Aalto-yliopisto
Teknillinen korkeakoulu

MATLAB

Lempeä johdatus

Vedos

Harri Hakula

30. syyskuuta, 2010

Sisällys

1 Yleistä

MATLAB on teollisuudessa ylivoimaisesti eniten käytetty matemaattinen ohjelmisto. Sen asema on useilla aloilla niin vankka, että jopa oppimateriaalia tuotetaan ns. MATLAB-muodossa – MATLABin omaa matemaattista notaatiota käyttäen.
MATLAB on saanut nimensä englannin kielen sanoista MATrix LABoratory. Aina viime aikoihin saakka MATLABin ainoa tietorakenne olikin matriisi; viimeisimmät versiot ovat tuoneet rikkaampia rakenteita käyttäjille, mutta olemassaoleva oppimateriaali pitäytyy vain klassisissa ominaisuuksissa.
MATLAB tarjoaa käyttäjälle useita käyttöliittymiä. Perinteisen komentoriviversion rinnalle on tullut integroitu kehitysympäristö (IDE), joka kuitenkin vain ohut lakkakerros komentoriviversion päällä. Tässä dokumentissa pitäydytään komentoriviversiossa UNIX-koneissa.

1.1 Käynnistys ATK-keskuksen koneissa

Satunnainen käyttäjä viihtyy parhaiten MATLABin IDEn parissa. Se on myös oletusarvoympäristö, joten käynnistyskomento `matlab` käynnistää nimenomaan IDEn. Usein IDEt on tapana käynnistää taustalle, joten `matlab &` lienee paras valinta. Tottunut käyttäjä (ns. voimakäyttäjä) useimmiten suorittaa komentojonoja tai omia ohjelmia, eikä tarvitse kaikkia käytettävyysominaisuuksia. Arkaaisen perusympäristön voi käynnistää komennolla `matlab -nojvm`, jolloin MATALBin komentorivi-istunto jatkuu samassa ikkunassa. MATLABin käynnistysoptiot saa listattua komennolla `matlab -h`. Alla on listattuna muutamia vaihtoehtoja.

Käynnistys

```
matlab &
```

Voimakäyttö

```
matlab -nojvm
matlab -nojvm -nosplash
matlab -nojvm -nosplash -nodisplay
```

2 Istunto

Tässä luvussa käsitellään perusteita, joiden avulla ainakin tehokas laskinkäyttö sekä kurssikirjojen esimerkkien läpikäynti onnistuu kyynelittä. MATALBin kehote on kaksikertainen "suurempi kuin"-merkki (>>). Allaolevissa esimerkeissä on annettu vain käyttäjän antamat syötteet. Syöte suoritetaan vain, jos se on kirjoitettu oikein ja päättyy rivinvaihtoon.

2.1 Manuaali

MATLABin sisäinen manuaali on kattava. Ohjelmisto on laaja, eikä kukaan tunne sen kaikkia ominaisuuksia. Komentorivimanuaalin perussyntaksi on muotoa `help komento`. Pelkkä `help` antaa listan otsikkoja, joista voi alkaa matkan syvemmälle. IDEssä on oma manuaaliselaimensa, jonka avulla voi lukea jopa viralliset oppaat!

Manuaalin käyttö

```
help
help sin
help :
```

2.2 Ympäristö

MATLABia voi käyttää tavallisena funktiolaskimena. MATLAB on numeerinen ohjelmisto eli kaikki lausekkeet suoritetaan eli evaluoidaan välittömästi. Viimeinen esimerkki allaolevassa listassa kertoo liukulukulaskennan ihmeellisyyksistä. Miksi tulos ei ole nolla?

Laskin

```
1 + 1
sin(2.3)
sqrt(4)
sin(1)
0.2 + 0.3 - 0.2 - 0.1 - 0.2
```

Huomaa, että jokaisen laskutoimituksen tulos sijoitetaan erikoismuuttujaan nimeltä `ans`. Tätä voi joskus käyttää hyväksi.

`ans`

```
2 + 2
ans + 2
```

MATLABin saa vaikenemaan antamalla komennon loppuun puolipisteen. Tämä ei kuitenkaan muuta sitä sääntöä, että jokaisen laskutoimituksen tulos talletataan johonkin muuttujan, joka on `ans`, jos ei muuta erikseen määrätä.

