# Numeerinen analyysi

Jukka Tuomela

23. marraskuuta 2012

# Sisältö


**1 Johdantoa** **11**

**2 Yhtälön ratkaisu** **13**
2.1 Juurien laskeminen . . . . . 13
2.2 Yleinen tapaus . . . . . 14
2.2.1 Kiintopisteiteraatio . . . . . 14
2.2.2 Newtonin menetelmä . . . . . 18
2.3 Algoritmin pysäyttäminen . . . . . 22
2.4 Erityiskysymyksiä . . . . . 23
2.4.1 Kompleksiset nollakohdat . . . . . 23
2.4.2 Polynomin kaikki nollakohdat . . . . . 24
2.4.3 Reaaliset nollakohdat . . . . . 25
2.5 Harjoitustehtäviä . . . . . 27

**3 Yhtälösysteemit** **29**
3.1 Lineaarinen tapaus . . . . . 31
3.2 Yleinen kiintopiste . . . . . 34
3.3 Newtonin menetelmä . . . . . 36
3.4 Harjoitustehtäviä . . . . . 39

**4 Käyrät** **41**
4.1 Approksimointi . . . . . 41
4.2 Interpolointi . . . . . 48
4.3 Splinit . . . . . 55
4.4 Yleisen tasokäyrän interpolointi . . . . . 62
4.5 Bézier-käyrät . . . . . 64
4.6 Harjoitustehtäviä . . . . . 67

**5 Numeerinen integrointi** **69**
5.1 Yksiulotteinen tapaus . . . . . 69
5.1.1 Bernstein ja Newton-Cotes . . . . . 69
5.1.2 Gaussin menetelmät . . . . . 74
5.1.3 Integroinnin jakaminen osaväleihin . . . . . 76
5.2 Virhearvio . . . . . 77
5.3 Muutamia erityiskysymyksiä . . . . . 79

5.3.1 Jaksolliset funktiot . . . 79
5.3.2 Singulaarisuudet . . . 80
5.3.3 Moniulotteinen integraali . . . 81
5.4 Harjoitustehtäviä . . . 82

**6 Differentiaaliyhtälöt 83**
6.1 Taustaa . . . 83
6.2 Kaksi yksinkertaista menetelmää . . . 84
6.3 Erilaisia menetelmäluokkia . . . 88
6.4 Runge-Kutta-menetelmät . . . 89
6.5 Moniaskelmenetelmät . . . 92
6.6 Erityisiä tehtäväluokkia ja muita kommentteja . . . 98
6.6.1 Kankeat tehtävät . . . 98
6.6.2 Difyhtälösysteemit . . . 101
6.6.3 Käytännön toteutus . . . 101
6.6.4 Hamiltonin systeemit . . . 102
6.7 Harjoitustehtäviä . . . 102

**A Hyödyllisiä asioita 105**
A.1 Liukuluvut . . . 105
A.2 Polynomit . . . 106
A.2.1 Yhden muuttujan polynomit . . . 106
A.2.2 Yleiset polynomit . . . 109
A.3 $\mathbb{R}^n$ . . . 110
A.4 Differentiaalilaskentaa . . . 113
A.4.1 Merkintöjä . . . 113
A.4.2 Taylorin lause . . . 114
A.4.3 Vektoriarvoiset funktiot . . . 115
A.5 Differenssiyhtälö . . . 115
A.6 Harjoitustehtäviä . . . 117

**Kirjallisuutta 117**

# Kuvat


2.1 Pascalin kolmio on Kiinassa Yang Huin kolmio. . . . . 15
2.2 Esimerkin 2.2 funktio $g$ tapauksissa $\gamma = 0.8$ ja $\gamma = 2.1$. . . . . 16
2.3 Esimerkin 2.2 iteraatio tapauksissa $\gamma = 0.8$, $\gamma = 1.1$, $\gamma = 1.35$ ja $\gamma = 2.1$. Kaikissa on käytetty alkuarvoa $x_0 = -2$. . . . . 16
2.4 Newtonin menetelmän ensimmäinen askel esimerkin 2.4 tapauksessa. . 19
2.5 Newtonin menetelmän suppeneminen kiintopisteiteraatioon verrattuna esimerkin 2.4 tapauksessa. . . . . 20
2.6 Newtonin menetelmä suppenee hitaasti kaksinkertaisen nollakohdan tapauksessa. Yhtenäinen viiva antaa teoreettisen suppenemisnopeuden $\sim 2^{-k}$. 21
2.7 Newtonin menetelmä esimerkissä 2.7. . . . . 22
2.8 Newtonin menetelmän suppeneminen kompleksitasossa. . . . . 24

3.1 Esimerkin 3.1 tapaus. . . . . 30
3.2 Vasemmalla kuvausten $f_1$ ja $f_2$ merkit esimerkin 3.1 erään ratkaisun ympäritössä. Oikealla kuvausten $\tilde{f}_1$ ja $\tilde{f}_2$ merkit eräässä tason osassa. Merkit muodostavat samanlaisen kuvion molemmissa tapauksissa, mutta jälkimmäisessä tapauksessa systeemillä ei ole ratkaisua kyseisellä alueella. . . . . 31
3.3 Kiintopisteiteraation suppeneminen esimerkin 3.1 tapauksessa kahdella eri alkuarvolla. . . . . 32
3.4 Vasemmalla esimerkin 3.3 matriisi $A$ on kuvannut "yksikkökiekon" $|x|_1 \leq 1$ ruskeaksi alueeksi. Oikealla ruskea alue on "kiekon" $|x|_\infty \leq 1$ kuvajoukko. . . . . 33
3.5 Esimerkin 3.3 matriisin $T$ normit. $\|T\|_1$ (vihreällä) ja $\|T\|_\infty$ (sinisellä) $\alpha$:n funktiona. . . . . 33
3.6 Newtonin menetelmän suppeneminen. Vihreällä piirretyillä käyrillä $x_2 = \pm x_1$ Newtonin kuvaus $N$ ei ole määritelty. . . . . 38

4.1 Vasemmalla approksimaatiot $T_1, \ldots, T_4$ välillä $[0.5, 1.5]$ ja oikealla välillä $[0, 4]$. Oikealla on myös $T_{20}$. . . . . 42
4.2 Bernsteinin polynomien avulla saadut approksimaatiot esimerkin 4.1 tapauksessa. Polynomien asteluvut ovat 5, 10, 15 ja 20. . . . . 45
4.3 Legendren polynomien avulla saadut approksimaatiot esimerkin 4.1 tapauksessa. Polynomien asteluvut ovat 2, 5 ja 8. . . . . 48

4.4 Funktio ja interpoloivat polynomit joitten aste on 2, 4 ja 8. 8. asteen polynomi antaa jo varsin hyvän approksimaation paitsi välin päissä. . 50
4.5 Esimerkin 4.4 interpolointivirheet sekä lineaarisen että kolmannen asteen interpoloinnin tapauksessa. . . . . 53
4.6 Tshebyshevin polynomin nollakohtien avulla saatu approksimaatio esimerkin 4.1 tapauksessa verrattuna tasaväliseen interpolointiin. Polynomien asteluvut on 8. Oikealla vastaavat virheet. Tshebyshevin approksimaatio on huomattavasti parempi välin päissä. . . . . 54
4.7 Esimerkin 4.5 virhe sekä tasavälisessä (sininen) että Tshebyshev-interpoloinnissa (vihreä). Molemmissa tapauksissa virheellä on selvä piikki kohdassa $x = 1$. 55
4.8 Esimerkin 4.6 funktio ja sen 24 ja 30 asteiset interpoloivat polynomit. 56
4.9 Kaksi pistejoukkoa ja niitä vastaavat interpoloivat polynomit. Huomaa, että oikealla on välillä $[0, 1]$ kaksi interpolointipistettä hyvin lähellä toisiaan. . . . . 56
4.10 Lineaarinen ja neliöllinen splini esimerkin 4.8 tapauksessa. . . . . 58
4.11 Esimerkin 4.7 dataa vastaavat kuutiosplinit. . . . . 61
4.12 ”Alkuperäinen” splini. . . . . 63
4.13 Esimerkin 4.11 pisteet ja interpoloivat käyrät. . . . . 64
4.14 Vasemmalla kolmen pisteen määrittämä Bézier-käyrä, oikealla 11 pistettä. Sinisellä on piirretty pisteitten määrittämän konveksin peitteen reuna. . . . . 66
4.15 De Casteljaun algoritmin eteneminen. Ensimmäisessä vaiheessa muodostetaan siniset suorat, sitten punaiset toisen asteen käyrät, ja lopuksi kaikkien pisteitten määrittämä Bézier-käyrä. . . . . 66
4.16 Erityyppisiä jousia. . . . . 67

