Laskennallinen lineaarinen algebra ja geometria

Matti Vihola

2011

1 Sisältö

- Numeriikkaa
  - äärellinen esitystarkkuus
  - numeerisen laskennan etuja ja haittoja
  - virhekäsitteitä
- Numeerista lineaarista algebraa
  - matriisinormi ja kuntoluku
  - matriisihajotelmia
  - pseudoinverssi
  - yhtälöryhmä/pienin neliösumma
  - ominaisarvot ja -vektorit
  - neliömuodot
- MATLAB-ohjelmiston käyttö
  - komentokehote
  - funktiot/skriptit
  - grafiikka

2 Numeriikkaa

- Numeerinen analyysi on jatkuvien ongelmien käsittelyä diskreeteillä algoritmeilla, jotka käyttävät äärellistä esitystarkkuutta.
- Tyypillisesti tehtävänä on laskea *likiarvoinen* vastaus annettuun tehtävään.
- Esimerkiksi "laske integraalin

$$\Gamma\big(\sqrt{2}+1\big) = \int_0^\infty t^{\sqrt{2}} e^{-t} \mathrm{d}t$$

arvo neljän merkitsevän numeron tarkkuudella" (≈ 1.254).
- Numeerinen (tai laskennallinen) lineaarinen algebra on yksi numeerisen analysiin osa-alue. Muita merkittäviä osa-alueita ovat mm. numeerinen optimointi ja differentiaaliyhtälöiden numeerinen ratkaisu.
- Mihin matemaatikko tarvitsee numeerisia menetelmiä?
  - Sovellukset: (esim. luonnontieteet, tekniikka).
  - Hahmotteluun: likiarvoiset vastaukset antavat vihiä siitä, voisiko jokin väite pitää paikkaansa vai ei.

2.1 Äärellinen esitystarkkuus

- Tietokoneen muistiin (tai paperille) ei voida tallentaa mielivaltaista lukua.
- Käytännön laskuissa luvut on pakko esittää äärellisellä esitystarkkuudella.
  - Fysiikassa/tekniikassa mittausten tarkkuus sanelee usein esitystarkkuuden.
- Tietokoneissa käytetään lukujen esittämiseen useimmiten ns. *liukulukuja* (*floating point number*).
  - Kutakin lukua varten on käytössä kaksi muistipaikkaa: toinen *mantissalle* ja toinen *eksponentille*. Esim.

$$0.01 = +1.000000 \times 10^{-002}$$

missä mantissa "+1.000000" esitetään seitsemällä merkitsevällä numerolla ja etumerkillä, ja kokonaislukueksponentti "-002" kolmella luvulla ja etumerkillä.

+	1	0	0	0	0	0	0	-	0	0	2

2.2 IEEE 754

- Yleisin liukulukujen esitys on standardin IEEE 754[1] mukainen.
- Usein ns. tieteellisessä laskennassa käytetään erityisesti "kaksoistarkkuisia" (*double precision*), eli 64-bittisiä liukulukuja.
  - Eksponentti 11 bittiä, mantissa 53 bittiä.
  - Itseisarvoltaan pienimmät luvut ovat

$$\pm 2^{-1074} \approx \pm 5 \times 10^{-324}$$

  - Itseisarvoltaan suurimmat luvut ovat

$$\pm((1-(1/2)^{53})2^{1024}) \approx \pm 1.8 \times 10^{308}$$

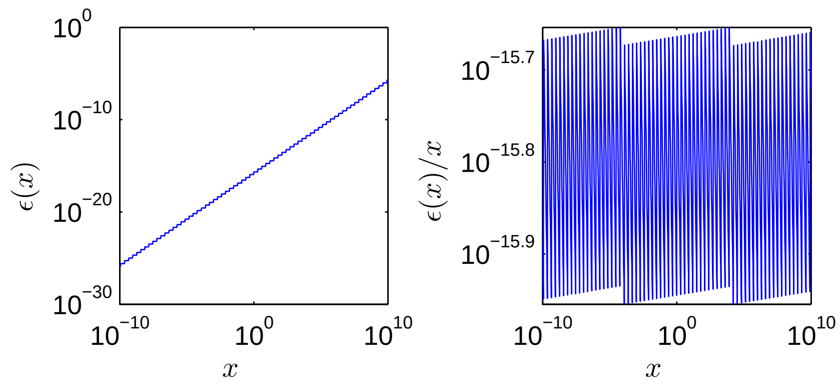
  - Voidaan esittää kokonaisluvut $-2^{53}, \ldots, 2^{53}$ tarkasti.
  - Lisäksi käytössä ± 0 ja seuraavat "luvut": "`+Inf`", "`-Inf`" (positiivinen ja negatiivinen äärettömyys), ja "`NaN`" (*Not a Number*)

2.3 Äärellisen esitystarkkuuden etuja ja haittoja

- Äärellisen esitystarkkuuden ilmeinen haitta on, että voimme esittää vain äärellisen joukon eri lukuja.
- Siitä huolimatta laskenta-algoritmit voidaan tehdä siten, että useimmiten virhe ei kasva niin suureksi, että sillä olisi käytännön merkitystä.
- Joissain tapauksissa äärellinen esitystarkkuus saattaa kuitenkin aiheuttaa lopputuloksessa *merkittävän* virheen.
- Käyttäjän pitää osata tulkita numeerisen laskennan tuloksia; ovatko ne luotettavia vai eivät.
- IEEE 754 -liukulukuesityksen etuna on, että tavallisilla kotitietokoneilla voidaan tehdä yksinkertaisia laskutoimituksia nopeasti (luokkaa miljardi laskutoimitusta sekunnissa).

