

En mycket kort introduktion till programmet Matlab

Frida Gleisner, Matz Lenells och Anders Hultgren 08 juni 2015.

Anders Hultgren 11 juni 2019

Texten avser vara en kort introduktion vid den första kontakten med beräkningsprogrammet Matlab.

Matlab är ett programpaket för tekniska beräkningar, simuleringar och visualiseringar. Det är allmänt spritt såväl inom högskolor och universitet som inom industrin.

Introduktionen till Matlab är skriven för att följas tillsammans med att man kör Matlab.

Inne i programmet Matlab finns mer hjälp och ett stort antal demonstrationer. Vad som återfinns är olika beroende vilken version av Matlab som används. Vissa versioner erbjuder en grundläggande introduktion till Matlab. [Mathworks.com](https://www.mathworks.com)

Introduktionen är skriven för att ge initial hjälp med Matlab. Efter genomgången ska du kunna:

- Starta Matlab
- Känna igen de olika fönstren för operatörens interaktion.
- Förstå vad som kallas "Current Folder".
- Förstå matrisformat som används i Matlab.
- Utföra enkla beräkningar i Command Window i Matlab.
- Generera diagram.
- Skriva enkla skript (program) som m-filer.
- Kör demos (demonstrationer) i Matlab.

Starta Matlab

Beroende på hur du kör Matlab förfar du på olika sätt. Om du har programmet installerat på din dator, starta det på normalt sätt. Kontakta annars din it-hjälp på företaget. Hur Matlab ser ut beror på vilken version du väljer att använda. Instruktionen nedan är skriven för version 2017 som även gäller för version 2019.

Att välja skrivbordets layout och att förstå vad som kallas "Current Folder"

I Fönstrets allra översta list står skrivet MATLAB Rxxxxx, där xxxxx anger version. Omedelbart under den övre listen, hittar du menyer, "New" och "Open" och många fler.

Där finns en meny "Layout" som möjliggör anpassning av hur Matlabs olika fönster visas. Välj "Layout", "Default". Resultatet bör ge tre kolumner med "Command Window" i mitten. Över till vänster visas filer, som t.ex. skript, i mappen "Current Folder", vilket är den mapp som är direkt kopplad till Matlab så att Matlab hittar filerna utan explicit sökväg. Vilken mapp detta är visas och kan ställas in omedelbart under menyknapparna. Nere till vänster visas eventuell hjälptext för markerad fil i Current folder. I högra kolumnen visas "Workspace", vilket är alla i primärminnet lagrade variablerna och parametrarna. I menyn Layout finns möjligheten att anpassa utseendet av Matlabs fönster.

Där finns också menyn "Preferences" där olika andra anpassningar kan göras. Som exempel kan fontstorleken i Matlabs olika fönster ställas in. Välj då "Preferences, "Matlab", "Font".

Några enkla beräkningar med figurgenerering gjorda i Command Window, respektive i ett skript.

Placera markören i Matlabs kommandofönstret och tryck enter för att göra fönstret aktivt.

Skriv

ver

i kommandofönstret och tryck enter. Du kommer då att se vilken version av Matlab du kör.

Skriv

`t = [0:2:10]`

och tryck enter. Denna instruktion kommer att tilldela en radvektor, `t`, värden. Det första värdet är 0 och sista värdet är 10 och avstånden mellan värdena är 2. Vektorn blir då `[ 0 2 4 6 8 10]`. I vårt fall kan vi se vektorn `t` som innehållande en uppsättning av tidpunkter.

Skriv

`t = [0:0.1:10];`

och tryck enter. Denna instruktion ger 101 värden mellan 0 och 10. Semikolonet i slutet kommer att undertrycka att Matlab visar resultatet av din instruktion.

Skriv

`w = 2;`

och tryck enter. `w` kan ses som vinkelhastighet. Skriv

`y = sin (w*t);`

och tryck enter. `w * t` får då storheten vinkel, och `w * t` blir en vektor samma storlek som `t`.

Sinusoperationen utförs för varje element i `w * t`. `y` blir då även den en vektor av samma storlek som `t`. Ge instruktionen `y=sin(w*t)` utan semikolon i slutet om du vill kontrollera.

Skriv

`plot (t,y)`

och tryck enter. Du kommer nu att kunna se ett nytt diagramfönster, figur 1. (Om du inte ser fönstret, skriv `figure (gcf)` och tryck enter. Kommandot `gcf` kan tolkas som `get current figure`, det ger referensen till det senast aktiva fönstret. Du ska kunna se en sinussignal i fönstret.

Observera att Matlab är ett program för numeriska beräkningar, dvs. vi har endast ett antal numeriska värden för några punkter på sinusfunktionen. Plot-operationen knyter samman dessa punkter med rätta linjer.