5.1 Vasemmalla laskettu MAPLEN oletustarkkudella, jossa käytetään 10 desimaalia. Oikealla on laskettu 20 desimaalilla. Kuvaan on myös piirretty kaavoja (5.4) vastaavat arviot virheen ylärajalle. . . . . 79
5.2 jotain. . . . . 80

6.1 Suuntakenttä ja kaksi askelta Eulerin menetelmällä. . . . . 86
6.2 Esimerkin 6.1 ratkaisu ja 2 numeerista ratkaisua käyttäen askelpituuksia $h = 0.2$ ja $h = 0.1$. . . . . 87
6.3 Esimerkin 6.1 ratkaisu ja numeerinen ratkaisu käyttäen trapetsimenetelmää ja askelpituutta $h = 0.2$. Selvästi näinkin isolla askeleella saadaan huomattavasti tarkempi tulos kuin Eulerin menetelmällä. . . . . 87
6.4 Esimerkin 6.4 numeerinen ratkaisu verrattuna tarkkaan ratkaisuun vasemmalla ja virheet oikealla. Euler: vihreä, Heun: punainen, klassinen RK: kellanruskea. . . . . 93
6.5 Esimerkin 6.6 menetelmällä saadut ratkaisut kolmella eri askelpituudella. 96
6.6 Menetelmällä (6.5) saatuja numeerisia ratkaisuja esimerkin 6.7 tapauksessa eri aika-askelilla. . . . . 98
6.7 Esimerkin 6.8 tehtävän ratkaisu klassisella RK-menetelmällä kahdella eri aika-askeleella. . . . . 99

6.8 Esimerkin 6.8 tehtävän ratkaisu BDF2-menetelmällä kahdella eri aika-askeleella. . . . . 101
6.9 Tehtävän (6.7) ratkaisu sekä Eulerin menetelmällä että klassisella RK-menetelmällä. . . . . 103

A.1 Kaavalla $\mathcal{I}_n = e - n\mathcal{I}_{n-1}$ lasketut arvot. Sininen: mantissan pituus 5, keltainen: mantissan pituus 10, punainen: mantissan pituus 15. . . . . 106
A.2 Legendren polynomit $L_2$, $L_3$, $L_4$ ja $L_5$. . . . . 108
A.3 Tshebyshevin polynomit $C_2$, $C_3$, $C_4$ ja $C_5$. . . . . 109
A.4 Bernsteinin polynomit $B_0^3$, $B_1^3$, $B_2^3$ ja $B_3^3$. . . . . 110

# Algoritmit


1 Neliöjuuri . . . . . 13
2 Kiintopisteiteraatio . . . . . 15
3 Newtonin menetelmä . . . . . 18
4 Kiintopisteiteraatio, versio 2 . . . . . 23
5 Suurin yhteinen tekijä . . . . . 27
6 Newtonin menetelmä $\mathbb{R}^n$:ssä . . . . . 37
7 De Casteljaun algoritmi . . . . . 65
8 Eulerin menetelmä . . . . . 86

# Luku 1

# Johdantoa

*It is the mark of an educated mind*
*to look for precision in each class of things*
*just so far as the nature of the subject admits.*

Aristoteles: Nikomakhoksen etiikka, I.3[1]

*Numerical analysis is the study of algorithms*
*for the problems of continuous mathematics.*

N. L. Trefethen [20, s. 323]

Tarkastellaan seuraavassa neljää numeerisen analyysin perustehtävää:

- yhtälön ratkaisu, tai yleisemmin yhtälöryhmien ratkaisu
- ”sopivan” käyrän piirtäminen.
- numeerinen integrointi
- differentiaaliyhtälön numeerinen ratkaisu

Seuraavissa numeerisen analyysin perusteoksissa on käsitelty samoja asioita kuin tällä kurssilla [18].

[1] http://www.constitution.org/ari/ethic_00.htm

# Luku 2

# Yhtälön ratkaisu

## 2.1 Juurien laskeminen

Lähdetään liikkeelle yksinkertaisesta tapauksesta: miten lasketaan luvun neliöjuuri? Tämähän on sama asia kuin laskea nollakohta polynomille $f(x) = x^2 - a$. Jo antiikin aikoina tiedettiin, että seuraavalla algoritmilla saadaan nopeasti hyvä approksimaatio.

**Algoritmi 1** NELIÖJUURI

**Input:** annetaan jokin $a > 0$ ja valitaan jokin $x_0 > 0$
**Output:** arvio luvulle $\sqrt{a}$
  **loop**
    $$x_{k+1} = \tfrac{1}{2}\Big(x_k + \tfrac{a}{x_k}\Big)$$
  **end loop**

Huomaa, että tämähän on päättymätön silmukka, siis tuotetaan jono $\{x_k\}_k$ ja toivotaan, että $x_k \to \sqrt{a}$ kun $k \to \infty$. Käytännön laskuissa pitäisi algoritmiin siis lisätä jokin suppenemiskriteeri joka pysäyttäisi algoritmin. Palataan tähän kysymykseen myöhemmin.

Mutta miten edellä mainittu algoritmi löydettiin? Ehkäpä huomattiin, että jos $x_k < \sqrt{a}$, niin $a/x_k > \sqrt{a}$ joten niitten keskiarvon luulisi olevan parempi approksimaatio. Mutta onko näin?

$$|x_{k+1} - \sqrt{a}| = \big|\tfrac{1}{2}\Big(x_k + \frac{a}{x_k}\Big) - \sqrt{a}\,\big| = \tfrac{1}{2}\Big|1 - \tfrac{\sqrt{a}}{x_k}\Big|\,\big|x_k - \sqrt{a}\big|$$

Siis virhe pienenee, jos

$$\tfrac{1}{2}\Big|1 - \tfrac{\sqrt{a}}{x_k}\Big| < 1 \quad \Leftrightarrow \quad x_k > \frac{\sqrt{a}}{3}$$

Luonnollisesti on helppo valita sellainen alkuarvo, että $x_0 > \sqrt{a}/3$. Mutta edes tämä ei ole tarpeen. Helposti nähdään, että jos valitaan mikä tahansa $x_0 > 0$, niin $x_1 > \sqrt{a}/3$. Siis lähtien mistä tahansa alkuarvosta virhe rupeaa pienenemään viimestään toisen iteraatiokierroksen jälkeen ja siis $x_k \to \sqrt{a}$ kun $k \to \infty$.