2.4 Virhekäsitteitä

- Absoluuttinen virhe: Jos oikea arvo on $\boldsymbol{x}$, ja saatu arvo $\boldsymbol{x}'$, absoluuttinen virhe on $\|\Delta \boldsymbol{x}\|$ missä $\Delta \boldsymbol{x} := \boldsymbol{x}' - \boldsymbol{x}$.
- Suhteellinen virhe: $\frac{\|\Delta \boldsymbol{x}\|}{\|\boldsymbol{x}\|}$, kun $\boldsymbol{x} \neq \mathbf{0}$.
- Pyöristysvirhe (*roundoff error*): Virhe, joka tehdään, kun luku esitetään jossain äärellisessä esityksessä. Esimerkkejä:
  - $10^{100} + 1$ esitettynä IEEE 754 kaksoistarkkuuden liukulukuna on 10^{100}, joten (absoluuttinen) pyöristysvirhe on 1.
  - Desimaaliluvun 0.1 esitys on 0.1000000000000000055..., joten (abs.) virhe on noin 6×10^{-18}.
- Ylivuoto (*overflow*) tapahtuu, kun laskun tulos menee yli lukualueen. (tuloksena `±Inf`).
- Alivuoto (*underflow*) tapahtuu, kun laskun tulos ei ole nolla, mutta itseisarvoltaan niin pieni, että vastaus pyöristetään nollaksi.
  - tämä saattaa aiheuttaa esim. nollalla jaon seuraavassa vaiheessa!
- Kone-epsilon: Etäisyys luvun 1 ja sitä seuraavaksi suuremman luvun välillä (2^{-52} IEEE 754 kaksoistarkkuudessa).
  - Yleisemmin, $\epsilon(x)$ on etäisyys luvun x ja sitä seuraavan itseisarvoltaan suuremman luvun välillä.



2.5 Yhtälöryhmän ratkaisun stabiilius

Esimerkki

Yhtälöryhmällä[2]

$$\begin{bmatrix} 2 & 2 \\ 2 & 2.0005 \end{bmatrix} \boldsymbol{x} = \begin{bmatrix} 6 \\ 6.001 \end{bmatrix}$$

1. Kts. esim. `http://en.wikipedia.org/wiki/IEEE_754`.

2. Esimerkki otettu opintomonisteesta A. Perttula: "Lisää lineaarialgebraa".

on yksikäsitteinen ratkaisu $\boldsymbol{x} = [1, 2]^T$. Oletetaan, että käytössä olisi neljän desimaalin esitystarkkuus. Tällöin ratkaistaisiinkin yhtälöryhmää

$$\begin{bmatrix} 2.000 & 2.000 \\ 2.000 & 2.001 \end{bmatrix} \hat{\boldsymbol{x}} = \begin{bmatrix} 6.000 \\ 6.001 \end{bmatrix}$$

jonka yksikäsitteinen ratkaisu on $\hat{\boldsymbol{x}} = [2.000, 1.000]^T$.

Matriisin esityksessä vain yhteen komponenttiin tuli suhteellista virhettä $|2.001 - 2.0005|/2.0005 \approx 0.00025$, mutta ratkaisussa virhettä on $\|[2,1]^T - [1,2]^T\|/\|[1,2]^T\| \approx 0.63$.

Matriisinormi ja matriisin kuntoluku

- Tällä kurssilla käytetään vektoreille $\boldsymbol{x} \in \mathbb{R}^n$ aina Euklidista normia $\|\boldsymbol{x}\| := \sqrt{\sum_{k=1}^n x_k^2}$ ja matriiseille sen indusoimaa (operaattori)normia:

$$\|\mathrm{A}\| := \max_{\boldsymbol{x} \neq \boldsymbol{0}} \frac{\|\mathrm{A}\boldsymbol{x}\|}{\|\boldsymbol{x}\|} = \max_{\|\boldsymbol{u}\|=1} \|\mathrm{A}\boldsymbol{u}\|.$$

- Huomataan, että em. matriisinormi on *sub-multiplikatiivinen*:

$$\begin{aligned} \|\mathrm{A}\boldsymbol{x}\| &\leq \|\mathrm{A}\|\|\boldsymbol{x}\| && \text{kaikilla } A \in \mathbb{R}^{m\times n} \text{ ja } \boldsymbol{x} \in \mathbb{R}^n \text{ ja} \\ \|\mathrm{AB}\| &\leq \|\mathrm{A}\|\|\mathrm{B}\| && \text{kaikilla } A \in \mathbb{R}^{m\times n} \text{ ja } B \in \mathbb{R}^{n\times p}. \end{aligned}$$

- Matriisin *kuntoluku* tai *häiriöalttiusluku* (*condition number*) $\mathrm{cond}(\mathrm{A}) := \|\mathrm{A}\| \, \|\mathrm{A}^{-1}\|$ kuvaa matriisiin liittyvän yhtälöryhmän ratkaisun stabiiliutta.