För att se detta tydligare skriv

`plot (t, y, '.')`

och tryck enter. Efter `y` finns ett kommatecken följt av en apostrof, en punkt och en ny apostrof.

Apostrofen genereras av knappen ovanför den högra shift-tangenten. I diagrammet kan du se de enskilt beräknade värdena.

Hjälp text för en instruktion med känd syntax får du genom att skriva `help` instruktionsnamnet. Försök att förstå kommandot `plot` genom att läsa hjälptexten du får när du skriver `help plot` i kommandofönstret och tryck enter.

I fönstret "Workspace", till höger, kan du se dina definierade variabler med lite information om dem. Du kan se att alla variabler presenteras som matriser, med sin matris dimension som antal rader gånger antal kolumner. Variabeln `t` bör ha storleken `1x101`, som en radvektor med 101 element. `w` har storleken `1x1`, som en skalär.

Att förstå hur matriser skrivs och hanteras i Matlab.

Gör kommandofönstret aktivt, genom att placera markören i kommandofönstret och tryck enter. Skriv `whos` och tryck enter. Du kommer även här se de olika variabler som du har definierat i Matlab.

När du följer nedan instruktioner, avsluta varje rad kommando med att trycka enter. Skriv

```
a = [1; 2];
```

```
b = [1 2];
```

```
c = [1
```

```
2];
```

```
a
```

```
b
```

```
c
```

```
a '
```

(Tecknet `'` finns normalt ovanför den högra shift-tangenten.) Du bör nu kunna se de tre definierade variablerna i kommandofönstret. Försök att förstå matrisformaten du fick för dem när du skrev på olika sätt. `;` betyder ny rad, enter ger också ny rad och Tecknet `'` ger transponering av en matris, dvs. rader blir kolonner och kolonner blir rader.

Skriv

```
A = 10;
```

```
a
```

```
A
```

Du ser att Matlab skiljer på versaler och gemener.

Att utföra enkla beräkningar i kommandofönstret i Matlab.

Skriv följande kommandon, som alla ska avslutas med att trycka enter.

```
a * b
```

```
b * a
```

```
b * c
```

```
a * c
```

Sistnämnda kommandot bör ge ett felmeddelande.

```
Error using *
```

```
Inner matrix dimensions must agree.
```

Utför kommandot

```
a '* c
```

Då bör Matlab eka 5 i kommandofönstret eftersom $a'*c = 1*1+2*2$.

Du har nu gjort olika matrismultiplikationer.

Om du önskar, konsultera Wikipedia för definition av matrismultiplikationer.

Skriv

```
hjälp *
```

och titta igenom (läs inte hela) texten om olika instruktioner.

Att skriva enkla skript som m-filer.

Skriv

```
edit sineplot
```

och tryck enter.

Svara ja på frågan i popup-fönstret.

Då ska du få ett nytt fönster med en matlabskripteditor. I detta fönster kan du stapla Matlab-instruktioner och av dem skapa ett Matlab-program, Matlabskript. Fönstret visar innehållet i filen du namngav sineplot. Filen kommer att få tillägget m och kallas m-fil. Skriv följande i editorfönstret.

```
%% Mitt Matlabskript
```

```
% Sine plot
```

```
%
```

```
%% Definition av parametrar och variabler
```

```
%
```

```
% Definition av tiden
```

```
t = [0:0.1:10];
```

```
% Definition av vinkelhastighet
```

```
w = 2;
```

```
% Definition av amplitud
```

```
A=2;
```

```
% Definition av fasförskjutning
```

```

phi=-45*pi/180;
%% Beräkningar
%
% Beräkning av sinus
y = A*sin (w*t+phi);
%% Plot av diagrammet
%
% Aktivering av ett fönster
figure(2)
% Generera ett diagram i det aktiva fönstret
plot (t, y)
grid on
figure (gcf)

```

All text som följer på en rad efter % kommer att tolkas som kommentarer och kommer i editorn att färgas grön. Editorn kommer att dra en linje ovanför varje textrad som börjar med dubbla %-tecken följt av ett tomt utrymme. Det är för att göra koden lättare att läsa, det påverkar inte exekveringen av koden.

Tryck på knappen Save i editorfönstret. Då sparas din m-fil på den aktuella katalogen.

För att köra, exekvera, en m-fil som du definierat kan du använda namnet på m-filen som en ny Matlabinstruktion. Gör kommandofönstret aktivt, skriv det namn du gav din m-fil, i det här fallet sineplot

i fönstret, och tryck på enter.

Då exekverar Matlab de instruktioner som du skrivit i filen och du kommer förhoppningsvis att kunna se ett figurfönster "figure 2" med samma plot som du har i figur 1. Du har nu skrivit ett Matlab-program. Om du tittar i Current folder katalogfönstret bör du kunna se filnamnet sineplot.m.

