Bor Plestenjak

priprave za predavanja

Numerične metode

2004/2005

verzija: 12.10.2004

Posamezna poglavja:

1. Uvod: 12.10.2004 (2004/2005)
2. Nelinearne enačbe: 2.11.2005 (2005/2006)
3. Linearni sistemi: 29.12.2003 (2003/2004)
4. Predoločeni sistemi: 29.12.2003 (2003/2004)
5. Nesimetrični lastni problem: 1.9.2001 (2001/2002)
6. Simetrični lastni problem: 20.2.2004 (2003/2004)
7. Singularni razcep: 1.9.2001 (2001/2002)
8. Iterativne metode za linearne sisteme: 1.9.2001 (2001/2002)
9. Interpolacija: 1.9.2001 (2001/2002)
10. Odvajanje: 1.9.2001 (2001/2002)
11. Integriranje: 1.9.2001 (2001/2002)
12. Diferencialne enačbe: 1.9.2001 (2001/2002)

Kazalo

I. Uvod

1.1 Uvod

Pri numeričnem reševanju problemov iščemo rešitev v numerični obliki. To pomeni, da npr. namesto $\sqrt{3}$ iščemo $1.73205\ldots$. *Numerična metoda* je postopek, s katerim iz začetnih numeričnih podatkov s končnim zaporedjem elementarnih operacij izračunamo numerični približek za rezultat določenega problema. *Elementarne operacije* so odvisne od okolja, mi bomo pod to šteli $+,-,/,*$ in $\sqrt{}$. *Numerična analiza* se ukvarja z analizo numeričnih metod.

Kdaj uporabljamo numerične metode:

- ko drugih ne poznamo:
  - iskanje ničel polinoma pete stopnje: $x^5 + 3x - 1 = 0$,
  - reševanje transcendentne enačbe: $x + \ln x = 0$,
  - računanje določenega integrala: $\int_0^1 e^{x^2} dx$;
- kadar so udobnejše oz. manj zahtevne od analitičnih rešitev:
  - računanje inverzne matrike velikosti 100×100,
  - Cardanove formule za ničle kubičnega polinoma.

Numerične metode se stalno razvijajo. Dejavniki razvoja so:

- novi pristopi in novi algoritmi,
- razvoj računalnikov,
- razvoj paralelnih računalnikov.

Glavne zahteve za dobro numerično metodo so:

- *zanesljivost*: na enostavnih problemih vedno deluje pravilno.
- *robustnost*: običajno deluje na težjih problemih, kadar pa ne deluje, vrne informacijo o tem.
- *natančnost*: izračuna rešitev tako natančno, kot je to možno glede na natančnost podanih začetnih podatkov.
- *ekonomičnost*: časovna (število operacij) in prostorska (poraba spomina).
- *uporabnost*: lahko jo uporabimo na širokem spektru problemov.
- *prijaznost do uporabnika*: je dobro dokumentirana in ima enostaven uporabniški vmesnik.

Naj bo x točna vrednost, $\widehat{x}$ pa približek za x:

- *absolutna napaka*: $\widehat{x} = x + d$, oziroma $d = \widehat{x} - x$,
- *relativna napaka*: $\widehat{x} = x(1 + d)$ oziroma $d = \dfrac{\widehat{x} - x}{x}$.

Nekaj problemov, ki jih bomo reševali:

- *nelinearne enačbe*: poišči ničlo funkcije $f : \mathbb{R} \to \mathbb{R}$.
- *linearni sistemi*: poišči $x \in \mathbb{R}^n$, ki reši sistem $Ax = b$ za $A \in \mathbb{R}^{n\times n}$ in $b \in \mathbb{R}^n$.
- *predoločeni sistemi*: poišči $x \in \mathbb{R}^n$, ki minimizira $\|Ax - b\|_2$ za $A \in \mathbb{R}^{m\times n}$ in $b \in \mathbb{R}^m$, kjer je $m > n$.
- *lastne vrednosti*: izračunaj lastne vrednosti in vektorje matrike A.
- *interpolacija*: poišči polinom, ki gre skozi točke $(x_0, y_0), \ldots, (x_n, y_n)$.
- *integriranje*: izračunaj integral $\int_a^b f(t)dt$.
- *diferencialne enačbe*: reši $y' = f(x, y)$, $y(x_0) = y_0$.

1.2 Plavajoča vejica

V računalniku so števila zapisana v plavajoči vejici kot

$$x = \pm m \cdot b^e,$$

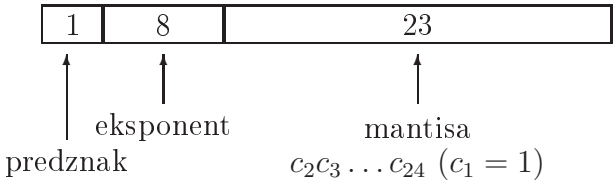
kjer je $m = 0.c_1c_2, \ldots c_t$, *mantisa* in:

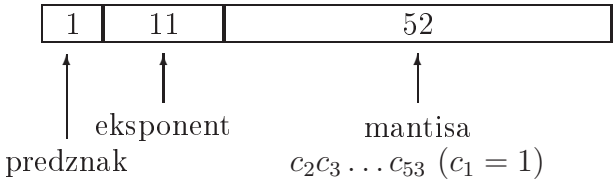
- b: *baza* (2, lahko tudi 10 (kalkulatorji) ali 16 (IBM)),
- t: *dolžina mantise*,
- e: *eksponent* v mejah $L \le e \le U$,
- c_i: *števke* v mejah od 0 do $b - 1$.
- Števila so *normalizirana*, kar pomeni $c_1 \neq 0$.

Tak zapis označimo s $P(b, t, L, U)$.

V *standardu IEEE* poznamo:

- *single*: $P(2, 24, -125, 128)$, število je shranjeno v 32 bitih,





- *double*: $P(2, 53, -1021, 1023)$, število je shranjeno v 64 bitih,
- standard IEEE pozna še števila 0, ∞, $-\infty$ in NaN.
- dopušča tudi denormalizirana števila, kjer je $c_1 = 0$, eksponent pa je fiksen (-125 za single).

Zgled 1 *Če je s predznak, $0 \le e \le 255$ eksponent in $0 \le f < 1$ število $0.c_2 \ldots c_{24}$, potem velja*

$0 < e < 255$	*poljuben* f	$x = (-1)^s(1+f)\cdot 2^{e-127}$
$e = 255$	$f = 0$	$x = (-1)^s\infty$
$e = 255$	$f \neq 0$	$x = \mathrm{NaN}$
$e = 0$	$f = 0$	$x = (-1)^s 0$
$e = 0$	$f \neq 0$	$x = (-1)^s(0+f)\cdot 2^{-126}$

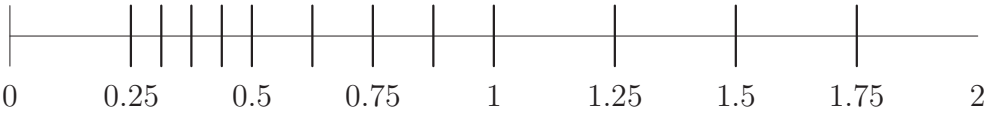
Primeri pozitivnih števil:

e	f	*število*
10000010	01100000000000000000000	$x = (1+2^{-2}+2^{-3})\cdot 2^{130-127} = 11$
11111111	00000000000000000000000	$x = \infty$
11111111	01011010100000000000000	$x = \mathrm{NaN}$
00000000	00000000000000000000000	$x = 0$
00000000	00000100000000000000000	$x = 2^{-6}\cdot 2^{-126} = 2^{-132}$

Zgled 2 *Vsa predstavljiva števila (pozitivna) iz množice $P(2, 3, -1, 1)$ so:*

$$
\begin{array}{lll}
0.100_2\ \cdot 2^{-1} = 0.2500 & 0.100_2\ \cdot 2^{0} = 0.500 & 0.100_2\ \cdot 2^{1} = 1.00 \\
0.101_2\ \cdot 2^{-1} = 0.3125 & 0.101_2\ \cdot 2^{0} = 0.625 & 0.101_2\ \cdot 2^{1} = 1.25 \\
0.110_2\ \cdot 2^{-1} = 0.3750 & 0.110_2\ \cdot 2^{0} = 0.750 & 0.110_2\ \cdot 2^{1} = 1.50 \\
0.111_2\ \cdot 2^{-1} = 0.4375 & 0.111_2\ \cdot 2^{0} = 0.875 & 0.111_2\ \cdot 2^{1} = 1.75
\end{array}
$$

■



Števila, ki niso predstavljiva, predstavimo s približki, ki jih dobimo z zaokrožanjem. Naj bo x število in $fl(x)$ najbližje predstavljivo število. Velja

$$fl(x) = x(1+\delta) \text{ in } |\delta| \leq u,$$

kjer je

$$u = \frac{1}{2}\, b^{1-t}$$

osnovna zaokrožitvena napaka:

- single: $u = 2^{-24} = 6 \cdot 10^{-8}$,
- double: $u = 2^{-53} = 1 \cdot 10^{-16}$.

Izrek 1 *Če število x leži znotraj intervala predstavljivih števil, potem velja*

$$\frac{|fl(x) - x|}{|x|} \leq \frac{u}{1+u}.$$

Dokaz. Naj bo $x = (y+z)b^e$, kjer je $y = 0.d_1 \dots d_t$ in $z = 0.0 \dots 0 d_{t+1} d_{t+2} \dots$ in naj bo $fl(x) = mb^e$, kjer je $m = 0.c_1 \dots c_t$. Predpostavimo lahko, da je x pozitiven.

a) $d_{t+1} < b/2 \implies m = y$

$$\frac{|fl(x) - x|}{|x|} = \frac{z}{y+z} \leq \frac{\frac{1}{2}b^{-t}}{b^{-1} + \frac{1}{2}b^{-t}} = \frac{u}{1+u}.$$

Pri oceni smo izbrali največji z in najmanjši y.

b) $d_{t+1} \geq b/2 \implies m = y + b^{-t}$

$$\frac{|fl(x) - x|}{|x|} = \frac{b^{-t} - z}{y+z} \leq \frac{b^{-t} - \frac{1}{2}b^{-t}}{b^{-1} + \frac{1}{2}b^{-t}} = \frac{u}{1+u}.$$

Pri oceni smo izbrali najmanjši z in najmanjši y. ■

Standard IEEE zagotavlja, da velja:

- $fl(x \oplus y) = (x \oplus y)(1+\delta)$, $|\delta| \leq u$ za $\oplus = +, -, /, *$,
- $fl(\sqrt{x}) = \sqrt{x}(1+\delta)$, $|\delta| \leq u$.

Izjema je, če pride do *prekoračitve* (overflow) ali *podkoračitve* (underflow) obsega predstavljivih števil. V tem primeru dobimo po IEEE:

- overflow: $\pm\infty$,
- underflow: 0.

1.3 Napake pri numeričnem računanju

Denimo, da želimo izračunati vrednost neke funkcije $f : X \to Y$ pri danem x. Numerična metoda vrne približek $\widehat{y}$ za y, razlika $D = y - \widehat{y}$ pa je *celotna napaka* približka.

Izvori napake so:

- nenatančnost začetnih podatkov,
- napaka numerične metode,
- zaokrožitvene napake med računanjem.

Celotno napako lahko razdelimo na tri dele.

Neodstranljiva napaka D_n

Ta napaka se pojavi zaradi napake začetnih podatkov. Nastane zaradi napake meritev ali pa zaradi aproksimacije števil s predstavljivimi. Ker ne moremo zahtevati točnih podatkov, se neodstranljivi napaki ne moremo izogniti.

Namesto z x računamo s približkom $\overline{x}$ in namesto $y = f(x)$ izračunamo $\overline{y} = f(\overline{x})$. Neodstranljiva napaka je $D_n = y - \overline{y}$.

Z analizo in ocenjevanjem neodstranljive napake se ukvarja *teorija motenj* (*perturbacij*). Pri teoriji motenj nas zanima, kako se spremeni rezultat, če malo zmotimo (perturbiramo) začetne podatke. Problem je:

- *občutljiv* (slabo pogojen): če pride do velikih sprememb,
- *neobčutljiv* (dobro pogojen): če so spremembe majhne.

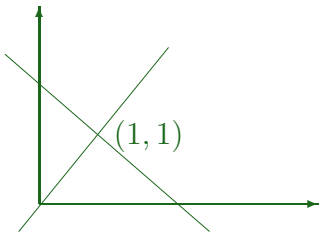
Zgled 3 *Iščemo presečišče dveh premic.*

a) $\begin{aligned} x + y &= 2 \\ x - y &= 0 \end{aligned} \implies x = y = 1.$

Zmotimo desno stran:

$\begin{aligned} x + y &= 1.9999 \\ x - y &= 0.0002 \end{aligned} \implies x = 1.00005,\ y = 0.99985.$

Ta sistem je neobčutljiv.

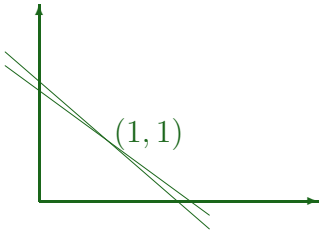


b) $\begin{aligned} x + 0.99y &= 1.99 \\ 0.99x + 0.98y &= 1.97 \end{aligned} \implies x = y = 1.$

Zmotimo desno stran:

$\begin{aligned} x + 0.99y &= 1.9899 \\ 0.99x + 0.98y &= 1.9701 \end{aligned} \implies x = 2.97,\ y = -0.99.$

Ta sistem je zelo občutljiv. ■



Stopnjo občutljivosti merimo z razmerjem med velikostjo neodstranljive napake in velikostjo napake v podatkih.

Zgled 4 *Naj bo $f : \mathbb{R} \to \mathbb{R}$ zvezna in odvedljiva funkcija. Zanima nas razlika med $f(x)$ in $f(x + \delta x)$, kjer je δx majhna motnja.*

Velja $|f(x + \delta x) - f(x)| \approx |f'(x)| \cdot |\delta x|$, torej je $|f'(x)|$ absolutna občutljivost f v točki x. Za oceno relativne napake dobimo

$$\frac{|f(x + \delta x) - f(x)|}{|f(x)|} \approx \frac{|f'(x)| \cdot |x|}{|f(x)|} \cdot \frac{|\delta x|}{|x|},$$

torej je $\frac{|f'(x)| \cdot |x|}{|f(x)|}$ relativna občutljivost f v točki x.

Napaka metode

Pri sami numerični metodi pogosto neskončen proces nadomestimo s končnim:

- seštejemo le končno členov neskončne vrste,
- po končnem številu korakov prekinemo iterativno metodo.

