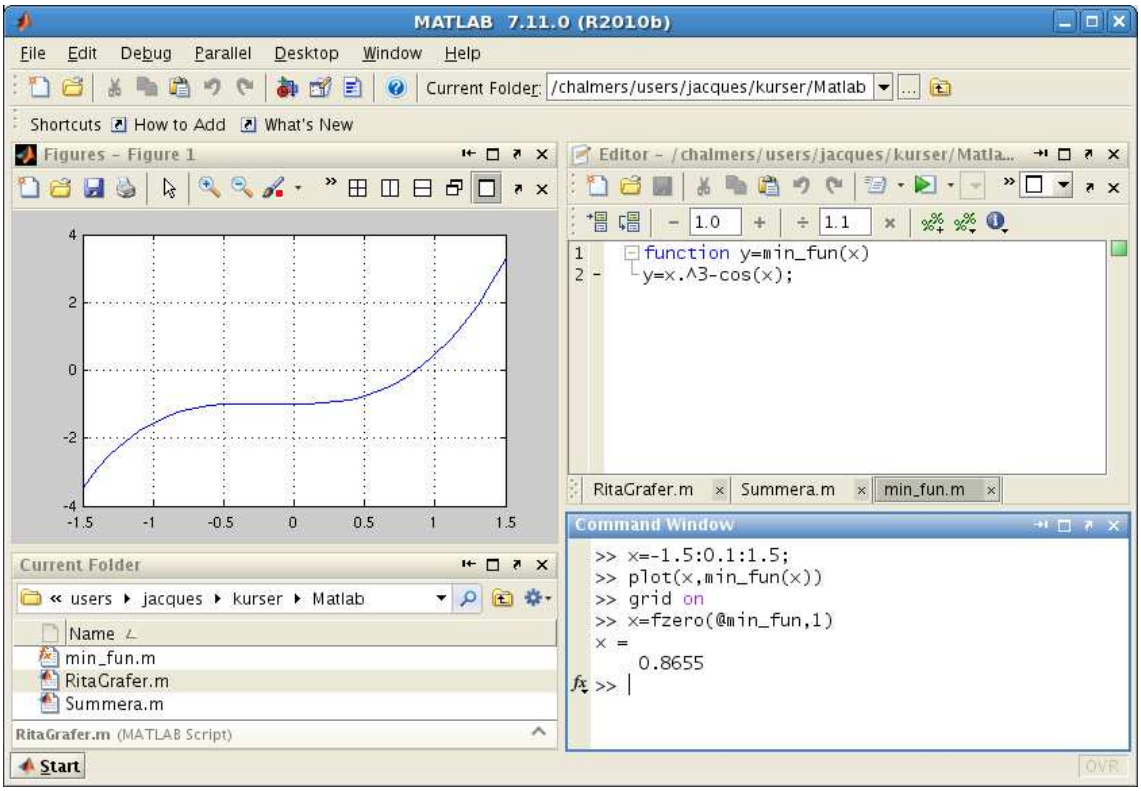
Exempel 5. Vi vill hitta ett nollställe till funktionen $f(x) = x^3 - \cos(x)$.

Det finns en funktion `fzero` i MATLAB som hittar nollställen. För att använda `fzero` måste vi beskriva vår funktion och det gör vi som en function, som vi skriver in i editorn.

Vi beskriver funktionen med

```
function y=min_fun(x)
y=x.^3-cos(x);
```

där `y` är funktionens värde (utdata), `x` är funktionens argument (indata) och `min_fun` är det namn vi har givit funktionen.