**Esimerkki 2.1.** Jos $a = 7$ ja $x_0 = 4$, niin saadaan

$$\begin{aligned} x_1 &= 2.875 \\ x_2 &= 2.65489\ldots \\ x_3 &= 2.645767\ldots \\ x_4 &= 2.64575131111\ldots \\ x_5 &= 2.64575131106459059\ldots \end{aligned}$$

Suppeneminen on erittäin nopeaa: iteraatin $x_5$ kaikki näytetyt desimaalit ovat oikeita! ✯

Hyvin varhain tämä algoritmi osattiin myös yleistää tapaukseen $f(x) = x^n - a$. Valitaan $n = 4$ ja oletetaan, että on jokin approksimaatio $\hat{x}$ ja haluttaisiin parantaa tätä siten, että $\hat{x} + h$ olisi lähempänä oikeaa ratkaisua. Siis halutaan löytää $h$ siten, että

$$(\hat{x} + h)^4 = \hat{x}^4 + 4\hat{x}^3 h + 6\hat{x}^2 h^2 + 4\hat{x}h^3 + h^4 = a$$

Mutta jos $\hat{x}$ on "lähellä" oikeaa arvoa $a^{1/4}$, niin $h$ on "pieni" joten

$$\hat{x}^4 + 4\hat{x}^3 h \approx a \quad \text{joten valitaan} \quad h = (a - \hat{x}^4)/4\hat{x}^3$$

Tämän idean takia on luonnollista, että tässä yhteydessä löydettiin myös "Pascalin kolmio", jo ainakin 500 vuotta ennen Pascalia, kuva 2.1. Saadaan siis iteraatio

$$x_{k+1} = x_k + (a - x_k^4)/4x_k^3 \tag{2.1}$$

## 2.2 Yleinen tapaus

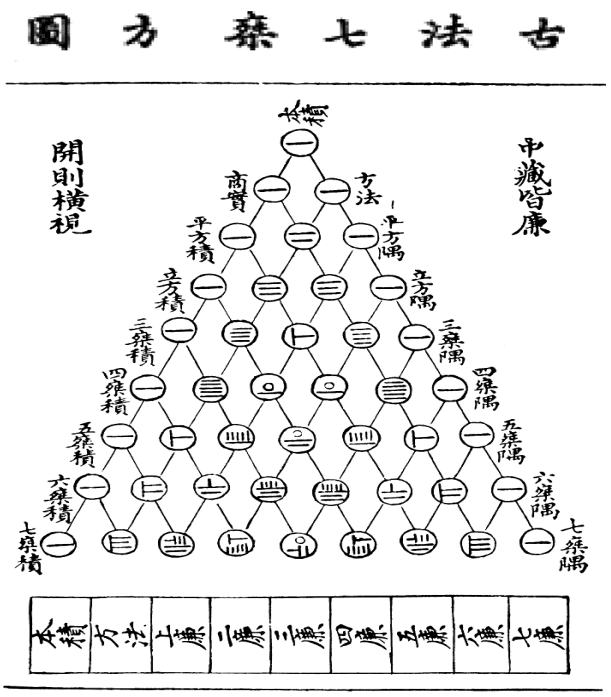
### 2.2.1 Kiintopisteiteraatio

Olkoon sitten $f$ jokin jatkuva funktio, jonka nollakohdat haluttaisiin laskea. Nollakohtien etsimisessä ensin kannattaa yrittää saada edes jonkinlainen käsitys siitä missä ratkaisu mahdollisesti on. Skalaaritapauksessa tämä on varsin helppoa kun voidaan piirtää kuvaaja ja muistetaan, että jos $a < b$ ja $f(a)f(b) < 0$, niin $f$:llä on nollakohta välillä $[a, b]$.

Eräs tapa yrittää etsiä nollakohta perustuu kiintopisteisiin:

**Määritelmä 2.1.** *Olkoon $g : \mathbb{R} \to \mathbb{R}$. Piste $z$ on $g$:n* kiintopiste*, jos $g(z) = z$.*

Olkoon sitten $g(x) = x + f(x)$. Selvästi nyt $f(x) = 0$ jos ja vain jos $g(x) = x$, joten $g$:n kiintopisteet ovat $f$:n nollakohdat. Geometrisesti siis nollakohdan etsiminen muutettiin käyrien $y = x$ ja $y = g(x)$ leikkauspisteen etsimiseksi. Tästä saadaan




Kuva 2.1: Pascalin kolmio on Kiinassa Yang Huin kolmio.

**Algoritmi 2** KIINTOPISTEITERAATIO

**Input:** valitaan jokin $x_0$ ja annetaan funktio $g$
**Output:** luku $\bar{x}$ siten että $g(\bar{x}) \approx \bar{x}$

**loop**

$$x_{k+1} = g(x_k)$$

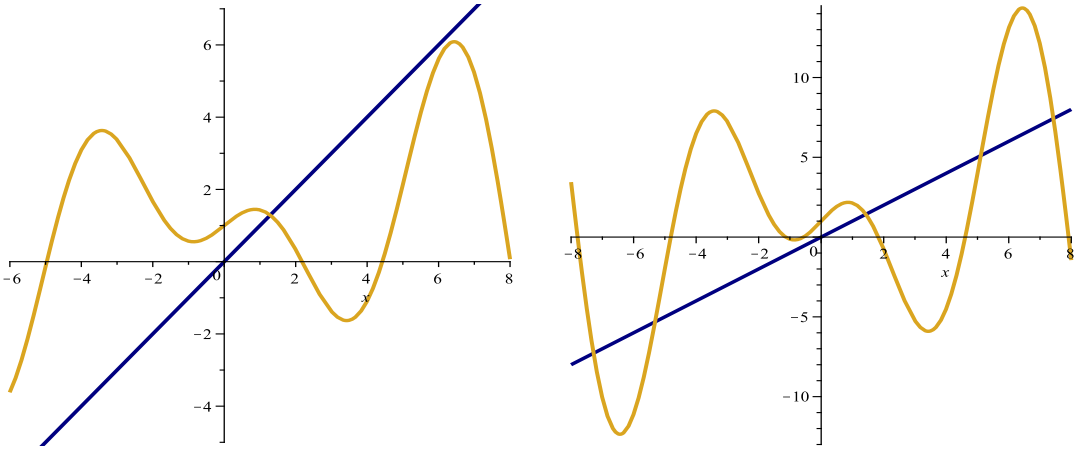
**end loop**

Esimerkkejä tarkastelemalla havaitaan, että kiintopisteiteraatio ei valitettavasti aina suppene.

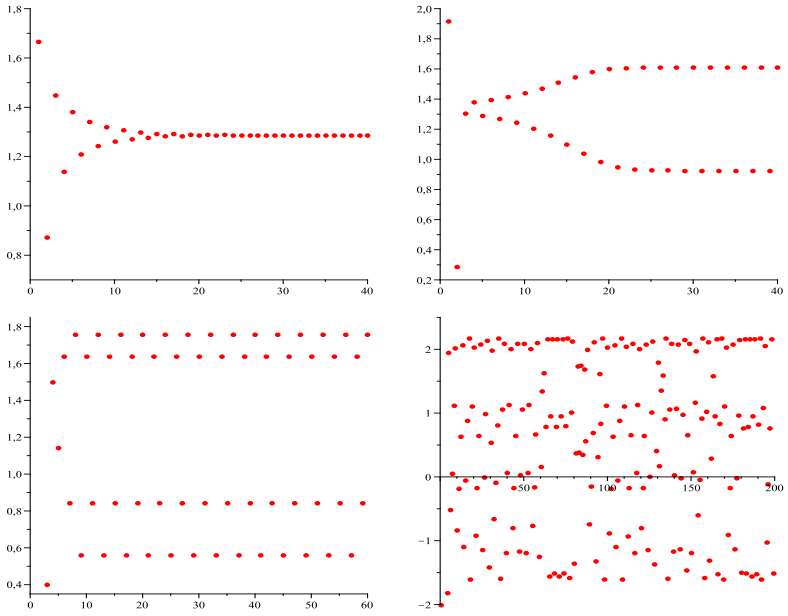
**Esimerkki 2.2.** Olkoon $g(x) = \gamma\, x \cos(x) + 1$, missä $\gamma$ on jokin parametri. Tarkastellaan parametrin arvoja $\gamma = 0.8$, 1.1, 1.35 ja 2.1. Kaikilla näillä arvoilla $g$:llä on yksi kiintopiste välillä $[1, 2]$, kuva 2.2. Kun $\gamma = 0.8$, niin iteraatio suppenee, kuva 2.3. Sen sijaan kun $\gamma$ kasvaa tapahtuu outoja. Kun $\gamma = 1.1$ iterointi suppenee kohti 2-jaksollista jonoa. Kun $\gamma = 1.35$ saadaan puolestaan 4-jaksollinen jono. Lopulta kun $\gamma = 2.1$ jono vaikuttaa kaoottiselta. Jono on kuitenkin siinä mielessä stabiili, että kaikki iteraatit pysyvät välillä $[-2, 3]$. ✯

Seuraava tulos antaa kriteetin kiintopisteiteraation suppenemiselle.