To pomeni, da namesto f računamo vrednost funkcije g, ki jo lahko izračunamo s končnim številom operacij. Namesto $\overline{y} = f(\overline{x})$ tako izračunamo $\widetilde{y} = g(\overline{x})$. Napaka metode je $D_m = \overline{y} - \widetilde{y}$.

Zaokrožitvena napaka

Pri računanju $\widetilde{y} = g(\overline{x})$ se pri vsaki računski operaciji pojavi zaokrožitvena napaka, tako da namesto $\widetilde{y}$ izračunamo $\widehat{y}$. Sama vrednost $\widehat{y}$ je odvisna od vrstnega reda operacij in načina izračuna $g(\overline{x})$. Zaokrožitvena napaka je $D_z = \widetilde{y} - \widehat{y}$.

Celotna napaka

Končna napaka je $D = D_n + D_m + D_z$. Velja

$$|D| \leq |D_n| + |D_m| + |D_z|,$$

zato nima smisla pretiravati z enim členom, če sta ostala dva višjega reda.

Praviloma iščemo točno rešitev, ki je ne znamo izračunati, npr. $y = \sin(\pi/10)$.

Zgled 5 *Izračunati želimo $\sin(\pi/10)$, na voljo pa imamo le kalkulator z osnovnimi štirimi operacijami, ki računa na 4 decimalna mesta točno ($P(10, 4, -5, 5)$). Namesto $f(x) = \sin x$ računamo vrednost aproksimacijske funkcije $g(x) = x - x^3/6$.*

a) neodstranljiva napaka: Namesto z $x = \pi/10$ *računamo z* $\overline{x} = 0.3142 \cdot 10^0$. *Tako dobimo*

$$D_n = y - \overline{y} = \sin(\pi/10) - \sin(0.3142) = -3.9 \cdot 10^{-5}.$$

b) napaka metode: Namesto $\sin(\overline{x})$ *izračunamo* $g(\overline{x})$ *za* $g(x) = x - x^3/6$. *Napaka je*

$$D_m = \overline{y} - \widetilde{y} = 2.5 \cdot 10^{-5}.$$

c) zaokrožitvena napaka: Odvisna je od vrstnega reda in načina računanja $g(\overline{x})$. *Denimo, da* $\overline{x} - \overline{x}^3/6$ *izračunamo po postopku:*

$$\begin{aligned} a_1 &= fl(\overline{x} * \overline{x}) = fl(0.09872164) = 0.9872 \cdot 10^{-1} \\ a_2 &= fl(a_1 * \overline{x}) = fl(0.03101154) = 0.3101 \cdot 10^{-1} \\ a_3 &= fl(a_2/6) = fl(0.0051683\ldots) = 0.5168 \cdot 10^{-2} \\ \widehat{y} &= fl(\overline{x} - a_3) = fl(0.309032) = 0.3090 \cdot 10^0 \end{aligned}$$

Ker je $\widetilde{y} = g(\overline{x}) = 0.3090302767\ldots$, *je napaka*

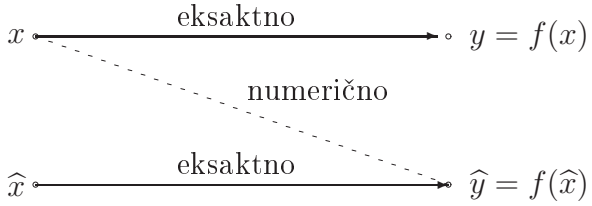
$$D_z = \widetilde{y} - \widehat{y} = 3.0 \cdot 10^{-5}.$$

Celotna napaka je $D = D_n + D_m + D_z = 1.6 \cdot 10^{-5}$. ■

1.4 Analiza zaokrožitvenih napak

Pri *analizi zaokrožitvenih napak* ločimo direktno in obratno analizo.

- direktna analiza: Iz x namesto $y = f(x)$ izračunamo $\widehat{y}$. Če je razlika med y in $\widehat{y}$ majhna (absolutno oz. relativno), je proces direktno stabilen (absolutno oz. relativno), sicer pa nestabilen.
- obratna analiza: Iz x namesto $y = f(x)$ izračunamo $\widehat{y}$. Sedaj se vprašamo, za koliko moramo spremeniti argument x v $\widehat{x}$, da bo $f(\widehat{x}) = \widehat{y}$. Če je razlika med x in $\widehat{x}$ majhna (absolutno oz. relativno), je proces obratno stabilen (absolutno oz. relativno), sicer pa nestabilen.



Zgled 6 *Računanje produkta $n+1$ prestavljivih števil.*

Dana so predstavljiva števila $x_0, x_1, \ldots, x_n$, računamo pa produkt $p = x_0 x_1 \cdots x_n$.

Eksaktni algoritem je:
$p_0 = x_0$
$i = 1, \ldots, n$
  $p_i = p_{i-1} x_i$
$p = p_n$

Dejanski algoritem pa:
$\widehat{p}_0 = x_0$
$i = 1, \ldots, n$
  $\widehat{p}_i = \widehat{p}_{i-1} x_i (1+\delta_i), \quad |\delta_i| \le u$
$\widehat{p} = \widehat{p}_n$

Dobimo

$$\widehat{p} = p(1+\gamma) = p(1+\delta_1)\cdots(1+\delta_n).$$

Velja

$$(1-u)^n \le (1+\gamma) \le (1+u)^n,$$

to pa ocenimo

$$(1+u)^n = 1 + \binom{n}{1}u + \binom{n}{2}u^2 + \cdots = 1 + nu + \mathcal{O}(u^2),$$

$$(1-u)^n \ge 1 - nu,$$

kar dokažemo z indukcijo, saj je $(1-u)^{n+1} \ge (1-nu)(1-u) = 1 - (n+1)u + nu^2$.

Tako lahko pri pogoju $nu \ll 1$ ocenimo $|\gamma| < nu$. To pomeni, da je relativna napaka odvisna od števila množenj in da se z vsakim množenjem poveča za u. ■

Zgled 7 *Računanje skalarnega produkta dveh vektorjev dolžine n.*

Imamo dva vektorja $x = (x_1 \ \cdots \ x_n)^T$ in $y = (y_1 \ \cdots \ y_n)^T$. Računamo $s = y^T x = \sum_{i=1}^n x_i y_i$.

Eksaktni algoritem je:
$s_0 = 0$
$i = 0, \ldots, n$
  $p_i = x_i y_i$
  $s_i = s_{i-1} + p_i$
$s = s_n$

Dejanski algoritem pa:
$\widehat{s}_0 = 0$
$i = 0, \ldots, n$
  $\widehat{p}_i = x_i y_i (1+\alpha_i), \ |\alpha_i| \le u$
  $\widehat{s}_i = (\widehat{s}_{i-1} + \widehat{p}_i)(1+\beta_i), \ |\beta_i| \le u$
$\widehat{s} = \widehat{s}_n$

Obratna analiza nam da

$$\widehat{s} = \sum_{i=1}^{n} x_i y_i (1+\gamma_i),$$

kjer je

$$1+\gamma_1 = (1+\alpha_1)(1+\beta_2)\cdots(1+\beta_n)$$

in

$$1+\gamma_i = (1+\alpha_i)(1+\beta_i)\cdots(1+\beta_n), \ i = 2, \ldots, n.$$

Tako dobimo ocene $|\gamma_1| \le nu$ in $|\gamma_i| \le (n-i+2)u$ za $i = 2, \ldots, n$. To pomeni, da je $\widehat{s}$ točni skalarni produkt relativno malo zmotenih vektorjev x in y. Računanje skalarnega produkta je tako obratno stabilno.

Pri direktni analizi najprej izračunamo absolutno napako:

$$\widehat{s} - s = \sum_{i=1}^{n} x_i y_i \gamma_i,$$

torej

$$|\widehat{s} - s| \leq \sum_{i=1}^{n} |x_i| \cdot |y_i| \cdot |\gamma_i| \leq nu \sum_{i=1}^{n} |x_i| \cdot |y_i| = nu|y|^T|x|.$$

Dobimo

$$\left|\frac{\widehat{s} - s}{s}\right| \leq \frac{|y|^T|x|}{|y^T x|} nu.$$

Če so vsi $x_i y_i$ enakega predznaka, dobimo $\left|\frac{\widehat{s}-s}{s}\right| \leq nu$ in računanje je direktno stabilno, sicer pa imamo v primeru, ko sta vektorja blizu ortogonalnosti, lahko veliko relativno napako. ■

Zgled 8 *Računanje vrednosti polinoma po Hornerjevem algoritmu.*

Dan je polinom

$$p(x) = a_0 x^n + a_1 x^{n-1} + \cdots + a_n,$$

računamo pa vrednost v točki x po Hornerju.

Eksaktni algoritem je:
$p_0 = a_0$
$i = 1, \ldots, n$
$\quad p_i = p_{i-1} x + a_i$
$p = p_n$

Dejanski algoritem pa:
$\widehat{p}_0 = a_0$
$i = 1, \ldots, n$
$\quad \widehat{p}_i = (\widehat{p}_{i-1} x(1 + \alpha_i) + a_i)(1 + \beta_i)$
$\widehat{p} = \widehat{p}_n$

Dobimo

$$\widehat{p} = a_0 x^n (1 + \gamma_0) + a_1 x^{n-1}(1 + \gamma_1) + \cdots + a_n (1 + \gamma_n),$$

kjer je

$$\begin{aligned} 1 + \gamma_0 &= (1 + \alpha_1) \cdots (1 + \alpha_n)(1 + \beta_1) \cdots (1 + \beta_n), \\ 1 + \gamma_i &= (1 + \alpha_{i+1}) \cdots (1 + \alpha_n)(1 + \beta_i) \cdots (1 + \beta_n), \ i = 1, \ldots, n-1, \\ 1 + \gamma_n &= (1 + \alpha_n), \end{aligned}$$

torej lahko ocenimo $|\gamma_0| \leq 2nu$ in $|\gamma_i| \leq (2(n-i)+1)u$ za $i = 1, \ldots, n$.

Računanje vrednosti polinoma je obratno stabilno, saj smo izračunali vrednost bližnjega polinoma s koeficienti $a_i(1 + \gamma_i)$ namesto a_i.

Iz absolutne napake $\widehat{p} - p = a_0 x^n \gamma_0 + a_1 x^{n-1} \gamma_1 + \cdots + a_n \gamma_n$ sledi

$$|\widehat{p} - p| \leq 2nu(|a_0||x^n| + |a_1||x^{n-1}| + \cdots + |a_n|),$$

od tod pa

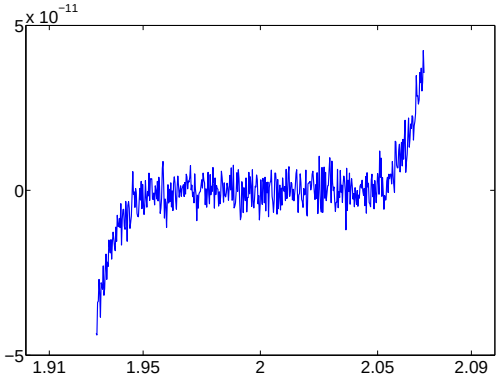
$$\frac{|\widehat{p} - p|}{|p|} \leq \frac{2nu(|a_0||x^n| + |a_1||x^{n-1}| + \cdots + |a_n|)}{|a_0 x^n + \cdots + a_n|}. \tag{1.1}$$

Računanje vrednosti polinoma po Hornerju ni direktno stabilno.

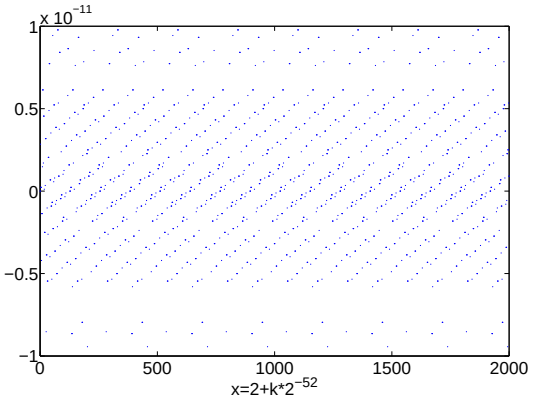
Za predstavo o dobljenih ocenah si poglejmo računanje polinoma

$$(x-2)^9 = x^9 - 18x^8 + 144x^7 - 672x^6 + 2016x^5 - 4032x^4 + 5376x^3 - 4608x^2 + 2304x - 512$$

v okolici točke 2. *Z Matlabom dobimo*



Napake niso povsem naključne, temveč tvorijo vzorec, ki ga opazimo, če pogledamo izračun pri prvih 2000 predstavljivih številih večjih od 2*:*



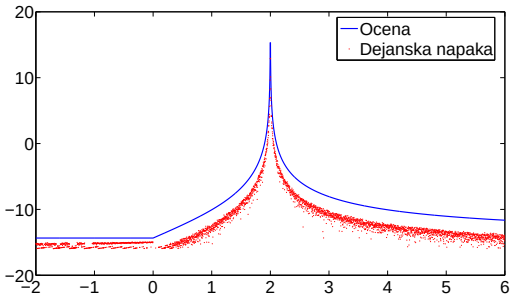
zcx

Formulo (1.1) *lahko uporabimo za sprotno računanje ocene. Če algoritem predelamo v*

$p_0 = a_0$
$e_0 = |a_0|$
$i = 1, \ldots, n$
  $p_i = p_{i-1}x + a_i$
  $e_i = e_{i-1}|x| + |a_i|$
$p = p_n$
$e = 2nu\frac{e_n}{|p|}$

je e *ocena za relativno napako.*

Zgornji graf prikazuje razmerje med resnično napako in oceno (minus desetiški logaritem nam da ravno število točnih decimalk). ■

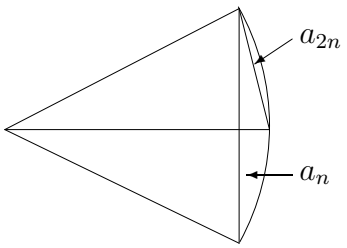


1.5 Zanimivi primeri

Poglejmo si nekaj zgledov, ki kažejo, da ni dovolj le izpeljati formulo in jo pognati v računalniku.