Yhtälöryhmän ratkaisun virhe

- Oletetaan, että haluttaisiin ratkaista $\boldsymbol{x}$ yhtälöryhmästä $\mathrm{A}\boldsymbol{x} = \boldsymbol{b}$, missä A on tunnettu kääntyvä matriisi ja $\boldsymbol{b}$ tunnettu vektori.
- Käytössämme olevalla tietokoneella ei voida kuitenkaan esittää A:ta ja $\boldsymbol{b}$:tä tarkasti, vaan likimääräisesti, jolloin ratkaisemmekin $\hat{\boldsymbol{x}}$:n yhtälöryhmästä $\hat{\mathrm{A}}\hat{\boldsymbol{x}} = \hat{\boldsymbol{b}}$.
- Olemme kiinnostuneita, kuinka suuri ratkaisun suhteellinen virhe $\frac{\|\hat{\boldsymbol{x}}-\boldsymbol{x}\|}{\|\boldsymbol{x}\|}$ on.
  - Mikäli $\hat{\mathrm{A}} = \mathrm{A}$, nähdään että

$$\frac{\|\hat{\boldsymbol{x}} - \boldsymbol{x}\|}{\|\boldsymbol{x}\|} \leq \left\|\mathrm{A}^{-1}\right\| \frac{\|\hat{\boldsymbol{b}} - \boldsymbol{b}\|}{\|\boldsymbol{x}\|} \leq \mathrm{cond}(\mathrm{A}) \frac{\|\hat{\boldsymbol{b}} - \boldsymbol{b}\|}{\|\boldsymbol{b}\|},$$

    sillä $\|\boldsymbol{b}\| = \|\mathrm{A}\boldsymbol{x}\| \leq \|\mathrm{A}\| \, \|\boldsymbol{x}\|$.
  - Mikäli $\hat{\boldsymbol{b}} = \boldsymbol{b}$, selvästi $\hat{\mathrm{A}}(\hat{\boldsymbol{x}} - \boldsymbol{x}) = (\mathrm{A} - \hat{\mathrm{A}})\boldsymbol{x}$, joten

$$\frac{\|\hat{\boldsymbol{x}} - \boldsymbol{x}\|}{\|\boldsymbol{x}\|} \leq \|\hat{\mathrm{A}}^{-1}\|\|\mathrm{A} - \hat{\mathrm{A}}\| = \mathrm{cond}(\hat{\mathrm{A}}) \frac{\|\hat{\mathrm{A}} - \mathrm{A}\|}{\|\hat{\mathrm{A}}\|}$$

  - Yleisemmin pätee, kun $\left\|\mathrm{A}^{-1}\right\| \|\hat{\mathrm{A}} - \mathrm{A}\| < 1$, että

$$\frac{\|\hat{\boldsymbol{x}} - \boldsymbol{x}\|}{\|\boldsymbol{x}\|} \leq \frac{\mathrm{cond}(\mathrm{A})}{1 - \mathrm{cond}(\mathrm{A}) \frac{\|\hat{\mathrm{A}} - \mathrm{A}\|}{\|\mathrm{A}\|}} \left[\frac{\|\hat{\boldsymbol{b}} - \boldsymbol{b}\|}{\|\boldsymbol{b}\|} + \frac{\|\hat{\mathrm{A}} - \mathrm{A}\|}{\|\mathrm{A}\|} \right]$$

- Huomaa, että yllä olevassa käsittelyssä jätetään huomiotta yhtälöryhmän ratkaisijan välivaiheissa tekemät virheet, ts. se, että ratkaisija antaakin vastauksen $\boldsymbol{x}' \neq \hat{\boldsymbol{x}}$ yhtälöryhmään $\hat{\mathrm{A}}\hat{\boldsymbol{x}} = \hat{\boldsymbol{b}}$. Tietyllä tavalla toteutetun ratkaisijan tekemät virheet voidaan kuitenkin ajatella olevan osa matriisin A virhettä, eli mukana virheessä $\hat{\mathrm{A}} - \mathrm{A}$.
- Laskennan virheiden lisäksi on mahdollista, että matriiseihin A ja $\boldsymbol{b}$ sisältyy muutakin virhettä kuin pelkkä äärellisen esityksen aiheuttama pyöristysvirhe, esim. mittausvirhe.

2.6 Hyödyllisiä matriisi-identiteettejä

Olkoon $\mathrm{A} \in \mathbb{R}^{m\times n}$ ja $\mathrm{B} \in \mathbb{R}^{n\times p}$. Merkitään A:n ja B:n sarake- ja rivivektoreita seuraavasti

$$\mathrm{A} = \begin{bmatrix} \boldsymbol{a}_1, \ldots, \boldsymbol{a}_n \end{bmatrix} = \begin{bmatrix} \boldsymbol{\alpha}_1^T \\ \vdots \\ \boldsymbol{\alpha}_m^T \end{bmatrix} \quad \text{ja} \quad \mathrm{B} = \begin{bmatrix} \boldsymbol{b}_1, \ldots, \boldsymbol{b}_p \end{bmatrix} = \begin{bmatrix} \boldsymbol{\beta}_1^T \\ \vdots \\ \boldsymbol{\beta}_n^T \end{bmatrix}.$$

Tällöin $\boldsymbol{a}_k \in \mathbb{R}^m$, $\boldsymbol{\alpha}_k \in \mathbb{R}^n$, $\boldsymbol{b}_k \in \mathbb{R}^n$, $\boldsymbol{\beta}_k \in \mathbb{R}^p$, ja

$$\begin{aligned} [\mathrm{AB}]_{ij} &= \boldsymbol{\alpha}_i^T \boldsymbol{b}_j = \sum_{k=1}^{n} [A]_{ik} [B]_{kj} \\ \left[\mathrm{A}\boldsymbol{b}_1, \ldots, \mathrm{A}\boldsymbol{b}_p\right] &= \mathrm{AB} = \begin{bmatrix} \boldsymbol{\alpha}_1^T \mathrm{B} \\ \vdots \\ \boldsymbol{\alpha}_n^T \mathrm{B} \end{bmatrix}. \end{aligned}$$