**Lemma 2.1.** *Olkoon $x_*$ kiintopiste ja $|g'(x_*)| < 1$. Tällöin kiintopisteiteraatio suppenee, jos alkuarvaus $x_0$ on riittävän lähellä kiintopistettä $x_*$.*




Kuva 2.2: Esimerkin 2.2 funktio $g$ tapauksissa $\gamma = 0.8$ ja $\gamma = 2.1$.




Kuva 2.3: Esimerkin 2.2 iteraatio tapauksissa $\gamma = 0.8$, $\gamma = 1.1$, $\gamma = 1.35$ ja $\gamma = 2.1$. Kaikissa on käytetty alkuarvoa $x_0 = -2$.

*Todistus.* Jatkuvuuden perusteella on olemassa $\varepsilon > 0$ siten, että $|g'(x)| \leq \mu < 1$ kaikilla $x \in I_\varepsilon = [x_* - \varepsilon, x_* + \varepsilon]$. Siis jos $x_k \in I_\varepsilon$, niin

$$\begin{aligned} |x_{k+1} - x_*| &= |g(x_k) - x_*| = |g(x_k) - g(x_*)| \\ &= \Big| \int_{x_*}^{x_k} g'(s)ds \Big| \leq \int_{x_*}^{x_k} |g'(s)|ds \leq \mu |x_k - x_*| \end{aligned}$$

Edelleen jos $x_0 \in I_\varepsilon$, niin induktiolla saadaan

$$|x_k - x_*| \leq \mu^k |x_0 - x_*|$$

✧

Luonnollisesti kiintopiste $x_*$ on tuntematon, joten ehtoa $|g'(x_*)| < 1$ ei voi suoraan tarkistaa. Käytännön testin saa kun tarkastelee lemman todistusta. Siinähän itse asiassa osoitettiin seuraava tulos.

**Seuraus 2.1.** *Oletetaan, että*

*(i)* $|g'(x)| \leq \mu < 1$ *kaikilla* $x \in I = [a, b]$

*(ii)* $g$*:llä on kiintopiste* $x_*$ *välillä* $I$

*Tällöin*

- $g(I) \subset I$

- *jos* $x_0 \in I$*, niin kiintopisteiteraatio suppenee kohti kiintopistettä* $x_*$*.*

Huomaa, että eritysesti $x_*$ on tällöin ainoa kiintopiste kyseisellä välillä.

Suppenemisehto $|g'(x_*)| < 1$ tuntuu varsin rajoittavalta, mutta itse asiassa näin ei ole! Ensinnäkin iteraatio voidaan käytännössä kirjoittaa eri tavoin.

**Esimerkki 2.3.** Olkoon $f(x) = x - e^{-ax}$ missä $a > 0$. Selvästi funktiolla $f$ on yksi positiivinen nollakohta. Nyt tämä voitaisiin kirjoittaa kiintopisteiteraationa kahdella tavalla:

$$x_{k+1} = e^{-ax_k} \quad \text{tai} \quad x_{k+1} = -\frac{1}{a} \ln(x_k)$$

Jätetään harjoitustehtäväksi osoittaa, että jompikumpi näistä iteraatioista suppenee.

✯

Lisäksi kiintopisteiteraatio saadaan aina suppenemaan seuraavan tempun avulla. Olkoon $\alpha$ jokin parametri ja tarkastellaan iteraatiota

$$x_{k+1} = g(x_k) = x_k + \alpha f(x_k)$$

Selvästi $g$:n kiintopiste on $f$:n nollakohta olipa $\alpha$ mikä tahansa. Toisaalta $g'(x) = 1 + \alpha f'(x)$, joten sopivalla $\alpha$:n valinnalla voidaan suppenemisehto saattaa voimaan. Luonnollisesti eri nollakohdat vaativat eri $\alpha$:n.

Koska kiintopisteiteraatio on niin tärkeä annetaan vielä kolmas muotoilu joka on hiukan vahvempi kuin kaksi edellistä versiota. Tämä on yksinkertaisin versio kuuluisasta *Banachin kiintopistelauseesta.*

**Lause 2.1.** *Oletetaan, että*

*(i)* $|g'(x)| \leq \mu < 1$ *kaikilla* $x \in I = [a, b]$

*(ii)* $g(I) \subset I$

*Tällöin*

- *g:llä on yksikäsitteinen kiintopiste* $x_*$ *välillä* $I$
- *jos* $x_0 \in I$*, niin kiintopisteiteraatio suppenee kohti kiintopistettä* $x_*$*.*

Huomaa, että yksikäsitteisen kiintopisteen olemassaolo seuraa hypoteeseista. Palataan myöhemmin kiintopistelauseen yleisempiin versioihin. Katsotaan sitten toista menetelmää nollakohtien laskemiseksi.

### 2.2.2 Newtonin menetelmä

Kiintopisteiteraatio on varsin yksinkertainen ja helposti toteutettava menetelmä. Siinä käytettiin vain itse funktion $f$ arvoja. Jos käytetään myös funktion derivaattaa saadaan tietyssä mielessä tehokkaampi menetelmä.

Tarkastellaan annetun funktion $f$ graafia, jonka nollakohdat halutaan laskea. Tunnetusti käyrän tangentti pisteessä $(p, f(p))$ on

$$y - f(p) = f'(p)(x - p)$$

Tangentti approksimoi hyvin alkuperäistä funktiota pisteen $p$ läheisyydessä. Siis jos etsitään nollakohtaa, ja $p$ on lähellä sitä, niin tangentin nollakohdan pitäisi olla vielä parempi approksimaatio, kuva 2.4. Tästä saadaan

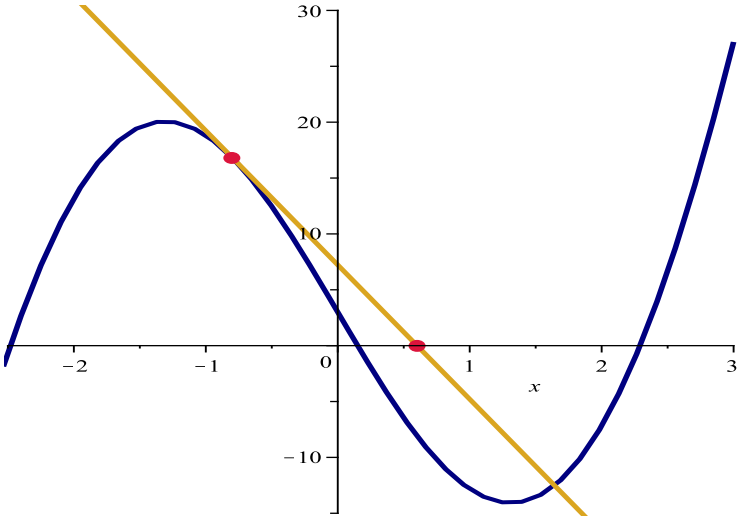
**Algoritmi 3** NEWTONIN MENETELMÄ

**Input:** valitaan jokin $x_0$ ja annetaan jokin $f$
**Output:** luku $\bar{x}$ siten että $f(\bar{x}) \approx 0$
**loop**
$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}$$
**end loop**

Vertaamalla Algoritmiin 1 huomataan, että neliöjuuri laskettiin siis soveltamalla Newtonin menetelmää polynomiin $f(x) = x^2 - a$.

**Lemma 2.2.** *Jos* $x_0$ *on riittävän lähellä nollakohtaa* $x_*$*, niin Newtonin iteraatio suppenee.*

*Todistus.* Olkoon $N(x) = x - f(x)/f'(x)$. Tällöin Newtonin iteraatio on kiintopisteiteraatio funktiolle $N$. Koska $N'(x_*) = 0$, niin Lemman 2.1 perusteella iteraatio suppenee. ✧




Kuva 2.4: Newtonin menetelmän ensimmäinen askel esimerkin 2.4 tapauksessa.

Kuten kiintopisteiteraation tapauksessa kaikki nollakohdat saadaan kun vaihdetaan ”sopivasti” alkuarvoa.