1.5.1 Računanje števila π po Arhimedovo

Število π je limita obsega S_n pravilnega mnogokotnika, včrtanega v krog s polmerom $r = \frac{1}{2}$. Naj bo a_n stranica pravilnega n-kotnika. Poiščimo zvezo med a_n in a_{2n}:



Velja

$$a_{2n} = \sqrt{\left(\frac{a_n}{2}\right)^2 + \left(\frac{1}{2} - \sqrt{\frac{1}{4} - \left(\frac{a_n}{2}\right)^2}\right)^2} = \sqrt{\frac{1 - \sqrt{1 - a_n^2}}{2}},$$

od tod pa iz $S_n = na_n$ sledi

$$S_{2n} = 2na_{2n} = 2n\sqrt{\frac{1 - \sqrt{1 - \left(\frac{S_n}{n}\right)^2}}{2}}. \tag{1.2}$$

Računanje začnemo pri $S_6 = 3$. Tako v enojni kot v dvojni natančnosti formula odpove, saj pride do odštevanja skoraj enako velikih števil, napaka pa se množi z $2n$.

n	Sn	n	Sn
6	3.0000000	768	3.1417003
12	3.1058285	1536	3.1430793
24	3.1326280	3072	3.1374769
48	3.1393509	6144	3.1819811
96	3.1410384	12288	3.3541021
192	3.1414828	24576	3.0000000
384	3.1414297	49152	0.0000000

Da je s formulo (1.2) nekaj narobe, pokaže že kratek razmislek. V limiti naj bi S_n šli proti π. Pri dovolj velikem n bo $fl\left(1-\frac{S_n}{n}\right)$ enako 1 in dobili bomo $S_{2n}=0$.

Za stabilno računanje je potrebno formulo preurediti. Stabilna oblika je

$$S_{2n}=2n\sqrt{\frac{\left(1-\sqrt{1-\left(\frac{S_n}{n}\right)^2}\right)\left(1+\sqrt{1-\left(\frac{S_n}{n}\right)^2}\right)}{2\left(1+\sqrt{1-\left(\frac{S_n}{n}\right)^2}\right)}}=S_n\sqrt{\frac{2}{1+\sqrt{1-\left(\frac{S_n}{n}\right)^2}}}\,. \tag{1.3}$$

Ko gre sedaj v limiti $n\to\infty$, gre spet $fl\left(1+\frac{S_n}{n}\right)$ proti 1, a sedaj iz formule 1.3 sledi, da bo v tem primeru $S_{2n}=S_n$.

Z novo formulo dobimo pravilne rezultate:

n	Sn	n	Sn
6	3.0000000	768	3.1415837
12	3.1058285	1536	3.1375901
24	3.1326284	3072	3.1415918
48	3.1393499	6144	3.1415923
96	3.1410317	12288	3.1415925
192	3.1414523	24576	3.1415925
384	3.1415575	49152	3.1415925

1.5.2 Seštevanje Taylorjeve vrste za e^{-x}

Vemo, da je

$$e^{-x}=\sum_{n=0}^{\infty}(-1)^n\frac{x^n}{n!}$$

in da vrsta konvergira za vsak $x\in\mathbb{C}$. Če pa to vrsto seštevamo numerično po vrsti, potem za $x>0$ ne dobimo najboljših rezultatov. Pri $x=10$ tako dobimo vsoto $-7.265709\cdot10^{-5}$, kar je očitni nesmisel.

x	e^{-x}	vrsta	relativna napaka
1	$3.678795\cdot10^{-1}$	$3.678794\cdot10^{-1}$	$1.6\cdot10^{-7}$
2	$1.353353\cdot10^{-1}$	$1.353353\cdot10^{-1}$	$2.2\cdot10^{-7}$
3	$4.978707\cdot10^{-2}$	$4.978701\cdot10^{-2}$	$1.1\cdot10^{-6}$
4	$1.831564\cdot10^{-2}$	$1.831532\cdot10^{-2}$	$1.7\cdot10^{-5}$
5	$6.737947\cdot10^{-3}$	$6.737461\cdot10^{-3}$	$7.2\cdot10^{-5}$
6	$2.478752\cdot10^{-3}$	$2.477056\cdot10^{-3}$	$6.8\cdot10^{-4}$
7	$9.118820\cdot10^{-4}$	$9.139091\cdot10^{-4}$	$2.2\cdot10^{-3}$
8	$3.354626\cdot10^{-4}$	$3.486091\cdot10^{-4}$	$3.9\cdot10^{-2}$
9	$1.234098\cdot10^{-4}$	$1.799157\cdot10^{-4}$	$4.6\cdot10^{-1}$
10	$4.539992\cdot10^{-5}$	$-7.265709\cdot10^{-5}$	$2.3\cdot10^{-0}$

Razlog je, da zaporedje členov vrste alternira, poleg tega pa po absolutni vrednosti nekaj časa naraščajo, preden začnejo padati proti 0. Ko so členi največji, se zameglijo majhne decimalke,

ki ostanejo netočne do konca računanja.

n	a_n	s_n	n	a_n	sn
0	1.000000	1.000000	20	41.103188	13.396751
1	−10.000000	−9.000000	21	−19.572947	−6.176195
2	50.000000	41.000000	22	8.896794	2.720599
3	−166.666672	−125.666672	23	−3.868171	−1.147572
4	416.666687	291.000000	24	1.611738	0.464166
5	−833.333374	−542.333374	25	−0.644695	−0.180529
6	1388.888916	846.555542	26	0.247960	0.067430
7	−1984.127075	−1137.571533	27	−0.091837	−0.024407
8	2480.158936	1342.587402	28	0.032799	0.008392
9	−2755.732178	−1413.144775	29	−0.011310	−0.002918
10	2755.732178	1342.587402	30	0.000380	0.000852
11	−2505.211182	−1162.623779	31	−0.001216	−0.000364
12	2087.676025	925.052246	32	0.000380	0.000016
13	−1605.904663	−680.852417	33	−0.000115	−0.000099
14	1147.074707	466.222290	34	0.000034	−0.000065
15	−764.716492	−298.494202	35	−0.000010	−0.000075
16	477.947815	179.453613	36	0.000003	−0.000072
17	−281.145782	−101.692169	37	−0.000001	−0.000073
18	156.192108	54.499939	38	0.000000	−0.000073
19	−82.206375	−27.706436	39	−0.000000	−0.000073

Pri računanju e^{10} teh težav ni, saj so vsi členi pozitivni, rezultat pa velik in je relativna napaka potem majhna.

1.5.3 Računanje I_{10}

Integrale

$$I_n = \int_0^1 x^n e^{x-1} dx,$$

$n = 0, 1, \ldots$, lahko numerično računamo rekurzivno preko formule

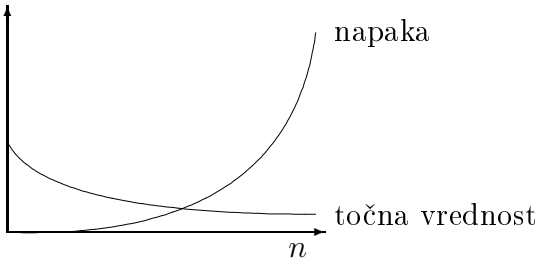
$$I_n = x^n e^{x-1}\big|_0^1 - n \int_0^1 x^{n-1} e^{x-1} dx = 1 - nI_{n-1},$$

saj poznamo začetno vrednost $I_0 = 1 - e^{-1}$.

Rezultati niso najboljši:

n	I_n	n	I_n
0	0.6321205	7	0.1124296
1	0.3678795	8	0.1005630
2	0.2642411	9	0.0949326
3	0.2072767	10	0.0506744
4	0.1708932	11	0.4425812
5	0.1455340	12	−4.3109741
6	0.1267958	13	57.0426636

Razlog je v formuli $I_n = 1 - nI_{n-1}$. Napaka pri členu I_{n-1} se pomnoži z n in torej po absolutni vrednosti hitro narašča, točne vrednosti I_n pa padajo.



Rešitev je, da vrednosti računamo v obratni smeri: $I_{n-1} = \frac{1-I_n}{n}$. Tako se napaka v vsakem koraku deli z n in če začnemo pri nekem dovolj velikem členu, lahko z začetnim $I_n = 0$ izračunamo vse začetne člene dovolj natančno. Če začnemo z $I_{26} = 0$ tako dobimo (v enojni natančnosti) vse člene od I_{12} do I_0 na vse decimalke točno.

n	I_n	n	I_n
0	0.6321205	8	0.1009320
1	0.3678795	9	0.0916123
2	0.2642411	10	0.0838771
3	0.2072766	11	0.0773522
4	0.1708934	12	0.0717733
5	0.1455329	13	0.0669477
6	0.1268024	⋮	⋮
7	0.1123835	26	0.0000000

1.5.4 Zaporedno korenjenje in kvadriranje

Vzamemo število $x > 0$, ga najprej 80-krat korenimo in nato 80-krat kvadriramo. Kaj dobimo? Izkaže se, da za $x \geq 1$ dobimo 1, za $0 < x < 1$ pa dobimo 0! Za razumljivejšo razlago si poglejmo model kalkulatorja HP 48G, kjer je baza desetiška, dolžina mantise pa je 12.

Poglejmo najprej primer $0 < x < 1$. Za takšne x velja $\sqrt{x} > x$. Največje predstavljivo število, ki je še manjše od 1, je $1 - 10^{-12}$ oziroma $0.\underbrace{9\ldots9}_{12}$. Zaradi $\sqrt{1+x} = 1 + \frac{1}{2}x - \frac{1}{8}x^2 + \cdots$ velja

$$\sqrt{1-10^{-12}} = 1 - \frac{1}{2}10^{-12} - \frac{1}{8}10^{-24} + \cdots = 0.\underbrace{9\ldots9}_{12}4\underbrace{9\ldots9}_{11}87\ldots,$$

in število se zaokroži na $0.\underbrace{9\ldots9}_{12}$. Tako s korenjenjem nikoli ne pridemo do 1, ko pa kvadriramo, je število vedno manjše, dokler ne pride do podkoračitve in dobimo 0.

Prvo predstavljivo število, ki je večje od 1 je $1 + 10^{-11}$. Tu dobimo

$$\sqrt{1+10^{-11}} = 1 + \frac{1}{2}10^{-11} - \frac{1}{8}10^{-22} + \cdots = 1.\underbrace{0\ldots0}_{11}4\underbrace{9\ldots9}_{10}87\ldots,$$

kar se zaokroži na 1. Tako s korenjenjem pridemo do 1, s kvadriranjem pa se to ne spremeni.

II. Reševanje nelinearnih enačb

2.1 Bisekcija

Iščemo rešitve enačbe $f(x) = 0$, kjer je $f : \mathbb{R} \to \mathbb{R}$ dana realna funkcija. Ponavadi zahtevamo, da je f zvezna, saj imamo potem naslednji eksistenčni izrek.

Izrek 2 *Če je f realna zvezna funkcija na $[a, b]$ in velja $f(a) \cdot f(b) < 0$, potem obstaja tak $\xi \in (a, b)$, da je $f(\xi) = 0$.* ■

Algoritem za bisekcijo je:

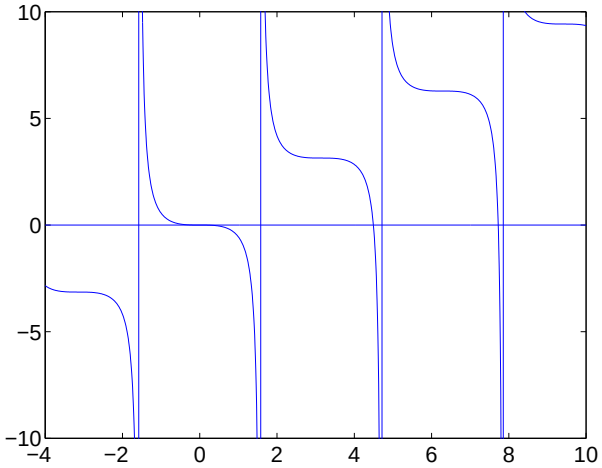
```
function c=bisekcija(f,a,b,ε)
    fa = f(a)
    fb = f(b)
    e = b − a
    if sgn(fa) = sgn(fb) then stop end if
    repeat
        e = e/2
        c = a + e
        fc = f(c)
        if sgn(fa) = sgn(fc) then
            a = c
            fa = fc
        else
            b = c
            fb = fc
        end if
    until |e| ≤ ε
```

Opombe:

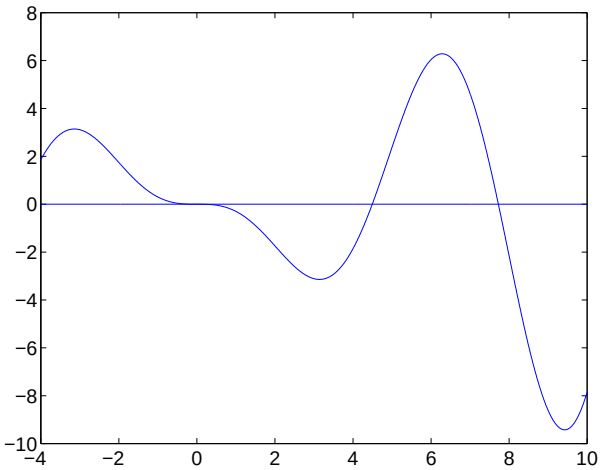
- Zakaj namesto $c = (a + b)/2$ uporabljamo $c = a + e$?
  - Pri numeričnem računanju lahko $fl((a + b)/2)$ sploh ne leži v $[a, b]$. Če npr. računamo v $P(10, 4, -10, 10)$, potem za $a = 0.6666$ in $b = 0.6667$ dobimo $fl((a+b)/2) = 0.6665$.
  - Potem bi morali uporabljati pogoj $|b - a| \leq \epsilon$, ki pa lahko nikoli ni izpolnjen. Če npr. računamo v IEEE single, potem iz $a = 1000$ in $b = 1001$ ne moremo dobiti intervala, za katerega bi veljalo $|b - a| \leq 10^{-10}$. Pri pogoju $|e| \leq \epsilon$ teh težav ni.
- Metoda očitno odpove pri sodih ničlah.
- Zakaj ne testiramo, če je $fc = 0$? Ker pri numeričnem računanju ta pogoj skoraj nikoli ni izpolnjen.

- Zakaj namesto $fa \cdot fc > 0$ preverjamo $\operatorname{sgn}(fa) = \operatorname{sgn}(fc)$? Računanje produkta $fa \cdot fc$ je zahtevnejše od preverjanja predznakov, poleg tega pa se tudi izognemu možnim težavam zaradi podkoračitve (na srečo ima IEEE zato $+0$, -0, ∞ in $-\infty$, da ne pride do težav) ali prekoračitve.
- Bisekcijo lahko uporabljamo tudi za iskanje lihih polov, računati pa moramo na težave zaradi prekoračitve v bližini pola.
- Koliko korakov bisekcije moramo narediti? k mora biti tako velik, da bo $|b-a|2^{-k} \leq \epsilon$. Rešitev je
$$k \geq \log_2 \left(\frac{\epsilon}{|b-a|} \right).$$
- Če je na $[a, b]$ več rešitev, bisekcija najde le eno.