Olkoon lisäksi $\mathrm{D} \in \mathbb{R}^{n\times n}$ diagonaalimatriisi $\mathrm{D} = \mathrm{diag}(d_1, \ldots, d_n)$.

$$\mathrm{AD} = \left[d_1\boldsymbol{a}_1, \ldots, d_n\boldsymbol{a}_n\right] \qquad \text{ja} \qquad \mathrm{DB} = \begin{bmatrix} d_1\boldsymbol{\beta}_1^T \\ \vdots \\ d_n\boldsymbol{\beta}_n^T \end{bmatrix}.$$

Olkoon vielä $C \in \mathbb{R}^{\ell\times q}$, $D \in \mathbb{R}^{q\times o}$, $E \in \mathbb{R}^{m\times q}$, $F \in \mathbb{R}^{n\times o}$, $G \in \mathbb{R}^{\ell\times n}$ ja $H \in \mathbb{R}^{q\times p}$. Tällöin lohkomatriisien tulo

$$\begin{bmatrix} A & E \\ G & C \end{bmatrix} \begin{bmatrix} B & F \\ H & D \end{bmatrix} = \begin{bmatrix} AB + EH & AF + ED \\ GB + CH & GF + CD \end{bmatrix}$$

Ylläolevat identiteetit voi (ja kannattaa!) tarkistaa suoraan matriisikertolaskun määritelmän avulla.

2.7 Singulaariarvohajotelma ja pseudoinverssi

Olkoon $\mathrm{A} \in \mathbb{R}^{m\times n}$, ja $p := \min\{n, m\} \geq 1$. On olemassa ortogonaaliset matriisit $\mathrm{U} \in \mathbb{R}^{m\times m}$, $\mathrm{V} \in \mathbb{R}^{n\times n}$ ja diagonaalinen matriisi $\Sigma \in \mathbb{R}^{m\times n}$ nollasta poikkeavina alkioinaan $[\Sigma]_{ii} = \sigma_i \geq 0$ jotka on järjestetty $\sigma_1 \geq \cdots \geq \sigma_p \geq 0$, ja joille $\mathrm{A} = \mathrm{U}\Sigma\mathrm{V}^T$. Tämä on A:n singulaariarvohajotelma (*singular value decomposition*, SVD). (vrt. Purmonen: LAG2, Lause 12.7)

- Matriisin $\mathrm{A} \in \mathbb{R}^{m\times n}$ pseudoinverssi on $\mathrm{A}^+ := \mathrm{V}\Sigma^+\mathrm{U}^T \in \mathbb{R}^{n\times m}$, missä $\Sigma^+ \in \mathbb{R}^{n\times m}$ on diagonaalimatriisi, jonka ainoat nollasta poikkeavat alkiot ovat $[\Sigma^+]_{ii} = \frac{1}{\sigma_i}$ kaikilla i joille $\sigma_i > 0$. (vrt. Purmonen: LAG2, Määritelmä 12.9)
- Mikäli $\mathrm{A} \in \mathbb{R}^{n\times n}$ on kääntyvä, niin singulaariarvot $\sigma_1 \geq \ldots \geq \sigma_n > 0$, käänteismatriisi ja pseudoinverssi yhtyvät $A^{-1} = A^+$. Lisäksi $\|\mathrm{A}\| = \sigma_1$ ja $\|\mathrm{A}^{-1}\| = \frac{1}{\sigma_n}$, joten $\mathrm{cond}(\mathrm{A}) = \frac{\sigma_1}{\sigma_n}$.

2.8 Pienimmän neliösumman menetelmä (*least squares*)

- Purmonen: LAG2, luvut 5.7 ja 12.11, jossa käytetään PNS-menetelmästä nimitystä ”Paras ratkaisuapproksimaatio”.
- Olkoon $\mathrm{A} \in \mathbb{R}^{m\times n}$ matriisi ja $\boldsymbol{b} \in \mathbb{R}^n$ vektori. Vektori $\hat{\boldsymbol{x}}$ on yhtälöryhmän $\mathrm{A}\boldsymbol{x} = \boldsymbol{b}$ *pienimmän neliösumman ratkaisu*, jos

  $$\|\mathrm{A}\hat{\boldsymbol{x}} - \boldsymbol{b}\| \leq \|\mathrm{A}\boldsymbol{x} - \boldsymbol{b}\|$$

  kaikilla $\boldsymbol{x} \in \mathbb{R}^n$.
- Mikäli $\mathrm{rank}(\mathrm{A}) = n$, eli A:n sarakkeet ovat lin. riippumattomat, ratkaisu $\hat{\boldsymbol{x}}$ on yksikäsitteinen, ja se saadaan ratkaisemalla normaaliryhmä

  $$\mathrm{A}^T\mathrm{A}\hat{\boldsymbol{x}} = \mathrm{A}^T\boldsymbol{b}.$$

  Tällöin ratkaisu voidaan lausua pseudoinverssin avulla

  $$\hat{\boldsymbol{x}} = \mathrm{A}^+\boldsymbol{b}.$$

2.9 Lyhyesti kompleksisista matriiseista

Seuraavassa joitain käsitteitä kompleksisille matriiseille $A \in \mathbb{C}^{m\times n}$. Kompleksisen matriisin yhteen- ja kertolasku määritellään kuten reaalisillekin matriiseille, käyttäen tietenkin kompleksilukujen yhteen- ja kertolaskuja. Merkitään A:n alkioita $[A]_{ij} = a_{ij}$.