Vi ritar grafen och använder `fzero` direkt i Command Window med

```
>> x=-1.5:0.1:1.5;
>> plot(x,min_fun(x))
>> grid on
```

Vi ser att vi har ett nollställe nära $x = 1$ och låter `fzero` söka nollstället genom

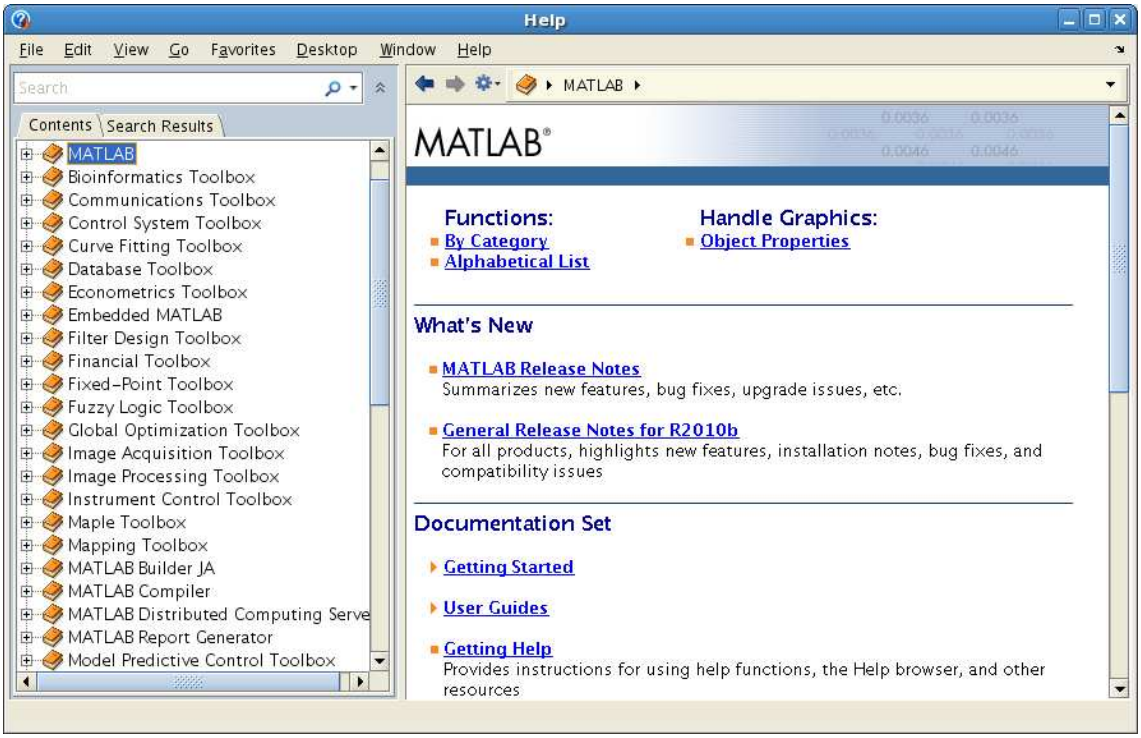
```
>> x=fzero(@min_fun,1)
x =
    0.8655
```

Med `@min_fun` talar vi om för `fzero` vilken funktion den skall hitta nollställe till. Vi använder en funktion `fzero` som i sin tur använder vår funktion `min_fun`. At-tecknet (`@`) ger ett s.k. funktionshandtag (function handle).

7 Helpdesk i MATLAB

Den mest utförliga och aktuella beskrivning som finns av MATLAB hittar man i det inbyggda hjälpsystemet Helpdesk.