**Esimerkki 2.4.** Olkoon $f(x) = x^3 - 20\sin(x) + 3$. Kuvassa 2.4 on ensimmäinen askel, kun alkupiste $x_0 = -0.8$. Verrataan Newtonin menetelmän suppenemista kiintopisteiteraatioon. Jos kirjoitetaan

$$x_{k+1} = g(x_k) = x_k + \alpha f(x_k)$$

niin huomataan, että tämä ei suppene kun $\alpha = 1$. Kokeilemalla tai muuten tarkemmin analysoimalla nähdään, että iterointi suppenee jos valitaan esimerkiksi $\alpha = 0.05$. Kuvassa 2.5 nähdään että Newtonin menetelmä suppenee huomattavasti nopeammin. Tässä on laskettu 40 merkitsevällä numerolla. ✯

Helposti voidaan keksiä sellaisia esimerkkejä, että Newtonin menetelmä ei suppene.

**Esimerkki 2.5.** Olkoon $f(x) = xe^{-x} + 1$. Selvästi tällä on täsmälleen yksi reaalinen nollakohta. Jos kuitenkin valitaan alkuarvo $x_0 > 1$ niin

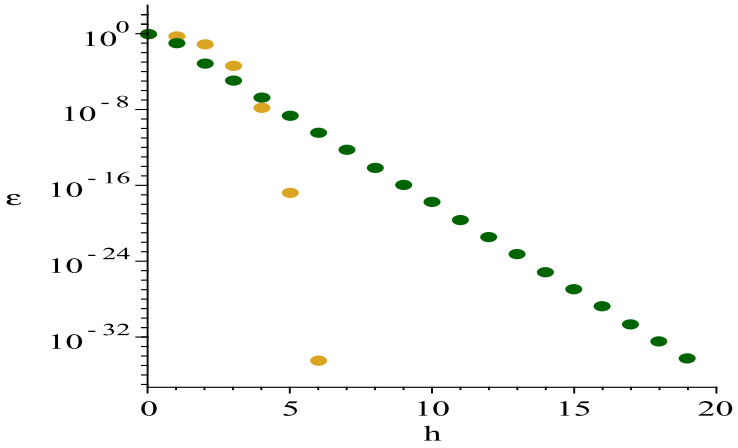
$$x_1 = x_0 - \frac{x_0 e^{-x_0} + 1}{e^{-x_0} - x_0 e^{-x_0}} = x_0 + \frac{x_0 + e^{x_0}}{x_0 - 1} > x_0 + 1$$

Selvästi siis $x_k \to \infty$. ✯

Tarkastellaan sitten lähemmin suppenemisnopeutta. Kuvan 2.5 perusteella tuntuisi, että Newtonin menetelmä olisi jossain mielessä kvalitatiivisesti parempi. Koska Newtonin iteraatio on kiintopisteiteraatio sovellettuna funktioon $N$, niin luulisi että suppenemisen nopeus liittyisi siihen, että $N'(x_*) = 0$.

Katsotaan ensin kiintopisteiteraatiota. Kiintopisteiteraation tapauksessa nähtiin että

$$|x_{k+1} - x_*| \leq \mu\,|x_k - x_*|$$




Kuva 2.5: Newtonin menetelmän suppeneminen kiintopisteiteraatioon verrattuna esimerkin 2.4 tapauksessa.

missä $\mu < 1$. Kun ollaan suhteellisen lähellä kiintopistettä niin $|x_{k+1} - x_*| \approx \mu\,|x_k - x_*|$. Olkoon $\mathcal{E}_k = |x_k - x_*|$ virhe. Tällöin

$$\mathcal{E}_k \approx \mu^k \mathcal{E}_0 \quad \text{joten} \quad \ln\left(\mathcal{E}_k\right) \approx \ln\left(\mathcal{E}_0\right) + k\,\ln(\mu)$$

Siis kiintopisteen tapauksessa pitäisi saada suora kun piirretään virheen logaritmi $k$:n funktiona. Vertaamalla kuvaan 2.5 nähdään, että tämä pitää paikkansa. Tämän takia joskus sanotaankin, että *kiintopisteiteraatio suppenee lineaarisesti.*

Näyttäisi siltä, että Newtonin menetelmä suppenee nopeammin kuin lineaarisesti.

**Määritelmä 2.2.** *Oletetaan, että $x_k \to x_*$, kun $k \to \infty$. Sanotaan, että suppeneminen on* neliöllistä*, jos on olemassa $c$ ja $N$ siten, että*

$$|x_{k+1} - x_*| \le c\,|x_k - x_*|^2 \quad \text{kun} \quad k > N$$

Neliöllisen suppenemisen tapauksessa

$$\mathcal{E}_k \approx c^k \mathcal{E}_0^{2^k} \quad \text{joten} \quad \ln\left(\mathcal{E}_k\right) \approx 2^k \ln\left(\mathcal{E}_0\right) + k\,\ln(c)$$

Nyt siis lineaarinen termi $k\,\ln(c)$ on isoilla $k$:n arvoilla merkityksettömän pieni verrattuna termiin $2^k \ln\left(\mathcal{E}_0\right)$.

**Lemma 2.3.** *Newtonin iteraation suppeneminen on neliöllistä.*

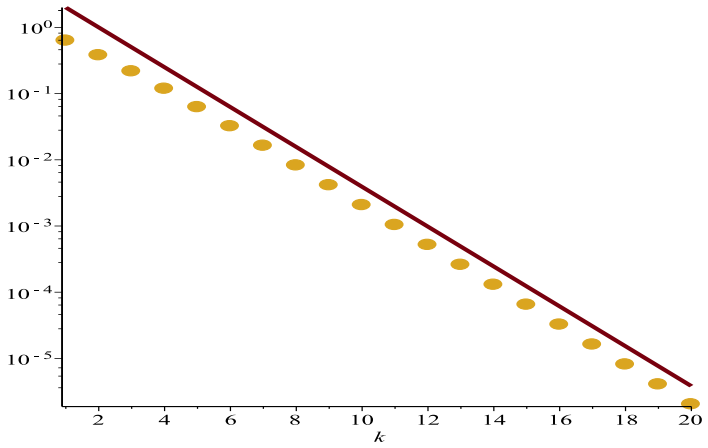
*Todistus.* Olkoon $|N''(x)| \le c$ välillä $I_\varepsilon = [x_* - \varepsilon, x_* + \varepsilon]$. Tällöin

$$|N'(x)| = |N'(x) - N'(x_*)| = \left| \int_{x_*}^{x} N''(s)ds \right| \le \int_{x_*}^{x} |N''(s)|ds \le c|x - x_*|$$

kun $x \in I_\varepsilon$. Siispä

$$\begin{aligned} |x_{k+1} - x_*| &= |N(x_k) - x_*| = |N(x_k) - N(x_*)| \\ &= \left| \int_{x_*}^{x_k} N'(s)ds \right| \le \int_{x_*}^{x_k} |N'(s)|ds \le c|x_k - x_*|^2 \end{aligned}$$

✧




Kuva 2.6: Newtonin menetelmä suppenee hitaasti kaksinkertaisen nollakohdan tapauksessa. Yhtenäinen viiva antaa teoreettisen suppenemisnopeuden $\sim 2^{-k}$.

Moninkertaiset nollakohdat aiheuttavat ongelmia. Tämä johtuu siitä, että moninkertaisessa nollakohdassa myös derivaatta on nolla. Koska Newtonin menetelmässä derivaatta on nimittäjässä, niin nollakohdan lähellä myös nimittäjä on melkein nolla. Olkoon funktiolla $f$ $m$-kertainen nollakohta $x_*$. Voidaan siis kirjoittaa $f(x) = (x - x_*)^m h(x)$ missä $h(x_*) \neq 0$. Olkoon edelleen $N(x) = x - f(x)/f'(x)$ kuten aiemmin.