Gå tillbaka till editorn och skriv följande sist i m-filen. Kopiera inte eftersom ' -tecknen ofta blir felaktiga vid kopiering.

```

title ('Sine plot')
xlabel ('Tid sekunder')
ylabel ('Sine (w*t)')

```

Apostroferna ska vara av samma typ som du använt ovan, tangenten ovanför den högra shift-tangenten. Tryck på spara och kör filen igen genom att skriva namnet i Commando-fönstret följt av enter. Du kommer då att få en figurrubrik och information om diagramaxlarna.

Dokumentation av parametervärden i figur.

Det är en god vana att automatiskt dokumentera parametervärden som är använda i en beräkning vars resultat visas i en figur. Detta kan göras för vår sinus på följande sätt

```
plot(t,y,'LineWidth',2)
set(gca,'FontSize',14);%font size change in axis numbers
title({'Sine wave,';...
      ['Frequency =', num2str(w/2/pi,2),'Hz,'];...%omvandling
number to string
      ['    Amplitude = ', num2str(A,2), ','];...
      ['    Phase shift = ', num2str(phi*180/pi,2), 'deg'];...
      },'FontSize',12);
xlabel('t [s]','FontSize',12),
ylabel('Amplitude ','FontSize',12)
grid on
```

Kör demos (demonstrationer) i Matlab.

Skriv

demos

i Matlab kommandofönstret. Du kommer då att få ett nytt hjälpfönster. Layouten av Hjälp-fönstret varierar mellan olika versioner av Matlab. Försök att hitta några intressanta demon att köra.

Kopiera en m-fil till den aktuella katalogen och exekvera filen.

På kursens hemsida kan du finna en m-fil som heter "rayrectangle3.m". Lägg den filen på Matlabs Current Folder. Kör filen genom att ge skriva

rayrectangle3

i kommandofönstret och tryck enter.

Förhoppningsvis kan du se en stråle utbreda sig i ett figurfönster.

Skriv

edit rayrectangle3

i kommandofönstret och tryck enter. Läs filen och försök förstå Matlabinstruktionerna i m-filen.

Ändra något i filen rayrectangle3, t ex ändra vissa parametrar som fönsterstorleken och strålens utbredningshastighet. Prova att exekvera den modifierade filen och se om rätt saker händer.

Filen är ganska rikligt kommenterad. Den är bland annat indelad i olika delar kallade operationer. Om man vill tydliggöra programmets struktur kan man rita flödesschema av typen som finns som illustration i powerpointserien **"Några programmeringsbegrepp"**. I det dokumentet presenteras även tre programmeringsbegrepp som används i rayrectangle3.

Filen rayrectangle3.m och framförallt dess struktur kommer att användas vid lösningen av den första inlämningsuppgiften, T1.

Namngivning av variabler och m-filer (skript)

Nedan text är kopierad från Matlabs "help" som finns i övre menyn. Man bör vara varsam vid namngivning av variabler, m-skript, funktioner samt modellfiler. Grundregeln är att välj inte samma namn på något av dessa. Det finns ett undantag, variabler inne i en funktion är endast lokala namn som alltså inte kan nå utifrån. De kan ha samma namn utan att konflikt uppstår.

Variable Names

Valid Names

A valid **variable name** starts with a letter, followed by letters, digits, or underscores. MATLAB® is case sensitive, so A and a are *not* the same **variable**. The maximum length of a **variable name** is the value that the `namelengthmax` command returns.

You cannot define **variables** with the same **names** as MATLAB keywords, such as `if` or `end`. For a complete list, run the `iskeyword` command.

Examples of valid names :	Invalid names :
x6	6x
lastValue	end
n_factorial	n!

Conflicts with Function Names

Avoid creating **variables** with the same **name** as a function (such as `i`, `j`, `mode`, `char`, `size`, and `path`). In general, **variable names** take precedence over **function names**. If you create a **variable** that uses the **name** of a function, you sometimes get unexpected results.

Check whether a proposed **name** is already in use with the `exist` or `which` function. `exist` returns 0 if there are no existing **variables**, functions, or other artifacts with the proposed **name**. For example:

```
exist checkname
ans =
    0
```

If you inadvertently create a **variable** with a **name** conflict, remove the **variable** from memory with the `clear` function.

Another potential source of **name** conflicts occurs when you define a function that calls `load` or `eval` (or similar functions) to add **variables** to the workspace. In some cases, `load` or `eval` add **variables** that have the same **names** as functions. Unless these **variables** are in the function workspace before the call to `load` or `eval`, the MATLAB parser interprets the **variable names** as function **names**. For more information, see:

- [Unexpected Results When Loading Variables Within a Function](#)
- [Alternatives to the eval Function](#)

See Also

[clear](#) | [exist](#) | [iskeyword](#) | [isvarname](#) | [namelengthmax](#) | [which](#)