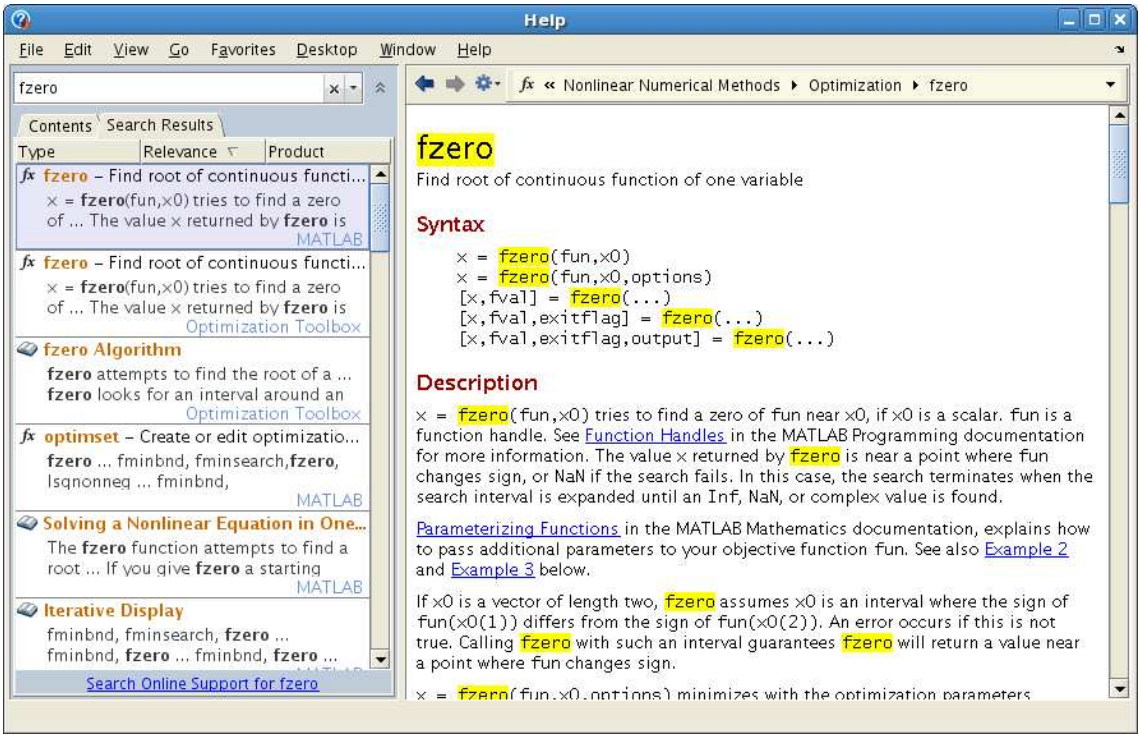
Tryck på  i verktygsfältet, eller på MATLAB Help under Window, och Help Navigator startas. Vi ser den stora uppsättningen av verktygslådor, för olika tillämpningsområden, som följer med.



Det är viktigt att lära sig att läsa dokumentationen. Den är inte skriven för att lära ut till nybörjare hur man löser ett problem med MATLAB, utan för att visa den något vane användaren exakt hur en funktion eller ett kommando används. Det är inte lättläst, och man måste lära sig att plocka

fram den informationen som är av intresse för tillfället, dvs. man måste lära sig att ”skumma” texterna.

Vi skriver `fzero` i sökrutan och trycker på enter. (Läs gärna lite i texten och titta tillbaka på exempel 5 där vi använde `fzero`.)



Man kan också titta på dessa hjälptexter med kommandon som ges i Command Window: `help` som ger texten i Command Window, t.ex. `help fzero`, och `doc` som plockar fram aktuell referenssida i webläsaren. Texten är ungefär samma, men för vissa kommandon (speciellt de för grafik) så innehåller doc-sidan bilder, vilket kan vara till hjälp, medan `help` enbart visar ren text.

8 Redovisning

Från och med nästa studio-övning kommer man varje vecka redovisa vissa uppgifter för handledaren i studion. Detta kommer sammantaget med redovisningen av en projektuppgift i programmering att utgöra examinationen på laborationsdelen av kursen.

9 Inför nästa studio-övning

Som lärobok kommer vi använda ”MATLAB for Engineers” av Holly Moore. Läs avsnitten 2.1-2.2, 2.4.3 och 3.2 som handlar om det vi jobbat med idag. Inför nästa studio-övning är det viktigt att man i förväg läser igenom studio-texten för aktuell vecka och läser de avsnitt i läroboken som anges i studio-texten, dvs. avsnitten 2.3, 3.1-3.4, 3.5.1-3.5.5, 4.1 och 7.4.