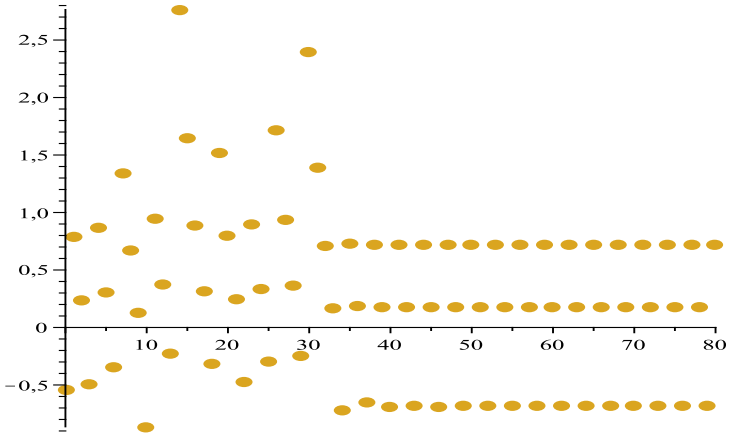
**Lemma 2.4.** *$N'(x_*) = (m-1)/m$, joten moninkertaisen nollakohdan tapauksessa Newtonin menetelmän suppeneminen on lineaarista.*

*Todistus.* Tämä on suora lasku joka jätetään harjoitustehtäväksi. ✧

**Esimerkki 2.6.** Olkoon $f = (x-1)^2(x^2+x+1)$. Edellisen Lemman perusteella $N'(1) = 1/2$. Kuvasta 2.6 nähdään, että suppeneminen tosiaan on paljon hitaampaa kuin yksinkertaisen nollakohdan tapauksessa. Huomattakoon kuitenkin, että tietyssä mielessä moninkertaiset nollakohdat ovat poikkeustapauksia. Toisaalta numeerisessa laskennassa aina esiintyy pyöristysvirheitä, joten suppeneminen voi olla käytännössä ongelmallista jos nollakohdat ovat lähellä toisiaan. ✯

Mielenkiintoista on että Newtonin menetelmä voi tuottaa stabiileja jaksollisia ratoja kuten kiintopisteiteraatiokin.

**Esimerkki 2.7.** Olkoon $f(x) = (x+2)(x^2+c)$ ja valitaan $x_0 = -0.54$ ja $c = 0.46$. Ainoa reaalinen nollakohta on $x_* = -2$. Kuvassa 2.7 nähdään, että iterointi on alussa epäsäännöllistä, mutta jonkin ajan kuluttua se sitten asettuu 3-jaksolliselle radalle. Numeerisen analyysin kannalta tämä on kiusallista, mutta dynaamisten systeemien teorian kannalta taas erittäin mielenkiintoista [6]. ✯




Kuva 2.7: Newtonin menetelmä esimerkissä 2.7.

## 2.3 Algoritmin pysäyttäminen

Edellä on sekä kiintopisteiteraation että Newtonin menetelmän tapauksessa tarkasteltu näin saadun lukujonon suppenemista. Käytännön algoritmissa pitää kuitenkin olla jokin *suppenemiskriteeri*, joka pysäyttää algoritmin. Olkoon $\bar{x}$ jokin approksimaatio funktion $f$ nollakohdalle $x_*$. Siis approksimaation *(absoluuttinen) virhe* on $\mathcal{E}_{\mathsf{abs}} = |\bar{x} - x_*|$ ja sen *suhteellinen virhe* on

$$\mathcal{E}_{\mathsf{suht}} = \frac{|\bar{x} - x_*|}{|x_*|}$$

Jos tarkastelleen lukujonon $x_k$ suppenemista niin luonnollisesti on sama kumpaa virhettä tarkastellaan: sekä suhteellinen että absoluuttinen virhe lähestyy nollaa jos lukujono suppenee. Käytännössä kuitenkin liukulukujen äärellisestä laskentatarkkuudesta johtuen suhteellinen virhe on lähes poikkeuksetta mielekkäämpi arvioitaessa approksimaation hyvyyttä.

**Esimerkki 2.8.** Olkoon funktiolla $f$ nollakohdat $x_* = 100$ ja $y_* = 0.1$ sekä approksimaatiot $\bar{x} = 99.9$ ja $\bar{y} = 0.2$. Absoluuttiset virheet ovat samat mutta

$$\frac{|\bar{x} - x_*|}{|x_*|} = 10^{-4} \quad , \quad \frac{|\bar{y} - y_*|}{|y_*|} = 1$$



Mutta miten suhteellista virhettä sitten voisi käyttää konkreettisessa algoritmissa koska virhettä ei kuitenkaan tunneta? Olkoon $x_*$ funktion $f$ nollakohta ja olkoon $x_k \to x_*$ kun $k \to \infty$. Jos jono on jo lähellä oikeaa arvoa, niin

$$\frac{|x_k - x_*|}{|x_*|} \approx \frac{|x_k - x_{k+1}|}{|x_{k+1}|}$$

Tätä voidaan käyttää suppenemiskriteerinä: jatketaan iterointia kunnes $|x_k - x_{k+1}|/|x_{k+1}|$ on pienempi kuin jokin annettu toleranssi. Tästä saadaan

**Algoritmi 4** KIINTOPISTEITERAATIO, VERSIO 2

**Input:** alkuarvo $x_0$, funktio $g$, toleranssi $\delta$, iteraatiokierrosten yläraja $n_{\mathsf{max}}$
**Output:** luku $\bar{x}$ siten että $g(\bar{x}) \approx \bar{x}$

```
k := 0 ; E := 1; x := x_0
while k < n_max and E > δ do
        y := g(x)
        E := |y − x|/|y|
        x := y
        k := k + 1
end while
x̄ := x
```

Täsmälleen samoin voidaan toteuttaa Newtonin menetelmä: laitetaan funktion $g$ paikalle $N(x) = x - f(x)/f'(x)$.

Algoritmeihin kannattaa laittaa jokin yläraja iteraatiokierroksille siltä varalta että suppenemista ei tapahdu. Entä miten toleranssi pitäisi valita? Liukuluvuilla laskettaessa suhteellinen virhe ei voi olla pienempi kuin kone-epsilon (katso liitteen A.1), ja optimitilanteessa suhteellinen virhe onkin varsin lähellä kone-epsilonia. Usein kuitenkin vähäisempikin tarkkuus on riittävä. Esimerkiksi MATLABILLA laskettaessa kone-epsilon $\approx 2 \cdot 10^{-16}$ ja toleranssiksi voisi valita vaikkapa $10^{-10}$.

Toinen mahdollisuus pysäyttää iteraatio on tarkastella *residuaalia*. Olkoon jälleen $f(x_*) = 0$ ja $x_k \to x_*$. Koska $f$ on jatkuva, niin tällöin $f(x_k) \to 0$, kun $k \to \infty$. Voitaisiin siis taas antaa jokin toleranssi $\delta$ ja lopettaa iterointi, kun $|f(x_k)| < \delta$. Valitettavasti aina ei residuaalin $|f(x_k)|$ pienuudesta seuraa, että virhe olisi pieni. Taylorin Lauseen A.1 mukaan

$$0 = f(x_*) = f(x_k) + f'(\beta)(x_* - x_k) \quad \Rightarrow \quad |x_k - x_*| = \frac{|f(x_k)|}{|f'(\beta)|}$$