Zgled 9 *Z bisekcijo poiščimo ničle funkcije* $f(x) = x - \tan(x)$.



Razen za trivialno ničlo $x = 0$ *je težko poiskati dober začetni interval, saj imamo težave s poli. Rešitev je, da namesto* f *iščemo ničle* $g(x) = x\cos(x) - \sin(x)$.



Sedaj nimamo težav s poli. ■

2.2 Navadna iteracija

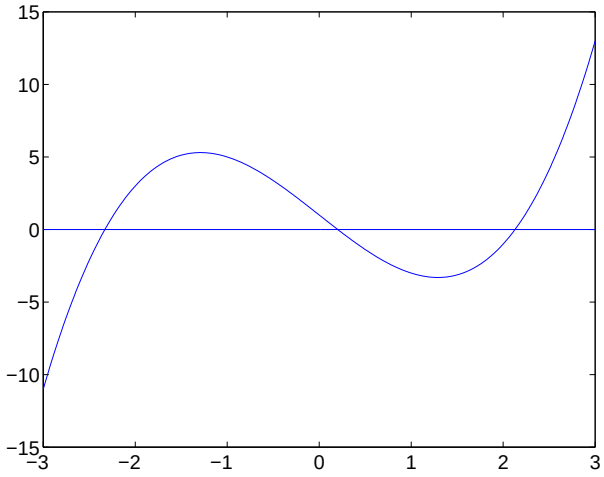
Enačbo $f(x) = 0$ ponavadi rešujemo *iterativno*, kar pomeni, da ponavljamo preprost postopek, s katerim dobimo zaporedje približkov h korenu α. Postopku pravimo *iteracija*.

Pri navadni iteraciji enačbo $f(x) = 0$ zapišemo v obliki $x = g(x)$ in nato delamo iteracijo $x_{r+1} = g(x_r)$, $r = 0, 1, \ldots$. Funkcijo g imenujemo *iteracijska funkcija*.

Zgledi iteracijskih funkcij:

- $g(x) = x - f(x)$,
- $g(x) = x - Cf(x)$, kjer je $C \neq 0$,
- $g(x) = x - h(x)f(x)$, kjer je $h(x) \neq 0$.

Zgled 10 *Iz grafa polinoma $p(x) = x^3 - 5x + 1$ vidimo, da ena ničla leži blizu 0. Enačbo*



zapišemo v obliki $x = \frac{1+x^3}{5}$ in tvorimo zaporedje približkov

$x_0 = 0$

$x_{r+1} = \frac{1+x_r^3}{5}$, $r = 0, 1, \ldots$

To zaporedje skonvergira k rešitvi $\alpha = 0.210639676\ldots$. To deluje le za koren v bližini točke 0, ne pa tudi za ostali dve ničli. Zakaj? Do ostalih dveh korenov pridemo npr. z iteracijo $x_{r+1} = \sqrt[3]{5x_r - 1}$.

Izrek 3 *Naj bo α koren enačbe $x = g(x)$, naj bo g zvezno odvedljiva na $I = [\alpha - d, \alpha + d]$ in naj velja $|g'(x)| \leq m < 1$ za vsak $x \in I$. Potem za vsak $x_0 \in I$ zaporedje*

$$x_{r+1} = g(x_r), \quad r = 0, 1, \ldots$$

konvergira k α in velja ocena za napako

$$|x_r - \alpha| \leq \frac{m}{1-m}|x_r - x_{r-1}|.$$

Dokaz. Najprej konvergenca. Z indukcijo pokažemo, da vsi približki x_r ležijo v I. Denimo, da je $x_r \in I$. Potem velja

$$|x_{r+1} - \alpha| = |g(x_r) - g(\alpha)| \leq m|x_r - \alpha| \leq md < d, \tag{2.4}$$

kar pomeni $x_{r+1} \in I$. Ker je $x_0 \in I$, ležijo vsi približki v I. Iz (2.4) sledi $|x_r - \alpha| \leq m^r|x_0 - \alpha|$, to pa pomeni $\lim_{r\to\infty} x_r = \alpha$, saj je $0 \leq m < 1$.

Velja tudi $|x_{r+1} - x_r| = |g(x_r) - g(x_{r-1})| \leq m|x_r - x_{r-1}|$, kar pomeni

$$|x_{r+p} - x_r| \leq |x_{r+p} - x_{r+p-1}| + \cdots + |x_{r+1} - x_r| \leq (m^p + \cdots + m)|x_r - x_{r-1}| = m\frac{1-m^p}{1-m}|x_r - x_{r-1}|.$$

Ko pošljemo $p \to \infty$, dobimo iskano oceno. ■

O hitrosti konvergence v bližini korena odloča število $g'(\alpha)$. Če je $|g'(\alpha)| < 1$, potem je α *privlačna točka* in izrek zagotavlja konvergenco v neki okolici okrog α. V primeru $|g'(\alpha)| > 1$ je α *odbojna točka*.

Definicija 1 *Red konvergence je p, če v bližini rešitve obstajata taki konstanti $C_1, C_2 > 0$, da velja*

$$C_1|x_r - \alpha|^p \leq |x_{r+1} - \alpha| \leq C_2|x_r - \alpha|^p.$$

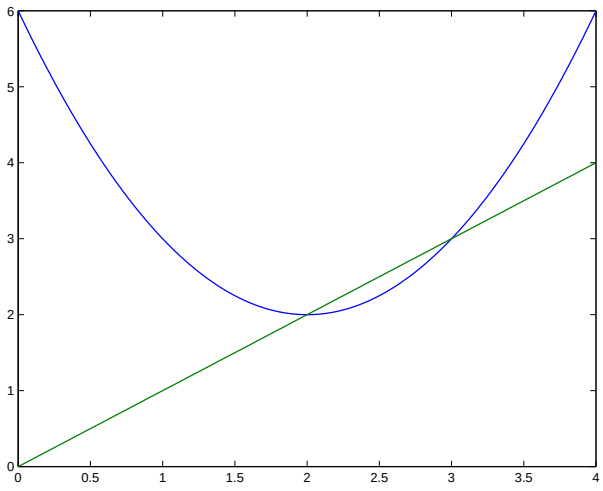
Lema 1 *Naj bo iterativna funkcija g v okolici korena rešitve α enačbe $x = g(x)$ p-krat zvezno odvedljiva in naj velja $g^{(k)}(\alpha) = 0$, $k = 1, \ldots, p-1$ in $g^{(p)}(\alpha) \neq 0$. Potem ima iterativna metoda $x_{r+1} = g(x_r)$, $r = 0, 1, \ldots$, v bližini rešitve red konvergence p.*

Dokaz. Iz razvoja g v Taylorjevo vrsto okrog α dobimo $g(x) = \alpha + \frac{1}{p!}(x-\alpha)^p g^{(p)}(\xi)$. ■

Posebni primeri konvergence:

- $p = 1$: linearna (konstantno korakov za novo točno decimalko)
- $p = 2$: kvadratična (število točnih decimalk se v vsakem koraku podvoji)
- $p = 3$: kubična (število točnih decimalk se v vsakem koraku potroji)

Zgled 11 *Poglejmo si, kaj se dogaja pri navadni iteraciji $g(x) = x^2 - 4x + 6$.*



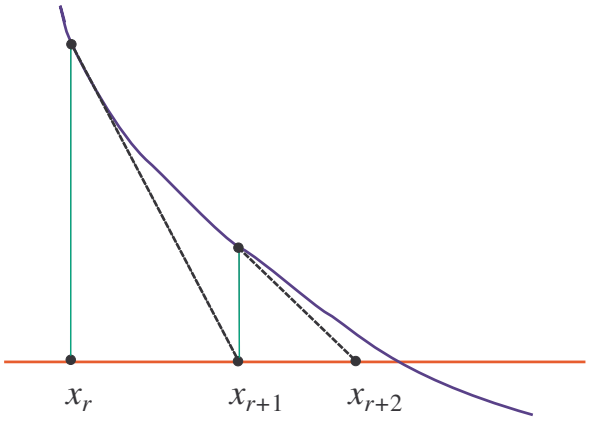
Iz slike opazimo, da imamo v primeru $x_0 \in (1,3)$ konvergenco proti 2, sicer pa za $x_0 < 1$ in $x_0 > 3$ zaporedje divergira. To lahko pokažemo tudi računsko, saj velja

$$|x_{r+1} - 2| = (x_r - 2)^2.$$

Konvergenca je v bližini 2 kvadratična, saj je $g'(2) = 0$ in $g''(2) \neq 0$. Izrek zagotavlja konvergenco le za $x_0 \in (\frac{3}{2}, \frac{5}{2})$, saj je tu $|g'(x)| < 1$, v praksi pa imamo konvergenco na širšem območju. ■

2.3 Tangentna metoda

Pri *tangentni* oz. *Newtonovi metodi* funkcijo f aproksimiramo s tangento v točki $(x_r, f(x_r))$ in za naslednji približek vzamemo presečišče tangente z osjo x. To je geometrijska razlaga.



Analitično je tangentna metoda posebna oblika navadne iteracije, kjer za iteracijsko funkcijo vzamemo

$$g(x) = x - \frac{f(x)}{f'(x)}.$$

Do zveze pridemo z računanjem presečišča tangente in osi x ali pa preko Taylorjeve vrste. Če je α rešitev enačbe $f(x) = 0$ in razvijemo $f(\alpha)$ okrog točke x_r, dobimo

$$0 = f(\alpha) = f(x_r) + f'(x_r)(\alpha - x_r) + \frac{f''(x_r)}{2}(\alpha - x_r)^2 + \cdots.$$

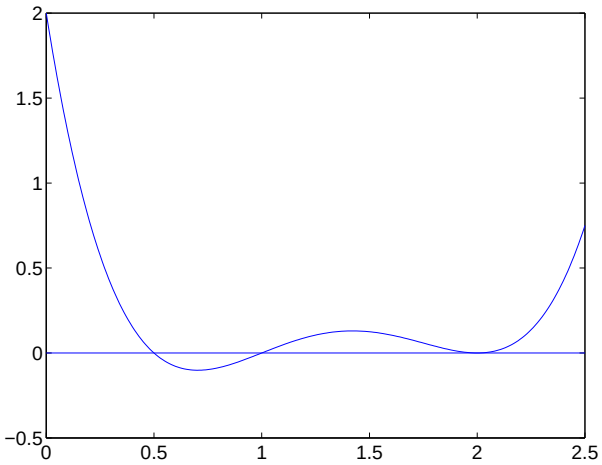
Če upoštevamo le prva dva člena, dobimo za α ravno naslednji približek po tangentni metodi.

Z odvajanjem dobimo $g'(x) = \frac{f(x)f''(x)}{f'^2(x)}$. Če je α enostaven koren, dobimo $g'(\alpha) = 0$ in konvergenca tangentne metode je vsaj kvadratična. V tem primeru je $g''(\alpha) = \frac{f''(\alpha)}{f'(\alpha)}$, kar pomeni, da je v primeru $f''(\alpha) \neq 0$ konvergenca kvadratična, sicer pa vsaj kubična. Če je α m-kratni koren, dobimo $\lim_{x\to\alpha} g'(x) = 1 - \frac{1}{m}$.

Konvergenca tangentne metode je

- kvadratična: enostaven koren in $f''(\alpha) \neq 0$ (standardna situacija),
- linearna: večkratni koren,
- vsaj kubična: enostaven koren in $f''(\alpha) = 0$.

Zgled 12 *Poglejmo polinom* $p(x) = (x-\frac{1}{2})(x-1)(x-2)^2$. *Če vzamemo začetne približke* 0.6,



1.1 *in* 2.1 *opazimo, da imamo pri* 0.6 *kvadratično konvergenco, pri* 1.1 *kubično konvergenco (prevoj) in pri* 2.1 *linearno konvergenco (dvojna ničla).*

Konvergenčni izrek za tangentno metodo:

Izrek 4 *Naj bo* α *enostavna ničla dvakrat zvezno odvedljive funkcije* f. *Potem obstajata okolica* I *točke* α *in konstanta* C, *da tangentna metoda konvergira za vsak* $x_0 \in I$ *in da približki* x_r *zadoščajo oceni*

$$|x_{r+1} - \alpha| \leq C(x_r - \alpha)^2.$$

Dokaz. Označimo napako r-tega približka z $\epsilon_r = x_r - \alpha$. Iz razvoja

$$0 = f(\alpha) = f(x_r) + f'(x_r)(\alpha - x_r) + \frac{f''(\eta_r)}{2}(\alpha - x_r)^2,$$

kjer je η_r med α in x_r, dobimo

$$x_{r+1} - \alpha = \frac{f''(\eta_r)}{2f'(x_r)}(x_r - \alpha)^2.$$

To pomeni

$$\epsilon_{r+1} = \frac{f''(\eta_r)}{2f'(x_r)}\epsilon_r^2. \tag{2.5}$$

Naj bo

$$C(\delta) = \frac{\max_{|x-\alpha|\le\delta} |f''(x)|}{2\min_{|x-\alpha|\le\delta} |f'(x)|}.$$

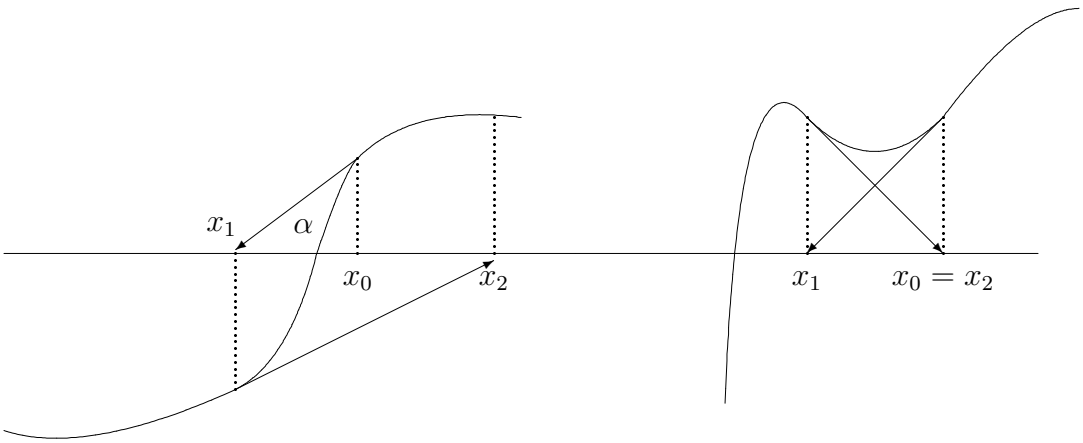
Ker je $C(0) = \frac{|f''(\alpha)|}{2|f'(\alpha)|}$, obstaja tak δ_0, da za vsak $0 \le \delta \le \delta_0$ velja $\delta C(\delta) < 1$. Vzamemo $I = [\alpha - \delta_0, \alpha + \delta_0]$. Za vsak $x_r \in I$ dobimo $\epsilon_{r+1} \le C(\delta_0)\epsilon_r^2 \le C(\delta_0)\delta_0\epsilon_r < \epsilon_r$. Torej po eni strani vsi približki ostanejo v I, po drugi strani pa velja $\lim_{r\to\infty} x_r = \alpha$. ■

Pri določenih pogojih lahko dokažemo tudi globalno konvergenco iz poljubne začetne točke.