- Kompleksiluvun $z = x + yi \in \mathbb{C}$ *kompleksikonjugaatti* (*liittoluku*) on $\overline{z} = x - yi \in \mathbb{C}$.
- Matriisin A *konjugaattimatriisi* (*liittomatriisi*) $\bar{A}$ saadaan konjugoimalla A alkiottain: $[\bar{A}]_{ij} = \overline{a_{ij}}$.
- Matriisin A *Hermiten transpoosi* A^H on konjugaattitranspoosi $A^H = \bar{A}^T = \overline{A^T}$.
- Neliömatriisi $A \in \mathbb{C}^{n\times n}$ on *Hermiittinen*, jos $A^H = A$.
  - Kun $A \in \mathbb{R}^{n\times n}$, niin $A^H = A^T$, joten A on **Hermiittinen jos ja vain jos se on symmetrinen!**
- Neliömatriisi $A \in \mathbb{C}^{n\times n}$ on *unitaarinen*, jos $A^{-1} = A^H$, eli $A^H A = I = AA^H$.
  - Jos merkitään A:n sarakkeita $A = [a_1, \ldots, a_n]$, selvästi $[A^H A]_{ij} = a_i^H a_j$, joka on *kompleksinen sisätulo* vektorien a_i ja a_j välillä. Toisin sanoen, jos A on unitaarinen, niin A:n sarakkeet ovat *ortonormaalit*.

– Kun $A \in \mathbb{R}^{n \times n}$, niin A on **unitaarinen jos ja vain jos se on ortogonaalinen!**

3 Matlab

- Matlab (”Matrix laboratory”) on MathWorksin kaupallinen numeerisen laskennan ohjelmisto, jota käytetään nykyään laajasti tekniikassa ja luonnontieteissä.
- Alkuperäisen nimenkin mukaan, Matlab keskittyy erityisesti matriisilaskentaan.
- Ohjelmiston etuna nykyään myös monipuoliset tulosten visualisointimahdollisuudet, sekä lukuisat valmiit työkalupakit (*toolbox*) eri alojen tarpeisiin.
- Vaikka Matlab onkin kaupallinen ja suljettu tuote, on olemassa myös ilmainen avoimen lähdekoodin vaihtoehto GNU Octave (`http://www.octave.org/`).
- Octave on jo niin hyvin Matlab-yhteensopiva, että kurssin tehtäviä voi halutessaan aivan hyvin tehdä myös Octavella.
- Muita samankaltaisia ohjelmistoja ovat mm.
  - SciLab (`http://www.scilab.org/`)
  - GNU R (`http://www.r-project.org/`), joka on tehty erityisesti tilastotieteen tarpeisiin.

3.1 Toimintaperiaate

- Matlab on *numeerisen laskennan* ohjelmisto. Sitä voi ajatella ”laskimena”, joka tekee suoraviivaisesti sen (ja vain sen), mitä sen käskee tehdä.
  - Kaikki luvut (muuttujat) tallennetaan muistiin äärellisellä (IEEE 754 kaksoistarkkuus) esitystarkkuudella.
  - Annettuja lausekkeita ei sievennellä, vaan lasketaan raa’asti, noudattaen normaalia laskujärjestystä (sulkeet, tulot, summat ym.). Esimerkki:

```
>> (1/65530)*65530-1
ans =
  -1.1102e-16
```

3.2 Matlab ja Mathematica

- Sekä Matlab että Mathematica ovat tulkkaavia matemaattisia ohjelmistoja.
- Matlab on numeeriseen ja Mathematica symboliseen laskentaan erikoistunut.
- Matlab:in etuna Mathematicaan on, että sillä voi käsitellä melko suuria tietomääriä, ja se on suhteellisen nopea. Mathematica taas lyö Matlab:in symbolisella laskennalla, jota Matlab ei osaa lainkaan[3].
- Matlab:in voi sijoittaa ”oikeiden” ohjelmointikielten ja symbolisten matemaattisten ohjelmistojen ”välimaastoon”.

3.3 Matlab ja GNU R

- Sekä Matlab että GNU R ovat tulkkaavia numeerisen laskennan ohjelmistoja.
- Matlab on erikoistunut matriiseihin: GNU R:ssä hieman kömpelömpi matriisilaskennan syntaksi ja vähemmän valmiita komentoja matriisilaskentaan.

3. Matlab:iinkin saa Maple-toolboxin, jolloin Matlab:ia voi käyttää myös symboliseen laskentaan.

- Matlab on tarkoitettu raskaampaan laskentaan; usein Matlab on kertaluokkaa nopeampi laskentaintensiivisissä tehtävissä kuin GNU R.
- GNU R on erityisesti tilastotieteen tarpeisiin tehty, ja siihen löytyy valmiita paketteja hyvinkin spesifisiin tilastollisiin tehtäviin.
- GNU R:ssä on joitain ”korkeamman tason” ohjelmointiominaisuuksia, jotka ovat joskus hyvin käteviä ja tehokkaita.