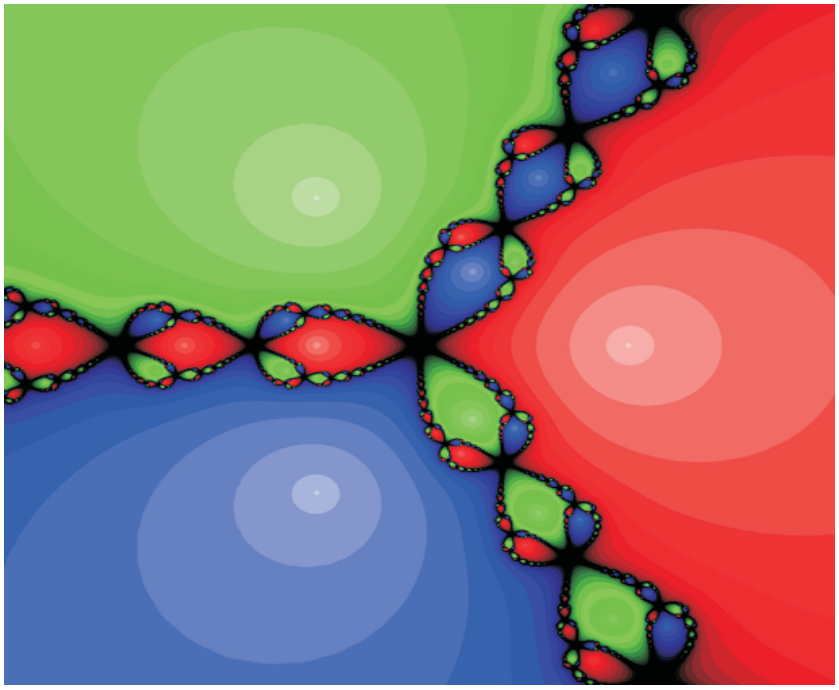
Tässä siis $\beta$ on jokin (tuntematon) luku pisteitten $x_k$ ja $x_*$ välissä. Virhe riippuu siis oleellisesti paitsi residuaalista, myös siitä mitä arvoja $f$:n derivaatta saa nollakohdan $x_*$ ympäristössä. Residuaalin käyttö ei siis ole ihan suoraviivaista mutta sen hyvä puoli on että se voidaan aina helposti laskea.

## 2.4 Erityiskysymyksiä

### 2.4.1 Kompleksiset nollakohdat

Sekä kiintopisteiteraatio että Newtonin menetelmä toimivat sellaisenaan myös kompleksisessa tapauksessa. Näitä kompleksitason iteraatioita on paljon tutkittu dynaamisten systeemien kannalta. Sovelluksissa ollaan lähinnä kiinnostuttu reaalisista ratkaisuista, tosin säätötekniikka on eräs merkittävä poikkeus. Katsotaan tähän vain eräs esimerkki Newtonin menetelmästä kompleksitasossa.

**Esimerkki 2.9.** Olkoon $f(x) = x^3 - 1$. Tällä on siis kolme nollakohtaa: $x_1 = 1$ ja $x_\pm = -1/2 \pm i\sqrt{3}/2$. Kuvassa 2.8 on väritetty kompleksitaso sen mukaan mihin näistä



Kuva 2.8: Newtonin menetelmän suppeneminen kompleksitasossa.

nollakohdista supetaan jos lähdetään kyseisestä pisteestä. Sävy on sitä vaaleampi mitä nopeampaa suppeneminen on. Luonnollisesti sinne ”väliin” jää joitain pisteitä joista lähtien iterointi ei suppene. Erikoiseksi asian tekee että näillä 3 alueella on sama reuna, katso määritelmä A.6. Tätä on intuitiivisesti hankala kuvitella, joten reunan rakenne on välttämättä hyvin monimutkainen. ✯

### 2.4.2 Polynomin kaikki nollakohdat

Yleisen jatkuvan funktion tapauksessa on tietysti mahdotonta sanoa mitään tarkkaa sen kaikkien nollakohtien joukosta. Polynomien tapauksessa on kuitenkin olemassa menetelmiä jotka ensin määrittävät kompleksitasosta jonkin rajoitetun joukon joka sisältää kaikki nollakohdat. Henricin kirjasta [13] löytyy paljon tämäntyyppisiä tuloksia. Katsotaan seuraavassa esimerkinomaisesti läpi yksi tyypillinen tulos.

Olkoon $f(x) = x^n + a_{n-1}x^{n-1} + \cdots + a_1 x + a_0$. Nyt tiedetään, että $f$:llä on korkeintaan $n$ nollakohtaa, ja jos kertaluvut lasketaan niin saadaan täsmälleen $n$ (kompleksista) nollakohtaa. Siis voidaan kirjoittaa

$$f(x) = (x - x_1)^{\ell_1} \dots (x - x_m)^{\ell_m}$$

missä $1 \leq m \leq n$ ja $\ell_1 + \cdots + \ell_m = n$. Huomattakoon, että *geneerisesti* nollakohdat ovat yksinkertaisia. Tällä tarkoitetaan sitä, että jos kertoimet $a_j$ valitaan satunnaisesti, niin nollakohdat ovat yksinkertaisia.

Yleisstrategia on siis ensin määrittää jokin rajoitettu alue $\Omega \subset \mathbb{C}$ joka sisältää kaikki nollakohdat. Kun on ensin saatu jokin yleiskäsitys, niin sitten osataan paremmin valita alkuarvot jolloin ratkaisu saadaan nopeasti Newtonin menetelmällä.

**Lemma 2.5.** *Olkoon*

$$r_1 = \max\big\{1, \sum_{j=0}^{n-1} |a_j|\big\} \quad , \quad r_2 = \Big(1 + \sum_{j=0}^{n-1} |a_j|^2\Big)^{1/2} \quad \text{ja} \quad r_\infty = 1 + \max_{0\le j\le n-1} |a_j|$$

*Olkoon* $r = \min\{r_1, r_2, r_\infty\}$. *Tällöin kaikki nollakohdat ovat kiekossa*

$$D_r = \{z \in \mathbb{C} \,|\, |z| \le r\}$$

*Todistus.* Olkoon $x_*$ nollakohta. Jos $|x_*| \le 1$, niin $x_* \in D_r$. Olkoon sitten $|x_*| > 1$. Merkitään $a = (a_0, \ldots, a_{n-1})$. Käyttäen normeja A.3 voidaan kirjoittaa

$$r_1 = \max\big\{1, |a|_1\big\} \quad , \quad r_2 = \Big(1 + |a|^2\Big)^{1/2} \quad \text{ja} \quad r_\infty = 1 + |a|_\infty$$

Tällöin

$$\begin{aligned} |x_*|^n =& | - x_*^n| = \big|a_{n-1}x_*^{n-1} + \cdots + a_1 x_* + a_0\big| \\ \le& |a_{n-1}|\,|x_*^{n-1}| + \cdots + |a_1|\,|x_*| + |a_0| \le |x_*|^{n-1}\big(|a_{n-1}| + \cdots + |a_1| + |a_0|\big) \end{aligned}$$

Siis $|x_*| \le |a|_1$ joten $x_* \in D_{r_1}$. Olkoon sitten $v_* = (1, x_*, \ldots, x_*^{n-1})$. Cauchyn epäyhtälön (A.2) avulla saadaan

$$\begin{aligned} |x_*|^{2n} =& \big|a_{n-1}x_*^{n-1} + \cdots + a_1 x_* + a_0\big|^2 = |\langle a, v_*\rangle|^2 \\ \le& |a|^2 |v_*|^2 = |a|^2 \frac{|x_*|^{2n} - 1}{|x_*|^2 - 1} \le |a|^2 \frac{|x_*|^{2n}}{|x_*|^2 - 1} \end{aligned}$$

Siis $|x_*|^2 \le 1 + |a|^2$ joten $x_* \in D_{r_2}$. Lopuksi

$$\begin{aligned} |x_*|^n =& \big|a_{n-1}x_*^{n-1} + \cdots + a_1 x_* + a_0\big| \le \max_{0\le j\le n-1} |a_j|\big(|x_*|^{n-1} + \cdots + |x_*| + 1\big) \\ =& |a|_\infty \frac{|x_*|^n - 1}{|x_*| - 1} \le |a|_\infty \frac{|x_*|^n}{|x_*| - 1} \end{aligned}$$

Siis $|x_*| \le 1 + |a|_\infty$, joten $x_* \in D_{r_\infty}$. ✧

Jos ollaan kiinostuttu kiekosta jonka keskipiste ei ole origo niin sopivalla muuttujan vaihdolla päästään tähän tapaukseen.