Kaiutus

```
2 + 2;
ans + 2
```

Joskus voi olla tarpeen unohtaa kaikki suoritetut laskutoimitukset ja alkaa alusta. Komennolla `clear all` MATLABin saa asetettua alkutilaan. Tätä on mukava ajatella ohjelman uudelleen käynnistyksenä.

Uudelleen käynnistys

```
clear all
```

Mikäli istunnon komennoista haluaa lokikirjan, automaattisen tallentimen saa käyttöön komennolla `diary`. Loki saattaa helposti kasvaa ja sisältää "tarpeetontakin" informaatiota, mutta sitä voi halutessaan muokata tekstieditorilla tai rajata jo suorituksen aikana (katso `help diary`).

Istunnon lokikirja

```
diary on
...
diary off
```

Ohjelma katkaistaan komennolla `quit`

Lopetus

```
quit
```

2.3 Muuttujat

Lausekkeiden arvoja voi sijoittaa nimettyihin muuttujiin. Sijoitusoperaattori on `=`. Muutamat nimet ovat varattuja, kuten `pi` ja `i`. Neperin luku e sen sijaan ei ole varattu, eksponenttifunktio on `exp` ja pätee $e =$ `exp(1.0)`. Muuttujien nimet kannattaa yleensä asettaa mielekkäiksi. Pelkkä muuttujan nimi palauttaa muuttujan arvon.

Muuttujan arvon asettaminen

```
a = 1
a
s = sin(pi)
```

Muuttujan arvo säilyy, mikäli sitä ei muuteta. Jos muuttujan haluaa poistaa istunnosta, on käytettävä komentoa `clear muuttujannimi`.

Muuttujan poistaminen

```
a = 2
clear a
a
```

2.4 Komentojonot

Useimmiten normaalikäytössä halutaan suorittaa samoja komentoryhmiä lukuisia kertoja. MATLAB tarjoaa mahdollisuuden ajaa komentojonoja ns. m-tiedostojen avulla. m-tiedostot ovat tavallisia tekstitiedostoja, jotka sisältävät MATLAB-komentoja. MATLAB suorittaa komentoja rivi riviltä, kunnes tiedoston loppu saavutetaan. Tiedoston nimen on oltava muotoa `komentojono.m`, jolloin MATLAB automaattisesti tunnistaa komennon `komentojono`. Nimi m-tiedosto tulee siis tiedoston suffiksista.
MATLABin on tietenkin löydettävä haluttu komentojonotiedosto. Komentojonoja ja m-tedostoja yleensä etsitään ns. polusta eli listasta hakemistoja, jotka on erikseen määritelty asennuksen yhteydessä. Se hakemisto mistä MATLAB-istunto on käynnistetty on aina polussa, joten helpointa lienee luoda oma hakemisto harjoituksia

varten ja tallettaa omat m-tiedostot sinne. Polkuasetuksia voi toki muuttaa (`help path`).
Esimerkissämme komentojonossa on kolme komentoa. Kun komentojono suoritetaan, kaiutussännöt ovat voimassa ja määritellyt muuttujat lisätään MATLAB-istuntoon aivan kuin ne olisi annettu komentoriviltä.

Komentojono esim.m

```
a = 2;
b = 3;
a + b
```

Ajoesimerkki

```
esim
a
b
```

2.5 Funktiot

MATLABia voi laajentaa omilla komennoilla eli funktioilla. Funktiot luodaan m-tiedostoina, joilla on erityinen muoto. MATLAB lukee m-tiedoston ensimmäisen rivin ja päättelee, onko kysessä tavallinen komentojon vai funktio. Funktion määrittely on muotoa `function palautusarvo = funktionnimi(argumentit)`.
Esimerkissä argumenttiin `a` lisätään kokonaisluku 1. Muuttujan `a` arvo ei muutu. Funktion palautusarvon nimeksi on määritelty `retval`, joten funktion suorituksen jälkeen sen arvoksi istunnossa tulee muuttuja `retval`in lopullinen arvo.