Izrek 5 *Naj bo f na intervalu $I = [a, \infty)$ dvakrat zvezno odvedljiva, naraščajoča in konveksna funkcija, ki ima ničlo $\alpha \in I$. Potem je α edina ničla f na I in za vsak $x_0 \in I$ tangentna metoda konvergira k α.*

Dokaz. Ker je f naraščajoča, je očitno $f' > 0$ in α je edina ničla na I. Konveksnost pomeni $f'' > 0$ na I. Iz (2.5) sledi $\epsilon_{r+1} > 0$, kar pomeni, da je $\alpha \le x_r$ za $r \ge 1$. Od tod za $r \ge 1$ sledi $\alpha \le x_{r+1} \le x_r$, saj je $f(x_r) \ge 0$. ■

Izreki nam zagotavljajo konvergenco tangentne metode dovolj blizu ničle ali pa za funkcijo prave oblike. Sicer pa imamo lahko zelo različno obnašanje in slepa uporaba tangentne metode ni priporočljiva. Naslednja dva primera kažeta divergenco, ko nismo dovolj blizu ničle in zaciklanje približkov.



Tangentno metodo lahko uporabljamo tudi za računanje kompleksnih korenov enačb. Naj bo f analitična funkcija in naj velja $f'(z_0) \neq 0$. Označimo ploskev $w(z) = |f(z)|$ nad kompleksno ravnino. Kompleksno število

$$z_1 = z_0 - \frac{f(z_0)}{f'(z_0)},$$

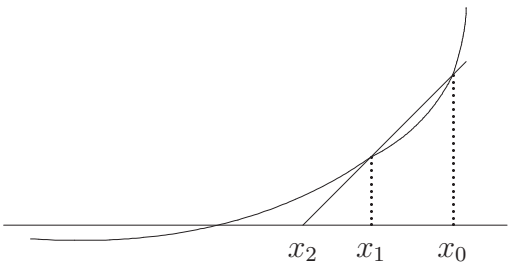
ki ga dobimo po enem koraku tangentne metode, je presečišče dveh premic:

- presečišča tangentne ravnine na ploskev $w = |w(z)|$ v točki $(z_0, w(z_0))$ z ravnino $w = 0$,
- projekcije gradienta na ploskev $w = |w(z)|$ v točki $(z_0, w(z_0))$ na ravnino $w = 0$.

2.4 Ostale metode

Na kratko omenimo še nekaj numeričnih metod za reševanje nelinearne enačbe:

- *Sekantna metoda*: Namesto tangente uporabljamo sekanto, potrebujemo dva začetna približka:
$$x_{r+1} = x_r - \frac{x_r - x_{r-1}}{f(x_r) - f(x_{r-1})} f(x_r), \quad r = 1, 2, \ldots.$$



  - Ne spada med navadno iteracijo, saj je naslednji približek odvisen od dveh zadnjih.
  - Ne potrebujemo odvodov.
  - V vsakem koraku izračunamo le eno vrednost funkcije (razen v prvem).
  - Red konvergence je 1.62 oz. $(1+\sqrt{5})/2$, kar pomeni, da je *superlinearna*.
  - Če namesto zadnjih dveh vzamemo zadnje tri točke in skozi napeljemo parabolo, dobimo Mullerjevo metodo, ki ima red konvergence 1.84. Prednost je, da jo lahko uporabljamo za iskanje kompleksnih ničel z realno aritmetiko.

Zgled 13 *Če primerjamo natančnost tangentne in sekantne metode, potem pri tangentni metodi število točnih decimalk narašča kot*

$$1, \quad 2, \quad 4, \quad 8, \quad 16,$$

pri sekantni metodi pa kot

$$1, \quad 1.6, \quad 2.6, \quad 4.1, \quad 6.6, \quad 10.5, \quad 16.8.$$

Za maksimalno natančnost v dvojni natančnosti tako pri tangentni metodi potrebujemo 4 korake in izračun $4f$ in $4f'$. Pri sekantni metodi porabimo 6 korakov in izračun $7f$. To pomeni, da je glede na porabljeno delo sekantna metoda lahko celo boljša od tangentne.

- *Inverzna interpolacija*: Skozi nekaj zadnjih točk $x_r, x_{r-1}, \ldots, x_{r-k}$ in $f(x_r), f(x_{r-1}), \ldots, f(x_{r-k})$ napeljemo polinom p stopnje k, pri čemer zamenjamo vlogo x in y. Za naslednji približek vzamemo $x_{r+1} = p(0)$. Konvergenca je superlinearna.

- *Kombinirane metode*: S kombiniranjem različnih metod lahko dosežemo robustnost in hitro konvergenco. Če imamo tak začetni interval $[a, b]$, da je $f(a)f(b) < 0$, potem lahko npr. kombiniramo bisekcijo, sekantno metodo in inverzno interpolacijo

  Najprej a in b preuredimo tako, da je $|f(b)| \leq |f(a)|$ ter vzamemo $c = a$.

  Dokler ni $|a - b| < \epsilon$ ponavljamo

- Izračunamo novo točko d: v primeru $c \neq a$ naredimo en korak inverzne kvadratične interpolacije, v primeru $c = a$ pa sekantno metodo.
- Če d ne leži znotraj $[a, b]$, potem za d vzamemo $(a + b)/2$.
- Glede na $f(d)$ zamenjamo a oziroma b z d, točki a, b uredimo tako, da je $f(a)f(b) < 0$ in $|f(b)| \leq |f(a)|$. Če se je b premaknila, je c stara vrednost b, sicer pa je $c = a$.

Ker nam točka nikoli ne pade iz $[a, b]$ imamo zagotovljeno konvergenco, ki je v bližini ničle superlinearna.

- *Metoda (f, f', f'')*: Posplošitev tangentne metode, kjer upoštevamo tudi drugi odvod in s tem pridobimo kubično konvergenco.

$$x_{r+1} = x_r - \frac{f(x_r)}{f'(x_r)} - \frac{f''(x_r)f^2(x_r)}{2f'^3(x_r)}, \quad r = 0, 1, \ldots.$$

2.5 Ničle polinomov

Ničle polinomov so lahko zelo občutljive. Znan je Wilkinsonov primer, kjer ima polinom

$$p(x) = (x-1)(x-2)\cdots(x-20) - 2^{-23}x^{19}$$

ničle

$$\begin{aligned} x_9 &= 8.91752 \\ x_{10,11} &= 10.0953 \pm 0.64310i \\ &\vdots \\ x_{16,17} &= 16.7307 \pm 2.81263i \\ x_{18,19} &= 19.5024 \pm 1.94033i \\ x_{20} &= 20.8469 \end{aligned}$$

Čeprav so vse ničle enostavne in lepo separirane, majhna motnja povzroči velike spremembe.

2.5.1 Laguerrova metoda

Za računanje ničel polinomov imamo na voljo nekaj posebnih metod. Kot prvo si bomo pogledali *Laguerrovo metodo*. Imamo polinom $f(z) = a_0 z^n + a_1 z^{n-1} + \cdots + a_n$. Njegove ničle naj bodo $\alpha_1, \ldots, \alpha_n$, kar pomeni

$$f(z) = a_0(z - \alpha_1)\cdots(z - \alpha_n).$$

Najprej definiramo

$$S_1(z) = \sum_{i=1}^{n} \frac{1}{z - \alpha_i} = \frac{f'(z)}{f(z)}$$

in

$$S_2(z) = \sum_{i=1}^{n} \frac{1}{(z - \alpha_i)^2} = -S_1'(z) = \frac{f'^2(z) - f(z)f''(z)}{f^2(z)}.$$

Denimo, da želimo izračunati α_n. Izbrani recipročni faktor označimo z

$$a(z) = \frac{1}{z - \alpha_n},$$

$b(z)$ pa naj bo aritmetična sredina preostalih recipročnih faktorjev

$$b(z) = \frac{1}{n-1} \sum_{i=1}^{n-1} \frac{1}{z - \alpha_i}.$$

Definirajmo še odstop i-tega faktorja od aritmetične sredine

$$d_i(z) = \frac{1}{z - \alpha_i} - b(z)$$

in vsoto kvadratov odstopov

$$d(z) = \sum_{i=1}^{n-1} d_i^2(z).$$

Sedaj bomo izpustili argument z in izrazili α_n. Če iz enačb

$$\begin{aligned} S_1 &= a + (n-1)b, \\ S_2 &= a^2 + \sum_{i=1}^{n-1} (b + d_i)^2 = a^2 + (n-1)b^2 + d \end{aligned}$$

izrazimo a, dobimo

$$a_{1,2} = \frac{1}{n} \left[S_1 \pm \sqrt{(n-1)(nS_2 - S_1^2 - nd)} \right].$$

in

$$\alpha_n = z - \frac{n}{S_1 \pm \sqrt{(n-1)(nS_2 - S_1^2 - nd)}}.$$

S_1 in S_2 lahko izračunamo za vsak z, ne moremo pa izračunati d. Vendar pa za z blizu α_n velja, da je $|S_1|, |S_2| \gg 0$ in člen nd lahko zanemarimo.

Tako dobimo Laguerrovo metodo:

$$\begin{aligned} S_1 &= \frac{f'(z_r)}{f(z_r)}, \quad S_2 = \frac{f'^2(z_r) - f(z_r)f''(z_r)}{f^2(z_r)} \\ z_{r+1} &= z_r - \frac{n}{S_1 \pm \sqrt{(n-1)(nS_2 - S_1^2)}}, \end{aligned}$$

oziroma

$$z_{r+1} = z_r - \frac{nf(z_r)}{f'(z_r) \pm \sqrt{(n-1)((n-1)f'(z_r) - nf(z_r)f''(z_r))}}.$$

Za stabilno računanje izberemo tisti predznak, ki da imenovalcu večjo absolutno vrednost. Če zanemarimo člen z f'', dobimo tangentno metodo.

Izrek 6 *Če ima polinom f same realne ničle, potem za poljubni začetni približek Laguerrova metoda skonvegira k levemu ali desnemu najbližjemu korenu, pri čemer si mislimo, da sta pozitivni in negativni krak realne osi povezana v neskončnosti. V primeru enostavne ničle je red konvergence v bližini ničle kubičen.* ■

2.5.2 Durand–Kernerjeva metoda

Pri tej metodi izračunamo vse ničle polinoma hkrati. Naj bo

$$p(z) = (z-\alpha_1)(z-\alpha_2)\cdots(z-\alpha_n),$$

kar pomeni, da je vodilni koeficient enak 1.

Naj bodo $z_1,\ldots,z_n$ po vrsti približki za ničle $\alpha_1,\ldots,\alpha_n$. Poiskati želimo takšne popravke $\Delta z_1,\ldots,\Delta z_n$, da bodo $z_1+\Delta z_1,\ldots,z_n+\Delta z_n$ ničle polinoma p.

Veljati mora

$$(z-(z_1+\Delta z_1))(z-(z_2+\Delta z_2))\cdots(z-(z_n+\Delta z_n)) = p(z).$$

Če to uredimo po členih Δ_i, dobimo

$$p(z) = \prod_{j=1}^{n}(z-z_j) - \sum_{j=1}^{n}\Delta z_j \prod_{\substack{k=1\\k\neq j}}^{n}(z-z_k) + \sum_{\substack{j,k=1\\j<k}}^{n}\Delta z_j\Delta z_k \prod_{\substack{l=1\\l\neq j,k}}^{n}(z-z_l) + \cdots.$$

Če zanemarimo kvadratne in višje člene, potem bodo Δ_i le približni popravki. Ko vstavimo $z=z_i$, dobimo

$$\Delta_i = \frac{-p(z_i)}{\prod_{\substack{k=1\\k\neq i}}^{n}(z_i-z_k)}.$$

Tako dobimo Durand–Kernerjevo metodo:

$$z_i^{(r+1)} = z_i^{(r)} - \frac{p(z_i^{(r)})}{\prod_{\substack{k=1\\k\neq i}}^{n}(z_i^{(r)}-z_k^{(r)})}, \quad i=1,\ldots,n.$$

Metoda ima v primeru, ko so vse ničle enostavne, kvadratično konvergenco v bližini rešitve, konvergira pa skoraj za vsak začetni vektor $z^{(0)} = (z_1^{(0)} \ \cdots \ z_1^{(0)})^T \in \mathbb{C}^n$. Za začetni vektor lahko izberemo kar naključnih n kompleksnih števil.

Če pri izračunu uporabljamo že nove vrednosti, dobimo superkvadratično konvergenco, algoritem pa je:

$$z_i^{(r+1)} = z_i^{(r)} - \frac{p(z_i^{(r)})}{\prod_{k=1}^{j-1}(z_i^{(r+1)}-z_k^{(r)})\prod_{k=j+1}^{n}(z_i^{(r)}-z_k^{(r)})}, \quad i=1,\ldots,n.$$

Pri računanju iteriramo le tiste približke z_i, ki še niso skonvergirali.

2.5.3 Redukcija

Ko najdemo približek α za ničlo, lahko polinom reduciramo in nadaljujemo z iskanjem ničel polinoma nižje stopnje. Možnosti so:

a) direktna redukcija:

$$f(x) = (x - \alpha)(b_0 x^{n-1} + \cdots + b_{n-1}) + b_n.$$

Koeficiente b_i dobimo s Hornerjevim algoritmom:

$b_0 = a_0$
$r = 1, \ldots, n$
$\quad b_r = \alpha b_{r-1} + a_r$

Na koncu dobimo $b_n = f(\alpha)$. Če je β ničla $g(x) = b_0 x^{n-1} + \ldots + b_{n-1}$, potem je β ničla polinoma $f(x) - f(\alpha)$, ki ima zmoten prosti člen. V primeru, ko je $|\alpha|$ velika, lahko pričakujemo veliko vrednost $|f(\alpha)|$ in veliko motnjo f. To pomeni, da je direktna redukcija stabilna, če korene izločamo po padajoči absolutni vrednosti.