### 2.4.3 Reaaliset nollakohdat

Sovelluksissa ollaan usein kiinnostuttu vain reaalisista nollakohdista. Tässä tapauksessa siis ensin haluttaisiin tietää montako reaalista nollakohtaa on, ja sitten jokin rajoitettu väli $I \subset \mathbb{R}$ joka sisältää nollakohdat. Toisaalta voidaan olla kiinnostuttu vain tietystä

välistä ja halutaan tietää montaako nollakohtaa siellä on. Tämä voidaan algoritmisesti selvittää. Mielenkiintoista on kuitenkin, että tämä reaalinen analyysi on hyvin erilaista kuin "perinteinen" polynomien analyysi, joka siis viime kädessä pohjautuu kompleksianalyysiin [3].

Olkoon

$$\sigma(x) = \begin{cases} 1 & , \ x > 0 \\ 0 & , \ x = 0 \\ -1 & , \ x < 0 \end{cases}$$

ja määritellään vektoreille $v = (v_0, v_1, \ldots, v_m)$, $\sigma(v) = \big(\sigma(v_0), \sigma(v_1), \ldots, \sigma(v_m)\big)$.

**Määritelmä 2.3.** *Oletetaan, että vektorille $v$ pätee, että $v_j \neq 0$ kaikilla $j$. Tällöin $v$:n* merkkivaihdot*, $\mathsf{mv}(v)$, on niiden tapausten lukumäärä, joissa $v_j v_{j+1} < 0$. Jos jokin $v$:n alkioista on nolla, niin muodostetaan vektori $\tilde{v}$, josta nollat on poistettu ja asetetaan $\mathsf{mv}(v) = \mathsf{mv}(\tilde{v})$.*

Esimerkiksi jos

$$v = (1, 0, -2, -3, 0, 2, 4, 0, -1, 0) \quad \text{niin} \quad \mathsf{mv}(v) = 3$$

Huomaa, että $\mathsf{mv}(v) = \mathsf{mv}(\sigma(v))$. Olkoon sitten $\mathcal{P} = (p_0, p_1, \ldots, p_k)$ joukko polynomeja ja merkitään $\mathcal{P}(a) = \big(p_0(a), p_1(a), \ldots, p_k(a)\big)$. Ollaan kiinostuttu vektorin $\mathcal{P}(a)$ merkin vaihdoksista. Mutta on kätevää laajentaa tämä tapaukseen, jossa $a = \pm\infty$. Olkoon $f = a_n x^n + \cdots + a_1 x + a_0$ jokin polynomi. Nyt on luonnollista asettaa

$$\sigma(f(\infty)) = \sigma(a_n) \quad \text{ja} \quad \sigma(f(-\infty)) = \sigma(a_n)(-1)^n$$

Nyt siis kaikissa tapauksissa voidaan asettaa $\mathsf{mv}(\mathcal{P}, a) = \mathsf{mv}\big(\sigma(\mathcal{P}(a))\big)$. Edelleen jos $a < b$, niin

$$\mathsf{mv}(\mathcal{P}; a, b) = \mathsf{mv}\big(\sigma(\mathcal{P}(a))\big) - \mathsf{mv}\big(\sigma(\mathcal{P}(b))\big)$$

Tässä siis voi olla $a = -\infty$ tai $b = \infty$.

Olkoon sitten $f = a_n x^n + \cdots + a_1 x + a_0$ jokin polynomi ja olkoon $\#(f)$ reaalisten nollakohtien lukumäärä ja $\#(f; a, b)$ välillä $(a, b]$ olevien nollakohtien lukumäärä. Olkoon edelleen $\mathcal{P}_f = \big(f, f', \ldots, f^{(n)}\big)$.

**Lause 2.2.** *$\#(f; a, b) \leq \mathsf{mv}(\mathcal{P}_f; a, b)$ ja lisäksi $\mathsf{mv}(\mathcal{P}_f; a, b) - \#(f; a, b)$ on parillinen.*

Erityisesti siis jos $\mathsf{mv}(\mathcal{P}_f; a, b)$ on pariton, niin $\#(f; a, b) > 0$. Toisaalta jos $\mathsf{mv}(\mathcal{P}_f; a, b) = 0$, niin kyseisellä välillä ei ole nollakohtia.

Tässä siis saatiin jo jotain tietoa nollakohdista halutulla välillä, mutta itse asiassa nollakohtien tarkka määräkin voidaan selvittää. Tässä tarvitaan polynomien jakolaskua. Olkoon $f$ ja $g$ polynomeja ja oletetaan, että $\mathsf{deg}(f) > \mathsf{deg}(g)$. Tällöin on olemassa polynomit $q$ ja $r$ siten, että

$$f = q\, g - r \quad \text{missä} \quad \mathsf{deg}(r) < \mathsf{deg}(g)$$

Sanotaan, että $q$ on *osamäärä* ja $r$ on *jakojäännös*. Huomaa, että miinusmerkki polynomin $r$ edessä on oleellinen. Seuraavassa ollaan kiinnostuttu vain jakojäännöksestä ja merkitään $r = \mathsf{jako}(f, g)$. Tunnetusti tämän avulla saadaan

**Algoritmi 5** SUURIN YHTEINEN TEKIJÄ

**Input:** polynomit $f$ ja $g$
**Output:** $\mathsf{syt}(f, g)$
  $r_0 := f$; $r_1 := g$; $k := 0$
  **repeat**
    $k := k + 1$
    $r_{k+1} := \mathsf{jako}(r_{k-1}, r_k)$
  **until** $r_{k+1} = 0$
  $\mathsf{syt}(f, g) := r_k$

Huomaa, että algoritmi päättyy koska jakojäännösten asteluvut pienenevät. Jos siis on annettu $n$-asteinen polynomi $f$, niin asettamalla $g = f'$ voidaan edellisen algoritmin avulla laskea jakojäännösjono $\mathcal{R}_f = (r_0, r_1, \ldots, r_k)$, missä $k \leq n$. Tästä saadaan lopulta *Sturmin lause*.

**Lause 2.3.** $\#(f; a, b) = \mathsf{mv}(\mathcal{R}_f; a, b)$.

**Esimerkki 2.10.** Olkoon $f = x^4 + x^3 - x^2 + x - 2$. Jakojäännösjonoksi saadaan

$$\begin{aligned} r_0 =& f = x^4 + x^3 - x^2 + x - 2 \\ r_1 =& f' = 4x^3 + 3x^2 - 2x + 1 \\ r_2 =& (11/16)x^2 - (7/8)x + 33/16 \\ r_3 =& (448/121)x + 256/11 \\ r_4 =& -27225/784 \end{aligned}$$

Nyt siis vaikkapa

$$\sigma(\mathcal{R}_f(0)) = (-1, 1, 1, 1, -1) \quad \text{ja} \quad \sigma(\mathcal{R}_f(3)) = (1, 1, 1, 1, -1)$$

joten välillä $(0, 3)$ on yksi nollakohta. Toisaalta

$$\sigma(\mathcal{R}_f(-\infty)) = (1, -1, 1, -1, -1) \quad \text{ja} \quad \sigma(\mathcal{R}_f(\infty)) = (1, 1, 1, 1, -1)$$

joten kaikkiaan on kaksi reaalista nollakohtaa. ✯

## 2.5 Harjoitustehtäviä

1. Totea, että iteraatio (2.1) on Newtonin menetelmä polynomille $f(x) = x^4 - a$. Suppeneeko iteraatio (2.1) kaikilla alkuarvoilla $x_0 > 0$?

2. Olkoon $f(x) = (x - x_*)^2 g(x)$ missä $g(x_*) \neq 0$. Olkoon edelleen $N(x) = x - f(x)/f'(x)$. Osoita, että
$$\lim_{x \to x_*} N'(x) = 1/2$$

3. Osoita, että Lipschitz jatkuva kuvaus on jatkuva. Osoita esimerkin avulla, että jatkuva kuvaus ei välttämättä ole Lipschitz jatkuva.

4. Näytä, että algoritmi 1 on itse asiassa erikoistapaus Newtonin menetelmästä.

5. Osoita, että esimerkissä 2.3 jompikumpi annetuista kiintopisteiteraatioista suppenee. Valitse sitten jokin $a$ ja etsi jokin väli $I$ jossa $g$ on kontraktio.