Matematikk 1000

Øvingeoppgaver i numerikk – leksjon 1

Å komme i gang

I denne øvinga skal vi bli litt kjent med MATLAB. Vi skal ikkje gjøre noen avanserte ting i dette oppgavesettet – bare få et visst innblikk i (overflata av) hva MATLAB kan gjøre. Så godt som alt vi skal gjøre i dette kurset, kan like godt gjøres i Octave, som er gratis, som i MATLAB. Når vi har valgt å bruke MATLAB i stdet for Octave, har blant anne med å gjære at MATLAB har et noe mer tilgjengelig brukergrensesnitt enn Octave – og at Octave iblant kan trenge noen justeringer når det kommer en oppdatering av operativsystemet.

Det kan være nyttig å ha heftet "MATLAB Primer" tilgjengelig (det ligger på Fronter). Dette kan leses gjennom eller brukes som en slags oppslagsbok. Men det bør også presiseres at dette heftet legger lista noe høyere enn det vi kommer til å gjøre i dette kurset hva numeriske metoder angår. Videre har MATLAB et søkevindu oppe i høgre hjørne. Der kan man søke etter diverse funksjoner og få informasjon om hva de gjør.

Det at det er mye tekst i oppgavesettet betyr ikke nødvendigvis at det er mye å gjøre. En del av dere vil nok oppleve at instuksjonene i disse oppgavene er mer detaljerte en strengt tatt nødvendig. Grunnen til dette, er at vi ønsker at det tekniske skal by på minst mulig utfordringe – også for de som ikke har brukt datamskiner på denne måten tideligere. Så vi håper du bærer over med "detaljstyringa", eventuelt.

Meningen med disse oppgavene er å pirre nysgjerrigheten din på om hva vi kan bruke MATLAB til. Dersom nysjerrigheten din drar deg i litt andre retninger enn det disse oppgavene forsøker å gjøre, så er det helt greit.

Oppgave 1 – Få MATLAB opp å gå

a) MATLAB kan lastes ned fra nettsida til MathWorks. På Fronter finner du notat som forklarer hvordan dette kan gjørest.

   Installér MATLAB på datamaskina di.

b) Når installasjonen er vel gjennomført, start opp MATLAB. Ved oppstart vil et panel med fire viduer dukke opp:

- Mappevinduet, "*Current Folder*", som viser hvilken mappe du jobbar i.
- Kommandovinduet, "*Command Window*", hvor en utfører de ulike kommandoane.
- Variabelvinduet, "*Workspace*", som viser hvilke variable som ligger i minnet.
- Kommandohistorien, "*Command History*" – ei liste som viser alle kommandoene du har utført i arbeidsøkta.

I dette oppgavesettet skal vi stort sett bare prate om kommandovinduet.

Oppgave 2 – MATLAB som kalkulator

Her skal vi kort forklare hvordan man utfører enkle aritmetiske operasjoner i MATLAB. Dette er definitivt å "skyte spurv med kanon", men vi må jo begynne et sted.

a) Forsøk å skrive inn enkle aritmetiske regnestykker i kommandovinduet. Du kan for eksempel prøve å regne ut

```
2+2
3-2
3*2
6^2
6/2
```

Gjør gjerne flere – og mer avanserte – regnestykker. Som du sikkert ser, er `'*'` symbolet som blir brukt for multiplikasjon, `'/'` blir brukt for divisjon og `'^'` blir brukt for å skrive potenser.

b) Hva blir regna ut her?:

```
sin(0.7)
log(5)
sqrt(9)
exp(1)
atan(2)
```

c) Finn ut hvordan MATLAB tolker disse regnestykkene:

```
1/0
0^0
5/Inf
0*Inf
Inf/Inf
```

```
10^999
1.2e4
NaN+1
```

Er du enig i alle disse tolkningene; virker de rimelige? 'Inf' betyr uendelig, ∞, (*infinty*), og 'NaN' betyr 'udefinert' (*Not a number*). Som du sikkert ser, betyr '1.2e4' $1.2 \cdot 10^4$ (tall skrevet på *normalform*).

d) Noen tall er så spesielle at de har fått egne navn. I kommandovinduet, skriv

```
pi
i
eps
```

'`eps`' refererer her til den greske bokstaven epsilon, ϵ, som ofte blir brukt for å symbolisere små tall. Dette tallet gir størrelsen på avrundingsfeilen maskina gjør når den regner. Vi kaller dette '*maskinpresisjonen*' og, som navnet tilsier, varierer denne presisjonen noe fra maskin til maskin. i blir kalt 'den imaginære enheten' og kan oppfattes som $\sqrt{-1}$ (joda, vi *kan* tillate oss å ta kvadratrota av minus én; dette kommer vi tilbake til).