Z Newtonovo ali Laguerrovo metodo lahko preprosto poiščemo ničle z največjo absolutno vrednostjo. Za tiste z najmanjšo absolutno vrednostjo pa uporabimo polinom

$$h(x) = x^n f(\frac{1}{x}) = a_n x^n + a_{n-1} x^{n-1} + \cdots + a_0,$$

za katerega velja $h(x) = 0$ natanko tedaj, ko je $f(1/x) = 0$.

b) obratna redukcija:

$$f(x) = (-\alpha + x)(c_0 + c_1 x + \cdots + c_{n-1} x^{n-1}) + c_n x^n.$$

Če zapišemo enačbe za c_i, dobimo algoritem:

$c_0 = -a_n / c_0$
$r = 1, \ldots, n-1$
$\quad c_r = (c_{r-1} - a_{n-r})/\alpha$
$c_n = a_0 - c_{n-1}$

Sedaj je $c_n = \frac{f(\alpha)}{\alpha^n}$. Če je β ničla $g(x) = c_0 + \ldots + c_{n-1} x^{n-1}$, je β ničla polinoma $f(x) - \frac{f(\alpha)}{\alpha^n} x^n$. Sedaj bomo imeli majhne motnje f pri veliki $|\alpha|$, kar pomeni, da je obratna redukcija stabilna, če izločamo ničle po naraščajoči absolutni vrednosti.

c) kombinirana redukcija:

$$f(x) = (x - \alpha)(b_0 x^{n-1} + \cdots + b_{n-r-1} x^r + c_{r-1} x^{r-1} + \cdots + c_0) + A x^r.$$

Sedaj dobimo $A = \frac{f(\alpha)}{\alpha^r}$ in če je to majhno, bo postopek stabilen.

2.6 Sistemi nelinearnih enačb

Rešujemo sistem nelinearnih enačb, ki ima obliko

$$\begin{aligned} f_1(x_1, x_2, \ldots, x_n) &= 0 \\ f_2(x_1, x_2, \ldots, x_n) &= 0 \\ &\vdots \end{aligned}$$

$$f_n(x_1, x_2, \ldots, x_n) \;=\; 0.$$

Krajše pišemo $F(x) = 0$, kjer je sedaj $x \in \mathbb{R}^n$ in $F : \mathbb{R}^n \to \mathbb{R}^n$.

Prva metoda je posplošitev navadne iteracije. Sistem $F(x) = 0$ zapišemo v ekvivalentni obliki $x = G(x)$, kjer je $G : \mathbb{R}^n \to \mathbb{R}^n$ in tvorimo zaporedje približkov:

$$x^{(r+1)} = G(x^{(r)}), \quad r = 0, 1, \ldots.$$

Izrek 7 *Če obstaja območje $\Omega \subset \mathbb{R}^n$ z lastnostmi:*

a) $x \in \Omega \Rightarrow G(x) \in \Omega$,

b) $x \in \Omega \Rightarrow \rho(JG(x)) \le q < 1$, *kjer je* $JG(x)$ *Jacobijeva matrika*

$$JG(x) = \begin{bmatrix} \frac{\partial g_1(x)}{\partial x_1} & \cdots & \frac{\partial g_1(x)}{\partial x_n} \\ \vdots & & \vdots \\ \frac{\partial g_n(x)}{\partial x_1} & \cdots & \frac{\partial g_n(x)}{\partial x_n} \end{bmatrix}$$

in ρ spektralni radij (največja absolutna vrednost lastne vrednosti),

potem ima $G(x) = x$ v Ω natanko eno rešitev α, zaporedje $x^{(r+1)} = G(x^{(r)})$, $r = 0, 1, \ldots$, pa za vsak $x^{(0)} \in \Omega$ konvergira k α. ■

Sedaj je red konvergence odvisen od Jacobijeve matrike. V primeru $JG(\alpha) = 0$ dobimo vsaj kvadratično konvergenco, ta pogoj pa je izpolnjen pri posplošitvi tangentne metode.

Pri Newtonovi metodi tvorimo zaporedje

$$x^{(r+1)} = x^{(r)} - JF(x^{(r)})^{-1} F(x^{(r)}), \quad r = 0, 1, \ldots.$$

V praksi ne računamo inverza Jacobijeve matrike, temveč rešujemo sistem:

$$\begin{aligned} JF(x^{(r)})\Delta x^{(r)} &= -F(x^{(r)}), \\ x^{(r+1)} &= x^{(r)} + \Delta x^{(r)}, \quad r = 0, 1, \ldots. \end{aligned}$$

Formalna izpeljava Newtonove metode poteka preko Taylorjeve vrste. Denimo, da so vse f_i dvakrat zvezno odvedljive v okolici rešitve. Tedaj lahko razvijemo:

$$f_i(\alpha) = f_i(x^{(r)}) + \sum_{k=1}^{n} \frac{\partial f_i(x^{(r)})}{\partial x_k}(\alpha_k - x_k^{(r)}) + \cdots, \quad i = 1, \ldots, n.$$

Če zanemarimo naslednje člene in želimo, da bo $f(\alpha) = 0$, dobimo linearni sistem za popravke

$$\sum_{k=1}^{n} \frac{\partial f_i(x^{(r)})}{\partial x_k} \Delta x_k^{(r)} = -f_i(x^{(r)}), \quad i = 1, \ldots, n,$$

nato pa popravimo približke

$$x_k^{(r+1)} = x_k^{(r)} + \Delta x_k^{(r)}, \quad k = 1, \ldots, n.$$

Konvergenca Newtonove metode je v bližini enostavne ničle kvadratična, težava pa je v tem, da moramo za konvergenco ponavadi poznati dovolj dober začetni približek.

Zgled 14 *Z Newtonovo metodo in z začetnim približkom $x_0 = 2$, $y_0 = 4$, izračunaj prva dva približka sistema*

$$\begin{aligned} x^2 + y^2 - 10x + y &= 1 \\ x^2 - y^2 - x + 10y &= 25. \end{aligned}$$

Dobimo

$$\begin{aligned} f_1(x,y) &= x^2 + y^2 - 10x + y - 1 \\ f_2(x,y) &= x^2 - y^2 - x + 10y - 25 \end{aligned}$$

in

$$JF(x,y) = \begin{bmatrix} 2x - 10 & 2y + 1 \\ 2x - 1 & -2y + 10 \end{bmatrix}.$$

V prvem koraku rešimo sistem

$$\begin{bmatrix} -6 & 9 \\ 3 & 2 \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} = \begin{bmatrix} -3 \\ -1 \end{bmatrix} \quad \Rightarrow \quad \Delta x = -\frac{1}{13},\ \Delta y = -\frac{15}{39}.$$

Torej je novi približek $x_1 = 1.9231$, $y_1 = 3.6154$. Če postopek nadaljujemo, dobimo $x_2 = 1.9625$, $y_2 = 3.6262$, točen rezultat pa je $x = 1.9623$, $y = 3.6258$. ■

2.7 Variacijske metode

Iščemo ekstrem funkcije $F\colon \mathbb{R}^n - \mathbb{R}^n$, kjer je F dvakrat zvezno odvedljiva na vse spremenljivke. Potreben pogoj za ekstrem je grad$F(x) = 0$ oziroma $\frac{\partial F(x)}{\partial x_k} = 0$ za $k = 1, \ldots, n$.

Če je x stacionarna točka, potem o vrsti in obstoju ekstrema odloča Hessejeva matrika

$$HF(x) = \begin{bmatrix} \frac{\partial^2 F(x)}{\partial x_1^2} & \cdots & \frac{\partial^2 F(x)}{\partial x_1 \partial x_n} \\ \vdots & & \vdots \\ \frac{\partial^2 F(x)}{\partial x_1 \partial x_n} & \cdots & \frac{\partial^2 F(x)}{\partial x_n^2} \end{bmatrix}.$$

Velja:

- $HF(x)$ pozitivno definitna: lokalni minimum,
- $HF(x)$ negativno definitna: lokalni maksimum,
- $HF(x)$ semidefinitna: odločajo višji odvodi,
- $HF(x)$ nedefinitna: ni ekstrema.

Torej lahko iskanje ekstrema funkcije več spremenljivk prevedemo na reševanje sistema nelinearnih enačb. Gre pa tudi obratno.

Denimo, da iščemo rešitev sistema $F(x) = 0$, kjer je $F\colon \mathbb{R}^n \to \mathbb{R}^n$. Funkcija

$$G(x) = \sum_{i=1}^{n} f_i^2(x)$$

ima globalni minimum ravno v točkah, kjer je $F(x) = 0$, zato lahko ničlo F poiščemo tako, da poiščemo globalni minimum G.

2.8 Reševanje nelinearnih enačb v Matlabu

V standardni verziji so na voljo naslednje funkcije:

- `fzero`: iskanje ničle realne funkcije ene spremenljivke
- `fminsearch`: iskanje minimuma realne funkcije iz $\mathbb{R}^n$ v $\mathbb{R}$.
- `fminbnd`: iskanje minimuma realne funkcije ene spremenljivke.

Primeri uporabe:

- `fzero('cos(x)-x',0)`
- `f=inline('cos(x)-x'); fzero(f,[-1,1])`
- `[x,fval]=fzero('x-tan(x)',1)`
- `fzero('cos(x)-x',0,optimset('Display','iter','TolX',1e-4))`
- `f=inline('x(1)^2+x(2)^2'); fminsearch(f,[0.3;0.2])`

Parametri, ki jih lahko nastavimo z `optimset`:

- `'Display'`: ali se izpisujejo tekoči približki, možnosti so `'iter'`, `'off'`, `'final'`.
- `'TolX'`: pogoj za konec (ko bo razlika zadnjih dveh približkov pod mejo).
- `'MaxIter'`: maksimalno število iteracij.
- `'TolFun'`: pogoj za konec (ko bo vrednost funkcije pod mejo).

III. Reševanje linearnih sistemov

3.1 Oznake

Sistem n linearnih enačb z n neznankami

$$\begin{aligned} a_{11}x_1 + \cdots + a_{1n}x_n &= b_1 \\ &\vdots \\ a_{n1}x_1 + \cdots + a_{nn}x_n &= b_n \end{aligned}$$

zapišemo v obliki

$$Ax = b, \quad A \in \mathbb{R}^{n\times n}\ (\mathbb{C}^{n\times n}), \quad x, b \in \mathbb{R}^n\ (\mathbb{C}^n),$$

kjer je A realna (kompleksna) matrika, x, b pa realna (kompleksna) vektorja. Pri tem (i, j)-ti element matrike A označimo z a_{ij}, x_i pa je i-ti element vektorja x. Vektor x zapišemo kot

$$x = \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} = [x_1 \ \cdots \ x_n]^T.$$

Matriko A zapišemo po stolpcih ali vrsticah v obliki

$$A = [a_1 \ \cdots \ a_n] = \begin{bmatrix} \alpha_1^T \\ \vdots \\ \alpha_n^T \end{bmatrix},$$

kjer so $a_i, \alpha_i \in \mathbb{C}^n$.

$e_i \in \mathbb{R}^n$ je enotski vektor, za katerega velja $e_{ij} = \delta_{ij}$, dobimo pa

- Ae_k je k-ti stolpec A,
- $e_i^T A$ je i-ta vrstica A,
- $e_i^T Ae_k$ je (i, k)-ti element A.

A^T pomeni transponirano matriko A, A^* pa je $\overline{A^T}$.

Skalarni produkt vektorjev x in y zapišemo v obliki

- x, y realna: $y^T x = \sum_{i=1}^n x_i y_i$,
- x, y kompleksna: $y^* x = \sum_{i=1}^n x_i \overline{y_i}$,

Množenje vektorja z matriko $y = Ax$ si lahko predstavljamo na dva načina:

- $y_i = \alpha_i^T x$: i-ti element y je produkt i-te vrstice A in vektorja x,

- $y = \sum_{i=1}^{n} x_i a_i$: y je linearna kombinacija stolpcev matrike A.

Podobno si lahko množenje matrik $C = AB$ predstavljamo na tri načine:

- $c_{ij} = \alpha_i^T b_j$: (i,j)-ti element C je produkt i-te vrstice A in j-tega stolpca B,
- $c_i = Ab_i$: i-ti stolpec C je produkt A in i-tega stolpca B,
- $C = \sum_{i=1}^{n} a_i \beta_i^T$: C je vsota n produktov i-tega stolpca A in i-te vrstice B.

Matriko z obliko xy^T, kjer je $x, y \neq 0$, imenujemo *diada* in ima rang ena.

3.2 Vektorske in matrične norme

Definicija 2 *Vektorska norma je preslikava $\|.\| : \mathbb{C}^n \to \mathbb{R}$, za katero velja*

1. $\|x\| \geq 0$, $\|x\| = 0 \iff x = 0$,
2. $\|\alpha x\| = |\alpha| \cdot \|x\|$,
3. $\|x + y\| \leq \|x\| + \|y\|$ *(trikotniška neenakost),*

za vsak $x, y \in \mathbb{C}^n$ in $\alpha \in \mathbb{C}$.

Najbolj znane vektorske norme so:

- $\|x\|_1 = \sum_{i=1}^{n} |x_i|$ (1-norma),
- $\|x\|_2 = (\sum_{i=1}^{n} |x_i|^2)^{1/2}$ (2-norma ali evklidska norma),
- $\|x\|_\infty = \max_{i=1,\ldots,n} |x_i|$ (∞-norma ali max norma).

Vse tri so posebni primeri Hölderjeve p-norme $\|x\|_p = (\sum_{i=1}^{n} |x_i|^p)^{1/p}$, kjer je $1 \leq p \leq \infty$. Hölderjeva neenakost pravi

$$|x^* y| \leq \|x\|_p \|y\|_q, \quad \frac{1}{p} + \frac{1}{q} = 1, \quad 1 \leq p, q \leq \infty.$$

Posebni primer je Cauchy-Schwartzeva neenakost $|x^* y| \leq \|x\|_2 \|y\|_2$.