Funktio kyy.m

```
function retval = kyy(a)
retval = a + 1;
```

Ajoesimerkki

```
a = 1
kyy(a)
b = kyy(a)
```

Omien funktioiden ohjelmoinnissa tulee usein tarve silmukoiden ja vastaavien rakenteiden käyttöön. Nämä kaikki ovat luonnollisesti tuettuja (`help for`, `help while`,

`help if`). Matriisilaskun osalta seuraavassa esitetään mekanismi, jolla usein voi välttää "turhien" silmukoiden kirjoittamisen.

2.6 Matriisinotaatio

MATALBin suosio perustuu sen kykyyn käsitellä matriiseja tehokkaasti ja vaivattomasti. Matriisi kirjoitetaan hakasulkujen sisään ja rivinvaihto osoitetaan puolipisteellä. Alkioon viitataan muodossa `a(i,j)`. Indeksointi alkaa luvusta 1.

Matriisi

```
a = [1 2; 3 4]
a(1,1)
```

Matriisityyppiä voi manipuloida monin tavoin. Erityisen voimallinen välinen on ns. puolipiste-operaattori tai -kvanttori. Erilaisia indeksilistoja on suorastaan nautinnollista kirjoittaa puolipistenotaation avulla. Perusmuoto on `alaraja:askel:yläraja`.

:-operaattori

```
1:10
1:2:10
1:0.1:2
2:-0.1:1
```

Muodostetaan satunnainen 10×10-matriisi komennolla `rand`. Parittomat sarakkeet voi kerätä seuraavasti:

Esimerkki

```
A = rand(10)
A(:,1:2:10)
```

Huomaa, että kaikki rivit on kuitattu pelkällä puolipisteellä. MATLAB tietää, että kysessä on viittaus matriisin alkioihin ja tulkitsee yksinäisen puolipisteen tarkoittavan kaikkia indeksejä. Matriisin muodon saa yleisessä tapauksessa selville komennolla `size(A)`, missä `A` on matriisimuuttujan nimi.
Notaatio laajenee luonnollisella tavalla kattamaan kaikki mahdolliset tapaukset. Tulkitse seuraavat:

Esimerkkejä

```
A = rand(10)
A(1:2,1:2)
A(10:-1:1,:)
A(:,:)
```

Matriisien muodostaminen toisilla matriiseilla eli lohkomatriisien muodostaminen on myöskin helppoa. Syntaksi on muuttumaton, yksittäset alkiot korvataan matriiseilla, joiden on oltava muodoltaan sopivia. Usein tarvitaan joko nolla- tai diagonaalilohkoja. Nollamatriisit muodostetaan komennolla `zeros`, matriisi, jonka alkiot ovat ykkösiä saadaan komennolla `ones` ja identiteettimatriisi komennolla `eye`. Tulkitse seuraavat:

Lohkomatriisit

```
A = rand(2)
[A A; A A]
[eye(2) A; zeros(2) ones(2)]
```

2.7 Matriisilasku

Kahden alkion laskutoimitus on oletusarvoisesti matriisilaskutoimitus.

Yhteenlasku

```
A = rand(2)
A + A
```

Tulo

```
A = rand(3)
B = ones(3)
A * B
```

Joskus on perusteltua laskea alkioittain kahden matriisin välillä. Esimerkiksi sisätulo on määritelty

$$r = a^T b = \sum_{i=1}^{n} a_i b_i.$$

Olkoon annettuna kaksi vektoria, `a` ja `b`. Muodostetaan kolmas vektori, jonka alkiot ovat `a`:n ja `b`:n vastinalkioiden tulot ja lasketaan tämän alkioiden summa. Alkioittaiset operaatiot merkitään pisteellä, alkioittainen tulo: `.*`

Sisätulo: Ensimmäinen versio

```
a = [1 2 3]
b = [4 5 6]
c = a .* b
sum(c)
sum(a.*b)
```

Välitulos on tietenkin turha. Yhtä hyvin voisimme laskea sisätulon matriisitulon avulla. Matriisin transponointi suoritetaan heittomerkin avulla. Yllättäen joudummekin transponoimaan `b`:n, koska olemmekin koko ajan kirjoittaneet vaakavektoreita!

Sisätulo: Toinen versio

```
a = [1 2 3]
b = [4 5 6]
a * b'
```

Matemaattisesti oikea versio saadaan transponoimalla heti sijoituksen yhteydessä. Laskennallisen suorituksen kannalta on hyvä muistaa, että kaikki laskentatavat antavat saman lopputuloksen.