Oppgave 3 – Tilordning

a) Vanligvis vil vi ha behov for å operere med flere tallstørrelser samtidig. Vi kan *tilordne* tallverdier til variabelnavn slik:

```
>x=7
```

"Større enn"-symbolet blir her brukt for å indikere at dette blir skrevet inn i kommandovinduet i MATLAB. Vi har nå sagt at variabelen x skal ha verdi 7. Vi kan så bruke denne variabelen i andre regnestykker. Regn ut

```
>x*2
>x^2
```

Vi kan fortsette å tilordne flere variable:

```
>y=2
>x/y
```

Etterhvert som du tilegner flere variable, vil de dukke opp i lista i variabelvinduet. (Du kan også få opp en liste over alle variable i kommandovinudet ved å skrive '`who`'.)

Når vi gjør en utregning uten å gi svaret noe navn, slik vi gjorde i oppgave 1, vil svaret automatisk få navnet `ans` (for *answer*).

b) Når vi bruker symbolet `'='` som i forige del-oppgave, betyr det slett ikke "er lik", som i matematisk notasjon. Her er likhetstegnet brukt som **tilordning**; vi putter tallet 7 inn i variabelen `x`. Det følgende gir faktisk mening i MATLAB:

```
>x=2
>x=x+1
```

Hva får vi til svar om vi gjør denne operasjonen? Hvorfor er den siste linja meningsløs om vi skal tolke den matematisk?

c) Undersøk hva som skjer om du utfører følgende:

```
>pi
>pi=10
```

Hvilken verdi vil variabelen `pi` få?

Oppgave 4 – Vektorer

a) Variablene i MATLAB kan være tall, vektorer eller tabeller (*matriser*). Retnok kan også variable være korte tekster, såkalte *strenger*,men det skal vi ikke gå nærmere inn på her. Vi skal heller ikke si noe mer om tabeller/matriser – enda.

Vi kan for eksempel gi vektoren $\mathbf{x} = [1, 0, -3, 7]$ på denne måten:

```
>x=[1, 0, -3, 7]
```

Man kan også gjøre det samme uten komma, bare mellomrom, mellom elementene i vektoren.

Undersøk hva som skjer om du skriver '`x(2)`' i kommandovinduet. Hva får du om du skriver '`x(end)`', '`x(1:3)`' eller '`x(3:end)`'?

Vi ser at når vi tilordner vektoren $[1, 0, -3, 7]$ til variabelen `x`, blir variabelen `x` skrevet ut til skjerm. Om `x` hadde vært en stor vektor, kunne dette bli ganske plagsomt. Undersøk hva som skjer om vi legger til et semikolon, `';'`, etter tilordinga; '`>x=[1, 0, -3, 7];`'.

b) Forsøk å regne ut $\mathbf{x}^2$. Gir meldinga du da får mening? Forsøk å skrive `'x.^2'`. Hvordan tolker du dette svaret?

c) Forsøk å skrive følgende: `1:5`. Ut fra dette, hvordan kan du enkelt skrive opp alle heltall fra -3 til 3? Om vi vil liste opp alle tall fra og med a til b i steg på d, kan det skrives slik: `a:d:b`. Forsøk å skrive en vektor med alle disse tallene på en effektiv måte: 0, 0.25, 0.5, ..., 1.75, 2.

d) La y vere en vektor som består av alle partall fra og med 2 til og med 10. Hva skjer om du skriver '`> length(y)`'? Og hva skjer om du skriver '`> fliplr(y)`'? (De to siste bokstavene navnet til funksjonen står for "left-right".) For å lære mer om funksjonen, kan man søke på `fliplr` i det lille søkevinduet oppe til høgre eller skrive `help fliplr` i kommandovinduet.

e) Lag en vektor med fem eller flere komponenter og gi den et eller annet navn. Navnet kan godt inneholde flere enn én bokstaver. Men det får ikke begynne med et tall. Forsøk å få omkalfatre vektoren din slik at tallene kommer i stigende rekkefølge. Ikke gjør dette "for hånd"; finn en funksjon i MATLAB som gjør dette for deg. Klarer du å finne en funksjon som summerer alle tallene i vektoren?

Du avslutter MATLAB ved å klikke på '`quit`' under nedtrekksmenyen oppe til venstre eller ved å skrive `exit` eller `quit` i kommandovinduet.