Poljubni dve vektorski normi $\|.\|_a$ in $\|.\|_b$ sta ekvivalentni, kar pomeni, da obstajata konstanti $C_1, C_2 > 0$, da za vsak $x \in \mathbb{C}^n$ velja

$$C_1 \|x\|_a \leq \|x\|_b \leq C_2 \|x\|_a.$$

Za najpogostejše norme veljajo ocene:

$$\begin{array}{ccccc} \|x\|_2 & \leq & \|x\|_1 & \leq & \sqrt{n}\|x\|_2 \\ \|x\|_\infty & \leq & \|x\|_2 & \leq & \sqrt{n}\|x\|_\infty \\ \|x\|_\infty & \leq & \|x\|_1 & \leq & n\|x\|_\infty \end{array}$$

Definicija 3 *Matrična norma je preslikava* $\|.\| : \mathbb{C}^{n\times n} \to \mathbb{R}$, *za katero velja*

1. $\|A\| \geq 0$, $\|A\| = 0 \iff A = 0$,
2. $\|\alpha A\| = |\alpha| \cdot \|A\|$,
3. $\|A + B\| \leq \|A\| + \|B\|$ *(trikotniška neenakost)*,
4. $\|AB\| \leq \|A\| \cdot \|B\|$ *(submultiplikativnost)*,

za vsak $A, B \in \mathbb{C}^{n\times n}$ *in* $\alpha \in \mathbb{C}$.

Ker je matrična norma vektorska norma plus submultiplikativnost, ne moremo vseh vektorskih norm razširiti na matrične. Če definiramo:

$$\begin{aligned} N_1(A) &:= \sum_{i,j=1}^{n} |a_{ij}|, \\ N_2(A) &:= (\sum_{i,j=1}^{n} |a_{ij}|^2)^{\frac{1}{2}}, \\ N_\infty(A) &:= \max_{i,j=1,\ldots,n} |a_{ij}|, \end{aligned}$$

se izkaže, da N_∞ ni matrična norma, saj ne izpolnjuje pogoja submultiplikativnosti. Ostali dve sta. N_1 se ne uporablja, $N_2(A)$ pa je *Frobeniusova norma* $\|A\|_F$.

Pomembnejše so *operatorske oz. inducirane norme*, ki so definirane z

$$\|A\| := \max_{\substack{x\neq 0 \\ x\in\mathbb{C}^n}} \frac{\|Ax\|}{\|x\|} = \max_{\|x\|=1} \|Ax\|.$$

Za različno izbiro vektorskih norm dobimo naslednje matrične norme: $\|A\|_1 := \max_{\substack{x\neq 0 \\ x\in\mathbb{C}^n}} \frac{\|Ax\|_1}{\|x\|_1}$, $\|A\|_2 := \max_{\substack{x\neq 0 \\ x\in\mathbb{C}^n}} \frac{\|Ax\|_2}{\|x\|_2}$, $\|A\|_\infty := \max_{\substack{x\neq 0 \\ x\in\mathbb{C}^n}} \frac{\|Ax\|_\infty}{\|x\|_\infty}$, ki pa se jih da izraziti na lepši način.

Lema 2 $\|A\|_1 = \max_{j=1,\ldots,n} (\sum_{i=1,\ldots,n} |a_{ij}|)$ *(1-norma).*

Dokaz. $A = [a_1 \ \cdots \ a_n]$, $Ax = \sum_{i=1}^{n} x_i a_i$.

$$\|Ax\|_1 = \|\sum_{i=1}^{n} x_i a_i\|_1 \leq \sum_{i=1}^{n} |x_i| \|a_i\|_1 \leq \max_{k=1,\ldots,n} \|a_k\|_1 \sum_{i=1}^{n} |x_i| = \max_{k=1,\ldots,n} \|a_k\|_1 \|x\|_1.$$

To pomeni $\|A\|_1 \leq \max_{k=1,\ldots,n} \|a_k\|_1$, enakost pa je dosežena, če vzamemo $x = e_j$, kjer je $\|a_j\|_1 \geq \|a_k\|_1$ za $k = 1, \ldots, n$. ■

Za poljubno matriko $A \in \mathbb{C}^{n\times n}$ je matrika $B = A^*A$ hermitska in nenegativno definitna, saj velja $B^* = B$ in $x^*Bx \geq 0$ za vsak $x \in \mathbb{C}^n$. Odtod sledi, da so vse lastne vrednosti B nenegativne in jih lahko zapišemo urejene po velikosti kot $\sigma_1^2 \geq \sigma_2^2 \geq \cdots \geq \sigma_n^2 \geq 0$. Pozitivne kvadratne korene $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_n \geq 0$ lastnih vrednosti A^*A imenujemo *singularne vrednosti* matrike A.

Lema 3 $\|A\|_2 = \sigma_1(A) = \max\limits_{i=1,\ldots,n} \sqrt{\lambda_i(A^*A)}$ *(spektralna norma).*

Dokaz. Obstaja ortonormirana baza za $\mathbb{C}^n$, ki jo sestavljajo lastni vektorji $A^*Au_i = \sigma_i^2 u_i$, $i = 1, \ldots, n$. Če vektor x zapišemo kot $x = \sum_{i=1}^n \alpha_i u_i$ dobimo

$$\|Ax\|_2^2 = x^*(A^*Ax) = (\sum_{i=1}^n \alpha_i u_i)^*(\sum_{i=1}^n \alpha_i \sigma_i^2 u_i) = \sum_{i=1}^n |\alpha_i|^2 \sigma_i^2 \le \sigma_1^2 \sum_{i=1}^n |\alpha_i|^2 = \sigma_1^2 \|x\|_2^2.$$

To pomeni $\|A\|_2 \le \sigma_1(A)$, enakost pa je dosežena pri $x = u_1$. ■

Lema 4 $\|A\|_\infty = \max\limits_{i=1,\ldots,n} (\sum\limits_{j=1,\ldots,n} |a_{ij}|)$ *(∞-norma).*

Dokaz. Za $A = \begin{bmatrix} \alpha_1^T \\ \vdots \\ \alpha_n^T \end{bmatrix}$ dobimo

$$\|Ax\|_\infty = \max_{i=1,\ldots,n} |\alpha_i^T x| \le \max_{i=1,\ldots,n} \|x\|_\infty \|\alpha_i\|_1.$$

To pomeni $\|A\|_\infty \le \max_{i=1,\ldots,n} \|\alpha_i\|_1$, enakost pa je dosežena pri

$$x_k = \begin{cases} \frac{\overline{a_{jk}}}{|a_{jk}|} & \text{za } a_{jk} \neq 0 \\ 0 & \text{sicer,} \end{cases}$$

kjer je $\|\alpha_j\|_1 \ge \|\alpha_k\|_1$ za $k = 1, \ldots, n$. ■

Zgled 15 *Za* $A = \begin{bmatrix} 3 & -1 & 2 \\ 4 & 1 & -8 \\ 1 & -5 & 0 \end{bmatrix}$ *dobimo* $\|A\|_1 = 10$, $\|A\|_\infty = 13$, $\|A\|_F = 11$ *in* $\|A\|_2 = 9.02316$. ■

Za operatorske norme očitno velja $\|I\| = 1$, medtem ko je $\|I\|_F = \sqrt{n}$.

Tako kot vektorske norme so tudi matrične norme medsebojno ekivalentne. Pomebne so naslednje ocene, s katerimi lahko ocenimo $\|A\|_2$ z lažje izračunljivimi normami:

$$\begin{array}{rcccl} \frac{1}{\sqrt{n}}\|A\|_F & \le & \|A\|_2 & \le & \|A\|_F \\ \frac{1}{\sqrt{n}}\|A\|_1 & \le & \|A\|_2 & \le & \sqrt{n}\|A\|_1 \\ \frac{1}{\sqrt{n}}\|A\|_\infty & \le & \|A\|_2 & \le & \sqrt{n}\|A\|_\infty \\ N_\infty(A) & \le & \|A\|_2 & \le & nN_\infty(A) \\ & & \|A\|_2 & \le & \sqrt{\|A\|_1\|A\|_\infty} \end{array}$$

Za matrično normo $\|.\|_M$ pravimo, da je *usklajena* z vektorsko normo $\|.\|_V$, če za vsako matriko A in za vsak vektor x velja

$$\|Ax\|_v \le \|A\|_M \cdot \|x\|_V.$$

Lema 5 *Za vsako matrično normo obstaja taka vektorska norma, ki je z njo usklajena.*

Dokaz. Naj bo M preslikava, ki vektorju x priredi matriko $M(x) = [x\ 0\ \cdots\ 0]$. Sedaj definiramo vektorsko normo $\|x\|_V := \|M(x)\|_M$. Dobimo

$$\|Ax\|_V = \|[Ax\ 0\ \cdots\ 0]\|_M = \|A[x\ 0\ \cdots\ 0]\|_M \leq \|A\|_M \cdot \|M(x)\|_M = \|A\|_M \cdot \|x\|_V. \quad \blacksquare$$

Posledica 1 *Za vsako matrično normo in poljubno lastno vrednost λ matrike A velja*

$$|\lambda| \leq \|A\|.$$

Dokaz. Naj bo vektorska norma usklajena z matrično in $Ax = \lambda x$ lastni par, $x \neq 0$. Dobimo

$$|\lambda| \cdot \|x\| = \|\lambda x\| = \|Ax\| \leq \|A\| \cdot \|x\|. \quad \blacksquare$$

Realna matrika Q je *ortogonalna*, če je $Q^{-1} = Q^T$. Kompleksna matrika U je *unitarna*, če je $U^{-1} = U^*$. Vemo, da za množenje z unitarno matriko velja $\|Ux\|_2 = \|x\|_2$.

Lema 6 *Frobeniusova in spektralna norma sta invariantni na množenje z unitarno matriko.*

Dokaz. Naj bo $A = [a_1\ \cdots\ a_n]$. Velja

$$\|UA\|_F^2 = \sum_{i=1}^{n} \|Ua_i\|_2^2 = \sum_{i=1}^{n} \|a_i\|_2^2 = \|A\|_F^2.$$

Za spektralno normo dobimo

$$\|UA\|_2 = \max_{\|x\|_2=1} \|UAx\|_2 = \max_{\|x\|_2=1} \|Ax\|_2 = \|A\|_2. \quad \blacksquare$$

3.3 Občutljivost linearnih sistemov

Lema 7 *Če je $\|X\| < 1$ in $\|I\| = 1$, potem je $I + X$ nesingularna matrika, $(I - X)^{-1} = \sum_{i=1}^{\infty} X^i$ in $\|(I - X)^{-1}\| \leq \frac{1}{1-\|X\|}$.*

Dokaz. Če bi bila $I + X$ singularna, bi obstajal tak vektor z, da je $(I + X)z = 0$. Potem bi bilo $z = -Xz$ in $\|z\| = \|Xz\| \leq \|X\| \cdot \|z\|$, torej $\|X\| \geq 1$.

Vrsta $\sum_{i=1}^{\infty} X^i$ je zaradi $\|X\| < 1$ konvergentna, enakost pa dobimo, če vrsto pomnožimo z $I - X$. Iz vrste sledi ocena $\|\sum_{i=1}^{\infty} X^i\| \leq \sum_{i=1}^{\infty} \|X\|^i = \frac{1}{1-\|X\|}$. $\blacksquare$

Zanima nas, kako je rešitev linearnega sistema $Ax = b$ občutljiva na spremembe A in b. Naj bo $Ax = b$ in $(A + \delta A)(x + \delta x) = b + \delta b$. Od tod dobimo

$$\delta x = (A + \delta A)^{-1}(-\delta Ax + \delta b) = (I + A^{-1}\delta A)^{-1}A^{-1}(-\delta Ax + \delta b).$$

Če predpostavimo, da je $\|A^{-1}\| \cdot \|\delta A\| < 1$, potem je $I + A^{-1}\delta A$ nesingularna in vemo

$$\|(I + A^{-1}\delta A)^{-1}\| \leq \frac{1}{1 - \|A^{-1}\delta A\|} \leq \frac{1}{1 - \|A^{-1}\| \cdot \|\delta A\|}.$$

Dobimo

$$\begin{aligned} \|\delta x\| &\leq \frac{\|A^{-1}\|}{1 - \|A^{-1}\| \cdot \|\delta A\|}(\|\delta A\| \cdot \|x\| + \|\delta b\|) \\ \frac{\|\delta x\|}{\|x\|} &\leq \frac{\|A^{-1}\|}{1 - \|A^{-1}\| \cdot \|\delta A\|}\left(\frac{\|\delta A\|}{\|A\|} \cdot \|A\| + \frac{\|\delta b\| \cdot \|A\|}{\|x\| \cdot \|A\|}\right) \\ &\leq \frac{\|A^{-1}\| \cdot \|A\|}{1 - \|A^{-1}\| \cdot \|A\| \cdot \frac{\|\delta A\|}{\|A\|}}\left(\frac{\|\delta A\|}{\|A\|} + \frac{\|\delta b\|}{\|b\|}\right). \end{aligned}$$

Število $\kappa(A) := \|A^{-1}\| \cdot \|A\|$ imenujemo *občutljivost oz. pogojenostno število* matrike A. Končna ocena je

$$\frac{\|\delta x\|}{\|x\|} \leq \frac{\kappa(A)}{1 - \kappa(A)\frac{\|\delta A\|}{\|A\|}}\left(\frac{\|\delta A\|}{\|A\|} + \frac{\|\delta b\|}{\|b\|}\right).$$

Izrek 8 *Če je matrika A nesingularna, potem je*

$$\min\left\{\frac{\|\delta A\|_2}{\|A\|_2};\ A + \delta A \text{ singularna}\right\} = \frac{1}{\|A^{-1}\|_2\|A\|_2} = \frac{1}{\kappa_2(A)}. \quad \blacksquare$$

Občutljivost je tako recipročna oddaljenosti od singularnega problema.

Za občutljivost velja $1 \leq \kappa(A)$, saj je $1 \leq \|I\| = \|AA^{-1}\| \leq \|A\|\|A^{-1}\|$. Edine matrike, ki imajo občutljivost 1, so z neničelnim skalarjem pomnožene unitarne matrike.

Za spektralno občutljivost velja

$$\kappa_2(A) = \frac{\sigma_1(A)}{\sigma_n(A)}.$$

Primeri zelo občutljivih matrik so Hilbertove matrike H_n, kjer je $h_{ij} = \frac{1}{i+j-1}$. Velja npr. $\kappa(H_4) = 1.6 \cdot 10^4$, $\kappa(H_7) = 4.8 \cdot 10^8$ in $\kappa(H_{10}) = 1.6 \cdot 10^{13}$.