Sisätulo: Matemaattinen versio

```
a = [1 2 3]'
b = [4 5 6]'
a' * b
```

2.8 Piirtäminen

MATLABin grafiikkatuki on kattava. Lähes kaikki (ellei peräti aivan kaikki!) visualisointitarpeet voidaan tyydyttää MATLABin sisällä. Alussa hyödyllisin komento on `plot`. Piirretään vaikkapa funktio $\sin(x)$ välillä $[-\pi,\pi]$. Koska MATLAB on matriisiohjelmisto, on kuvadatakin annettava matriisimuodossa. Kuvaaja on murtoviiva pisteistä, joiden koordinaatit tunnetaan. Tässä tapauksessa on ensin pilkottava tarkasteluväli sopiviin osiin, joissa itse funktio sitten evaluoidaan. Pilkkominen on mukavinta komennon `linspace` avulla.

Kuvaajan piirtäminen

```
t = linspace(-pi,pi);
plot(sin(t), t)
```

Huomaa, että sini-funktion arvo laksetaan jokaisessa argumenttivektorin (-matriisin) alkiossa.
Kolmas argumentti määrää kuvaajan ominaisuuksia kuten väriä, viivatyyppiä, jne.

Punaisen kuvaajan piirtäminen

```
t = linspace(-pi,pi);
plot(t, sin(t), 'r')
```

Useita kuvaajia voi piirtää samaan kuvaan joko antamalla `plot`-komennolle useita argumentteja tai lukitsemalla kuvan `hold on` komennolla (lukon voi vapauttaa komennolla `hold off`).

Kaksi kuvaajaa samassa kuvassa

```
t = linspace(-pi,pi);
plot(t, sin(t), 'r', t, cos(t), 'k')
plot(t, sin(t), 'r')
hold on
plot(t, cos(t), 'k')
hold off
```

Tulomuotoisen funktion kuvaajan piirto ei ole suoraviivaista. Tarkastellaan aluksi tapausta $f(x) = \sin(x)\cos(x)$. Seuraava esimerkki ei toimi. Miksi?

Tulomuotoisen funktion kuvaaja: Virheellinen versio

```
t = linspace(-pi,pi);
plot(t, sin(t) * cos(t))
```

Alkeisfunktiot evaluoituvat kaikissa matriisin alkioissa ja palautusarvo on matriisi. Virheilmoitus heijastaa juuri tätä. Kuvaajaa piirrettäessä halutaan kuitenkin funktion arvo jokaisessa pisteessä eli oikea ratkaisu on käyttää alkioittaista tuloa.

Tulomuotoisen funktion kuvaaja: Oikein

```
t = linspace(-pi,pi);
plot(t, sin(t) .* cos(t))
```

Ensimmäisen esimerkin aiheuttaman hämmennyksen jälkeen on syytä rauhoittua ja todeta, että lausekkeen muodostaminen ei sittenkään ole kovin monimutkaista: On vain tiedettävä, mikä välitulos on matriisiarvoinen ja mikä skalaari. Tarkastellaan seuraavaksi monimutkaisempaa esimerkkiä:

$$f(x) = \frac{\sin(x)}{\sqrt{1+x^2}}.$$

Funktio $f(x)$ evaluoidaan pistejoukossa x. Määrittelemättömät laskutoimitukset ovat vektorin neliö ja vektoreiden jakolasku. Nämä vaativat siis ainakin alkioittaisen laskutoimituksen. Saadaan ensimmäinen versio: `sin(x) ./ sqrt(1 + x.^2)`. Matemaattisesti ongelmallinen kohta on skalaarin ja vektorin (matriisin) summa $1+x^2$! MATLABissa skalaari on tarvittaessa vakiovektori tai -matriisi ja samaa muotoa kuin vastinpari. Summa $1 + x^2$ lisää ykkösen jokaiseen neliön alkioon. Alkeisfunktiot sin ja $\sqrt{}$ evaluoituvat tutusti. Ensimmäinen versio onkin samalla lopullinen.

Yleisen funktion kuvaaja

```
t = linspace(-pi,pi);
plot(t, sin(t) ./ sqrt(1 + t.^2))
```