3.4 Ohjeet

- `doc`: avaa interaktiivinen ohje
- `lookfor`: etsi hakusanalla funktioita
- `help`: näytä ohje (funktiosta)

```
>> lookfor identity
EYE Identity matrix.
SPEYE  Sparse identity matrix.
>> help eye
 EYE Identity matrix.
     EYE(N) is the N-by-N identity matrix.
...
>> a=eye(2)
a =
     1     0
     0     1
```

3.5 Muuttujat ja perustoimintoja

- Perustietotyyppinä Matlab:issa on ”matriisi”. Skalaari on 1×1-matriisi, vektorit joko $N \times 1$ (pysty) tai $1 \times N$ (vaaka).
- Muuttujien (ja funktioiden) nimet alkavat kirjaimella (`a-zA-Z`, isot ja pienet kirjaimet ovat eri merkkejä), muita sallittuja merkkejä ovat numerot (`0-9`) ja alaviiva (_)
- Syntaksista:
  - Yhtäsuuruusmerkki `=` on sijoitusmerkki, vasemmalla puolella olevaan muuttujaan sijoitetaan oikean puolen arvo.
  - Rivin vaihto `<Enter>` on komentoerotin. Kun komentokehotteessa painaa `<Enter>`, kyseisen rivin komento suoritetaan.
  - Puolipiste `;` ja `,` toimivat myös komentoerottimina. Mikäli komennon perään laitetaan puolipiste, lopputulos jätetään näyttämättä. **Huom: Muista lisätä puolipiste komennon perään aina, kun käsittelet suuria matriiseja!**
  - Hakasulkeita `[ ]` käytetään matriiseja syötettäessä, lohkomatriiseja koottaessa, sekä sellaisia funktioita kutsuttaessa, joista tulee useampi kuin yksi ulostulo.
  - Matriiseja syöttäessä `,` ja `<Space>` ovat alkion erottimia, ja `;` tai `<Enter>` rivin vaihto.
  - Kaikki tavalliset laskutoimitukset `+`, `-`, `*`, `/`, \, `^` ovat *matriisilaskutoimituksia*.
- Esimerkki: matriisien/vektorien luonti ja perusoperaatioita.

```
>> a=[1 3; 2,4]
a =
     1     3
     2     4
>> x = [5;6]
x =
     5
     6
>> x2=a*x
x2 =
    23
    34
>> x2*x
??? Error using ==> mtimes
Inner matrix dimensions must agree.
>> x2+x
ans =
    28
    40
>> pi*a
ans =
    3.1416    9.4248
    6.2832   12.5664
```

- Lisää syntaksista:
  - *Komponenteittaiset* laskutoimitukset saadaan lisäämällä operaattorin eteen piste: `.*`, `./`, `.\`, `.^`.
  - Sulkeita `( )` käytetään matriisien ja vektoreiden indeksoinnissa sekä funktiokutsuissa.
  - Yksittäinen lainausmerkki `'` on Hermiten transpoosi ja `.'` transpoosi (reaalisilla matriiseilla sama asia!).
  - Yksittäisiä lainausmerkkejä `' '` käytetään myös merkkijonojen syöttämiseen.
  - Kompleksiluvut voidaan antaa käyttäen imaginaariyksikköä (`i` tai `j`).
  - Desimaalieroitin on piste `.`
  - Prosenttimerkki `%` on kommenttimerkki: loput rivistä jätetään huomiotta.
  - Rivin voi katkaista laittamalla rivin loppuun kolme pistettä: `...`
- Matemaattiset alkeisfunktiot: `sin`, `cos`, `tan`, `asin`, `log`, `exp`, `sqrt`, `abs`, ... toimivat myös kompleksiarvoisille suureille, ja vektoreille/matriiseille alkioittain. (lisää: `help elfun`)

```
>> y=sin(x); % Lasketaan sini x:n alkioista
>> y'
ans =
   -0.9589   -0.2794
>> x2.*x
ans =
   115
   204
>> x2'*x
ans =
   319
>> a*[1+3i; 2.5]
ans =
    8.5000 + 3.0000i
   12.0000 + 6.0000i
```

- Matriisien indeksoinnista: matriisista voidaan poimia alkioita sulkeilla. **Huom: Matlab:issa indeksointi alkaa ykkösestä, ei nollasta!**
- Yksittäisten indeksien lisäksi on mahdollista antaa useampi indeksi, ja poimia esim. sarakkeita tai rivejä, tai alimatriiseja.
- Kaksoispisteellä `:` voi luoda vektorin, jossa on peräkkäisiä kokonaislukuja. Yksistään `:` tarkoittaa "kaikkia" rivin tai sarakkeen alkioita.

```
>> a
a =
     1     4     7
     2     5     8
     3     6     9
>> a(1,2)
ans =
     4
>> a(1,:)
ans =
     1     4    7
>> a(:,2:end)
ans =
     4     7
     5     8
     6     9
>> b = 1:3
b =
     1     2     3
>> a(b, [1 3])
ans =
     1     7
     2     8
     3     9
>> a(1:2,1:2) = eye(2)
a =
     1     0     7
     0     1     8
     3     6     9
```

3.6 Matriisien luontifunktioita

- `diag`, `eye`, `ones`, `zeros`, `rand`: Luo diagonaali-, identiteetti-, ykkös-, nolla-, tai satunnaislukuja välillä $(0, 1)$ sisältävä matriisi. (`help elmat`)

```
>> a=rand(3)
a =
    0.9501    0.4860    0.4565
    0.2311    0.8913    0.0185
    0.6068    0.7621    0.8214
>> b=ones(3,1)
b =
     1
     1
     1
>> c=zeros(1,3)
c =
     0     0     0
>> d = [a b;c 42]
d =
    0.9501    0.4860    0.4565    1.0000
    0.2311    0.8913    0.0185    1.0000
    0.6068    0.7621    0.8214    1.0000
         0         0         0   42.0000
>> diag(d)
ans =
    0.9501
    0.8913
    0.8214
   42.0000
>> e = diag([pi exp(1)])
e =
    3.1416         0
         0    2.7183
```

3.7 Omat funktiot ja M-tiedostot

- MATLAB:issa on helppo tehdä omia funktioita.
- Perusperiaate on, että kutakin funktiota varten tehdään vastaavan niminen tiedosto. Esimerkiksi halutessamme tehdä funktion `foo`, luomme tiedoston `foo.m`.

```
>> edit foo
(kirjoita funktio editorissa ja tallenna se)
>> z = foo(15,4)
z =
   15.5242
```

- Tiedoston `foo.m` sisältö:

```
function d = foo(x, y)
d = sqrt(x^2+y^2);
```

- Funktioiden lisäksi MATLAB:issa on mahdollista tehdä ns. ”M-tiedostoja” (M-file), jotka sisältävät vain listan ajettavia komentoja.

```
N = 3;
a = ones(N,1);
x = [a, rand(N,N)]
```

3.8 Silmukat, ehtolauseet ym.

- MATLAB:issa on muistakin ohjelmointikielistä tutut ehtolauseet ja silmukat.
  - `if`-ehtolause suorittaa lausekkeen sisäisen arvon, mikäli ehto on tosi[4].
  - `for`-silmukka käy läpi annetun (vaaka)vektorin alkiot.
  - `while`-silmukka jatkuu niin pitkään, kun ehtolausekkeen arvo on tosi.
- Ehtolauseissa voi käyttää mm. relaatioita `<`, `<=`, `==`, `~=`, `&`, `|` (`help relop`).

4. Ehtolauseke on ”tosi”, mikäli se ei ole nolla.

```
N = 6;
a = eye(N);
for k=1:N
  j = 1;
  while j<k
    if mod(k,j) == 0
      a(k,j) = 1;
    end
    j = j+1;
  end
end
```

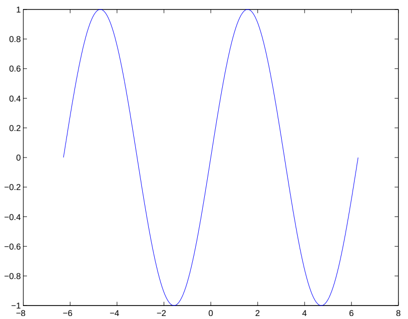
- Loogiset relaatiot toimivat myös vektori/matriisiargumenteille

```
>> a = 1:5, b = rand(1,5)*5
a =
     1     2     3     4     5
b =
    1.0138    0.9936    3.0190    1.3609    0.9941
>> a<b
ans =
     1     0     1     0     0
```

3.9 Grafiikka

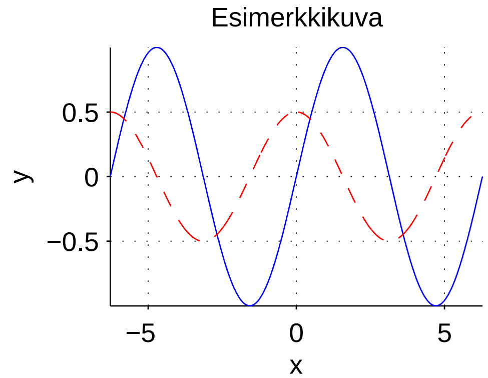
- Yksi MATLAB:in vahvuusalueista ovat sen visualisointiominaisuudet.
- MATLAB:issa on valmiina erilaisia visualisointifunktioita, mutta jätetty myös mahdollisuus säätää lähes kaikkia visualisoinnin yksityiskohtia.
- Grafiikankin perusideana on numeriikka: sen sijaan että käskettäisiin esimerkiksi ”Piirrä funktio $\sin(x)$ kun $-2\pi \leq x \leq 2\pi$”, lasketaankin funktion (liki)arvot tietyissä pisteissä ko. välillä, ja käsketään MATLAB:in vain visualisoida saadut *numerot*:

```
>> x = linspace(-2*pi,2*pi,1000);
>> y = sin(x);
>> plot(x,y)
```



- Lisätään samaan kuvaan toinen käyrä, ja kustomoidaan kuvan asetuksia:

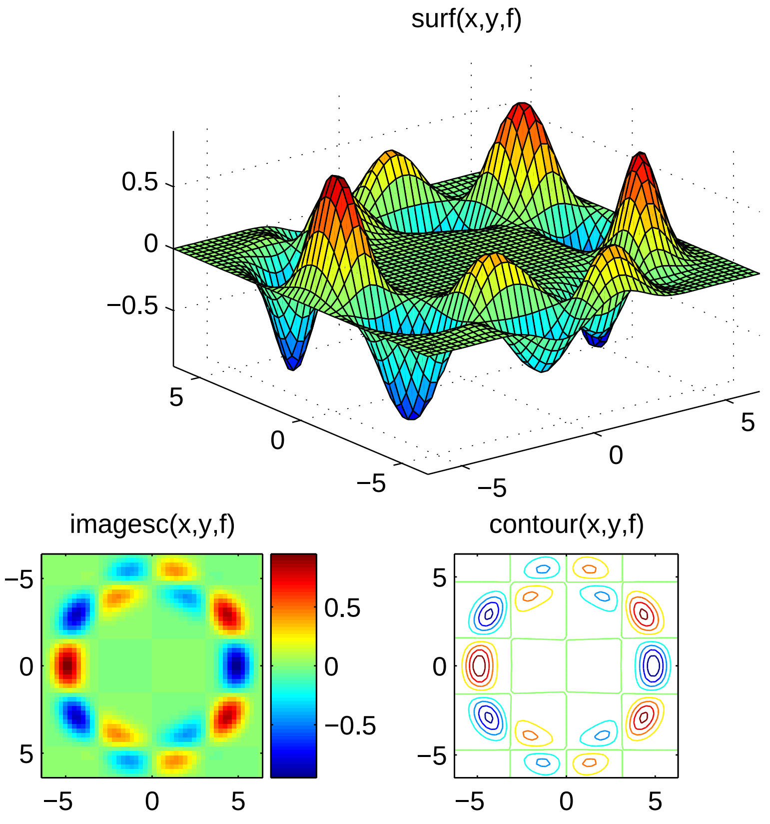
```
>> hold on
>> plot(x, 0.5*cos(x), 'r--')
>> axis tight, grid on, box off
>> xlabel 'x'; ylabel 'y'; title 'Esimerkkikuva'
>> set(gca,'tickdir','out');
>> set(gcf,'paperposition',[0 0 7 5]);
>> print -depsc example.eps
```



- Kahden muuttujan funktion visualisointia:

```
>> x = linspace(-2*pi,2*pi,100); y=x;
>> [xx,yy] = meshgrid(x,y);
>> f =  sin(xx).*cos(yy) ...
.*exp(-(sqrt(xx.^2+yy.^2)-5).^2);
>> surf(x,y,f);
```

```
>> imagesc(x,y,f); colorbar
>> contour(x,y,f)
```



3.10 (*) Muita tietotyyppejä

- Matriisien alkiot voivat olla liukulukujen lisäksi merkkejä, jolloin vektorit voivat sisältää merkkijonoja.

```
>> a = 'foo'
a =
foo
>> a(1)
ans =
f
```

- MATLAB osaa käsitellä myös ns. harvoja (*sparse*) matriiseita.

```
>> s = sparse([1 0 0; 0 0 2; 0 3 0])
s =
   (1,1)        1
   (3,2)        3
   (2,3)        2
```

- Matriisien lisäksi MATLAB:illa voi luoda useampiulotteisia taulukoita.

```
>> a = zeros(2,2,2);
>> a(:,:,1) = eye(2); a(:,:,2) = rand(2)
a(:,:,1) =
     1     0
     0     1
a(:,:,2) =
    0.0153    0.4451
    0.7468    0.9318
```

- Solutaulukon (*cell array*) alkiot voivat olla mitä tahansa tietotyyppiä, esim. matriiseja, merkkijonoja tai solutaulukoita. Solutaulukoita luodaan ja indeksoidaan kaarisuluilla { }.

```
>> a = {'foo', rand(2)}
a =
    'foo'    [2x2 double]
>> a{1}
ans =
foo
>> a{3} = a
a =
    'foo'    [2x2 double]    {1x2 cell}
```

- Rakennetaulun (*struct*) kukin alkio sisältää joukon kenttiä, jotka voivat saada arvokseen mitä tahansa. Kenttiä indeksoidaan pistenotaatiolla:

```
>> a = struct('sat', rand(2), 'id', eye(2))
a =
    sat: [2x2 double]
     id: [2x2 double]
```

```
>> a.id
ans =
     1     0
     0     1
```

- Myös rakennetauluja voi olla taulukossa (*struct array*).

```
>> a(2).sat = rand(2);
>> a(1).sat
ans =
    0.2026    0.8381
    0.6721    0.0196
>> a(2).sat
ans =
    0.6813    0.8318
    0.3795    0.5028
>> a(2).id
ans =
     []
```

3.11 (*) Datan lataus ja tallennus

- Muuttujia voi tallentaa tiedostoon komennolla `save`, ja ladata niitä komennolla `load`

```
>> a=1:5;
>> save foo
>> clear
>> a
??? Undefined function or variable 'a'.

>> load foo
>> a
a =
     1     2     3     4     5
```

- Matlab:issa on myös mahdollista ladata muista ohjelmista tiedostoja. Esimerkiksi useimmista taulukko-ohjelmista saa tallennettua datan ns. CSV-muodossa (Comma Separated Values). Tällaista dataa voi lukea Matlab:iin komennoilla `csvread` tai `dlmread`.
- Binääristä dataakin voi käsitellä, kts. esim. `help fread` ja `help fwrite`.

3.12 (*) Sudenkuoppia

- Matlab:issa voi määritellä itse uudelleen lähes minkä tahansa muuttujan tai funktion.

```
>> cos(2*pi)
ans =
     1
>> pi=3;
>> cos(2*pi)
ans =
    0.9602
>> cos = 0.96;
>> cos(2*pi)
??? Index exceeds matrix dimensions.

>> clear pi
>> cos(2*pi)
??? Subscript indices must either be real
positive integers or logicals.

>> clear
>> cos(2*pi)
ans =
     1
```

- Syötettäessä matriiseja, tyhjä merkki yksinään on alkioiden erotin.

```
>> a=[1;1 +3i]
??? Error using ==> vertcat
All rows in the bracketed expression must have
the same number of columns.
```

```
>> a = [1; 1+3i]
a =
   1.0000
   1.0000 + 3.0000i
```

- Funktioita, jotka saavat argumentikseen merkkijonoja, voi kutsua myös ilman sulkeita:

```
>> cos pi
??? Undefined function or method 'cos' for input
arguments of type 'char'.

>> clear(pi)
??? Error using ==> clear
Argument must contain a string.

>> clear('pi')
>> clear pi
```

3.13 (*) Omat laajennokset

- Matlab:iin voi kirjoittaa helpohkosti omia ns. MEX-laajennoksia C/C++ ja Fortran-kielillä.
- Laajennoksia voi tarvita laskennan nopeuttamiseksi, tai vaikkapa jonkin mittauslaitteiston ohjaamiseksi Matlab:ista käsin.