3.4 Permutacijske matrike in elementarne eliminacije

3.4.1 Permutacijske matrike

Naj bo $A = \begin{bmatrix} \alpha_1^T \\ \vdots \\ \alpha_n^T \end{bmatrix}$ matrika, zapisana po vrsticah, $\sigma = \begin{pmatrix} 1 & 2 & \cdots & n \\ \sigma_1 & \sigma_2 & \cdots & \sigma_n \end{pmatrix}$ pa permutacija, s katero želimo urediti vrstice A. To naredimo tako, da A z leve pomnožimo s *permutacijsko*

matriko $P_\sigma = \begin{bmatrix} e_{\sigma_1}^T \\ \vdots \\ e_{\sigma_n}^T \end{bmatrix}$. Dobljena matrika $P_\sigma A = \begin{bmatrix} \alpha_{\sigma_1}^T \\ \vdots \\ \alpha_{\sigma_n^T} \end{bmatrix}$ ima s permutacijo σ urejene vrstice matrike A.

Če želimo stolpce matrike $A = [\, a_1 \;\; \cdots \;\; a_n \,]$ urediti s permutacijo σ, potem matriko A z desne pomnožimo z matriko P_σ^T, saj je $(P_\sigma A^T)^T = A P_\sigma^T$.

3.4.2 Elementarne eliminacije

Denimo, da imamo tak vektor $x \in \mathbb{R}^n$, da je $x_k \neq 0$. Iščemo nesingularno matriko L_k, da bo

$$L_k \begin{bmatrix} x_1 \\ \vdots \\ x_k \\ x_{k+1} \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} x_1 \\ \vdots \\ x_k \\ 0 \\ \vdots \\ 0 \end{bmatrix}.$$

Če definiramo $l_{jk} = \frac{x_j}{x_k}$, $j = k+1, \ldots, n$, potem je

$$L_k = \begin{bmatrix} 1 & & & & \\ & \ddots & & & \\ & & 1 & & \\ & & -l_{k+1,k} & 1 & \\ & & \vdots & & \ddots \\ & & -l_{n,k} & & 1 \end{bmatrix}$$

ustrezna matrika. Matriko L_k imenujemo *elementarna eliminacija*, zapišemo pa jo lahko tudi kot $L_k = I - l_k e_k^T$, kjer je

$$l_k = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ l_{k+1,k} \\ \vdots \\ l_{n,k} \end{bmatrix}.$$

Inverz matrike L_k je

$$L_k^{-1} = I + l_k e_k^T = \begin{bmatrix} 1 & & & & \\ & \ddots & & & \\ & & 1 & & \\ & & l_{k+1,k} & 1 & \\ & & \vdots & & \ddots \\ & & l_{n,k} & & 1 \end{bmatrix}.$$

Ker za $i < j$ velja $(I + l_i e_i^T)(I + l_j e_j^T) = I + l_i e_i^T + l_j e_j^T$, je produkt matrik $L_1^{-1} L_2^{-1} \cdots L_{n-1}^{-1}$ enak

$$\begin{bmatrix} 1 & & & & \\ l_{21} & 1 & & & \\ l_{31} & l_{32} & 1 & & \\ \vdots & \vdots & \ddots & \ddots & \\ l_{n1} & l_{n2} & \cdots & l_{n,n-1} & 1 \end{bmatrix}.$$

3.5 LU razcep

Večina algoritmov za reševanje matričnih problemov deluje na naslednjem principu:

1) Problem najprej prevedemo na enostavnejši problem, ki ima posebno obliko.
2) Ekonomično rešimo enostavnejši problem, pri čemer si pomagamo s posebno strukturo.
3) Iz rešitve enostavnejšega problema dobimo rešitev originalnega problema.

Enostavnejše oblike:

- reševanje linearnega sistema: trikotna oblika,
- računanje lastnih vrednosti: Hessenbergova ali tridiagonalna oblika,
- računanje singularnih vrednosti: bidiagonalna oblika.

Do redukcije na enostavnejšo obliko pridemo s pomočjo permutacij, elementarnih eliminacij ali ortogonalnih transformacij (rotacije oz. zrcaljenja).

Poglejmo, kako z elementarnimi eliminacijami matriko A zapišemo v obliki $A = LU$, kjer je L spodnja trikotna matrika z enicami na diagonali, U pa zgornja trikotna matrika. Naj bo

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix}$$

in $a_{11} \neq 0$. Za eliminacijsko matriko

$$L_1 = \begin{bmatrix} 1 & & & \\ -l_{21} & 1 & & \\ \vdots & & \ddots & \\ -l_{n1} & & & 1 \end{bmatrix},$$

kjer je $l_{21} = \frac{a_{21}}{a_{11}}$, $\ldots$, $l_{n1} = \frac{a_{n1}}{a_{11}}$ velja

$$L_1 \begin{bmatrix} a_{11} \\ a_{21} \\ \vdots \\ a_{n1} \end{bmatrix} = \begin{bmatrix} a_{11} \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \quad \text{torej} \quad A^{(1)} := L_1 A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ 0 & a_{22}^{(1)} & \cdots & a_{2n}^{(1)} \\ \vdots & \vdots & & \vdots \\ 0 & a_{n2}^{(1)} & \cdots & a_{nn}^{(1)} \end{bmatrix}.$$

Sedaj je

$$A^{(2)} := L_2 A^{(1)} = \begin{bmatrix} a_{11} & a_{12} & a_{13} & \cdots & a_{1n} \\ 0 & a_{22}^{(1)} & a_{23}^{(1)} & \cdots & a_{2n}^{(1)} \\ 0 & 0 & a_{33}^{(2)} & \cdots & a_{3n}^{(2)} \\ \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & a_{n3}^{(2)} & \cdots & a_{nn}^{(2)} \end{bmatrix},$$

kjer je

$$L_2 = \begin{bmatrix} 1 & & & & \\ & 1 & & & \\ & -l_{32} & 1 & & \\ & \vdots & & \ddots & \\ & -l_{n2} & & & 1 \end{bmatrix},$$

in $l_{32} = \frac{a_{32}^{(1)}}{a_{22}^{(1)}}, \ldots, l_{n2} = \frac{a_{n2}^{(1)}}{a_{22}^{(1)}}$. Na koncu dobimo

$$U := \underbrace{L_{n-1} \cdots L_2 L_1}_{=L^{-1}} A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ & a_{22}^{(1)} & \cdots & a_{2n}^{(1)} \\ & & \ddots & \vdots \\ & & & a_{nn}^{(n-1)} \end{bmatrix},$$

in

$$L = L_1^{-1} L_2^{-1} \cdots L_{n-1}^{-1} = \begin{bmatrix} 1 & & & & \\ l_{21} & 1 & & & \\ l_{31} & l_{32} & 1 & & \\ \vdots & \vdots & \ddots & \ddots & \\ l_{n1} & l_{n2} & \cdots & l_{n,n-1} & 1 \end{bmatrix}.$$

Tako smo dobili $A = LU$. Dobljeni razcep imenujemo *LU razcep brez pivotiranja* ali *Gaussov razcep brez pivotiranja.*

Diagonalni elementi $a_{11}, a_{22}^{(1)}, \ldots, a_{n-1,n-1}^{(n-2)}$, s katerimi delimo, se imenujejo *pivoti*, l_{ij} pa *kvocienti*. Med samim procesom smo opazili, da morajo biti pivoti neničelni, sicer metoda odpove. Naslednji izrek pove, kdaj lahko razcep izračunamo brez težav.

Izrek 9 *Za matriko A je ekvivalentno:*

1) *Obstaja enolični razcep $A = LU$, kjer je L spodnja trikotna matrika z enicami na diagonali in U nesingularna zgornja trikotna matrika.*

2) *Vsi vodilne podmatrike $A(1:k, 1:k)$ so nesingularne.*

Dokaz. Pokažimo, da iz 1) sledi 2). Razcep $A = LU$ za poljubno vodilno podmatriko A_{11} bločno zapišemo

$$\begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} = \begin{bmatrix} L_{11} & 0 \\ L_{21} & L_{22} \end{bmatrix} \begin{bmatrix} U_{11} & U_{12} \\ 0 & U_{22} \end{bmatrix} = \begin{bmatrix} L_{11}U_{11} & L_{11}U_{12} \\ L_{21}U_{11} & L_{21}U_{12} + L_{22}U_{22} \end{bmatrix},$$

od koder sledi $A_{11} = L_{11}U_{11}$. Dobimo $\det(A_{11}) = \det(U_{11}) \neq 0$, saj je U nesingularna.

Za dokaz v obratni smeri uporabimo indukcijo. Pri matrikah 1×1 ni težav, saj je $a_{11} = 1\cdot a_{11}$. Naj bo $A = LU$ in $\widetilde{A} = \begin{bmatrix} A & b \\ c^T & \delta \end{bmatrix}$. Razcep za $\widetilde{A}$ mora imeti obliko

$$\begin{bmatrix} A & b \\ c^T & \delta \end{bmatrix} = \begin{bmatrix} L & 0 \\ l^T & 1 \end{bmatrix} \begin{bmatrix} U & u \\ 0 & \eta \end{bmatrix} = \begin{bmatrix} LU & Lu \\ l^T U & l^T u + \eta \end{bmatrix}.$$

Ker sta L in U nesingularni, dobimo $u = L^{-1}b$, $l = U^{-T}c$ in $\eta = \delta - l^T u$. Pri tem mora biti $\eta \neq 0$, saj je $0 \neq \det A = \eta \cdot \det U$. ■

Zgled 16 *Izračunajmo LU razcep za* $A = \begin{bmatrix} 2 & 2 & 3 \\ 4 & 5 & 6 \\ 1 & 2 & 4 \end{bmatrix}$.

$$A^{(1)} = \underbrace{\begin{bmatrix} 1 & 0 & 0 \\ -2 & 1 & 0 \\ -\frac{1}{2} & 0 & 1 \end{bmatrix}}_{L_1} \cdot A = \begin{bmatrix} 2 & 2 & 3 \\ 0 & 1 & 0 \\ 0 & 1 & \frac{5}{2} \end{bmatrix}$$

$$A^{(2)} = \underbrace{\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -1 & 1 \end{bmatrix}}_{L_2} \cdot A^{(1)} = \begin{bmatrix} 2 & 2 & 3 \\ 0 & 1 & 0 \\ 0 & 0 & \frac{5}{2} \end{bmatrix} = U.$$

$$L = L_1^{-1} L_2^{-1} = \begin{bmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ \frac{1}{2} & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ \frac{1}{2} & 1 & 1 \end{bmatrix}. \quad ■$$

Algoritem lahko zapišemo v obliki

$j = 1, \ldots, n-1$
  $i = j+1, \ldots, n$
    $l_{ij} = \frac{a_{ij}}{a_{jj}}$
    $k = j+1, \ldots, n$
      $a_{ik} = a_{ik} - l_{ij} a_{jk}$

Pri matriki L na koncu manjka še I, v zgornjem trikotniku A pa na koncu ostane U. Elemente L lahko shranjujemo v spodnji trikotnik A in tako ne potrebujemo dodatnega prostora.

Število operacij:

$$\begin{aligned} \sum_{j=1}^{n-1} \sum_{i=j+1}^{n} \left(1 + \sum_{k=j+1}^{n} 2 \right) &= \sum_{j=1}^{n-1} (n-j)(1+2(n-j)) = \\ &= \sum_{j=1}^{n-1} \left(2(n-j)^2 + n - j \right) = \sum_{l=1}^{n-1} (2l^2 + l) = \\ &= 2\frac{(n-1)n(2n-1)}{6} + \frac{(n-1)n}{2} = \\ &= \frac{2}{3}n^3 - \frac{1}{2}n^2 - \frac{1}{6}n = \frac{2}{3}n^3 + \mathcal{O}(n^2). \end{aligned}$$

Zgled 17 *Metoda odpove, če je pivot enak 0, numerično pa odpove tudi, če je pivot blizu 0. Če na tri decimalke točno računamo LU razcep* $A = \begin{bmatrix} 0.0001 & 1 \\ 1 & 1 \end{bmatrix}$, *dobimo* $L = \begin{bmatrix} 1 & 0 \\ 10000 & 1 \end{bmatrix}$ *in* $U = \begin{bmatrix} 0.0001 & 1 \\ 0 & fl(1-10000) \end{bmatrix} = \begin{bmatrix} 0.0001 & 1 \\ 0 & -10000 \end{bmatrix}$. *Velja* $LU = \begin{bmatrix} 0.0001 & 1 \\ 1 & 0 \end{bmatrix} \neq A$, *napaka pa je ogromna.* ■

Rešitev obeh težav je pivotiranje, kjer med algoritmom dopuščamo zamenjavo vrstic (*delno pivotiranje*), lahko pa tudi stolpcev (*kompletno pivotiranje*).

Pri delnem pivotiranju pred eliminacijo v j-tem stolpcu primerjamo $|a_{jj}|, |a_{j+1,j}|, \ldots, |a_{nj}|$ in zamenjamo j-to vrstico s tisto, ki vsebuje maksimalni element. Tako je pri nesingularni matriki v vsakem koraku pivot neničelen.

Kot rezultat dobimo $PA = LU$, kjer je P permutacijska matrika. Zaradi pivotiranja so v matriki L vsi elementi po absolutni vrednosti omejeni z 1.

Izrek 10 *Če je A nesingularna, potem obstaja taka permutacijska matrika P, da obstaja LU razcep $PA = LU$, kjer je L spodnja trikotna matrika z enicami na diagonali in U zgornja trikotna matrika.*

Dokaz. Pred uničevanjem elementov v j-tem stolpcu je situacija naslednja:

$$A^{(j-1)} = \begin{bmatrix} U_{11} & U_{12} \\ 0 & A_{22}^{(j-1)} \end{bmatrix}.$$

Ker je matrika A nesingularna, je nesingularna tudi $A^{(j-1)}$, to pa pomeni, da mora biti $A_{22}^{(j-1)}$ nesingularna in ne morejo biti vsi elementi v prvem stolpcu $A_{22}^{(j-1)}$ hkrati enaki 0. ■

Algoritem za LU razcep z delnim pivotiranjem je:

$j = 1, \ldots, n-1$
  poišči $|a_{qj}| = \max_{j \le p \le n} |a_{pj}|$
  zamenjaj vrstici q in j
  $i = j+1, \ldots, n$
    $l_{ij} = \frac{a_{ij}}{a_{jj}}$
    $k = j+1, \ldots, n$
      $a_{ik} = a_{ik} - l_{ij} a_{jk}$

Dodatno delo je $\mathcal{O}(n^2)$ primerjanj.

Zgled 18 *Izračunajmo LU razcep z delnim pivotiranjem za* $A = \begin{bmatrix} 0 & 1 & 2 \\ 1 & 2 & 3 \\ 1 & 0 & 1 \end{bmatrix}$. *Na začetku vzamemo $P = I$, potem pa vsakič, ko zamenjamo vrstici, zamenjavo naredimo tudi v P. Matriko L hranimo v spodnjem trikotniku A.*

$$A^{(1)} = \begin{bmatrix} 1 & 2 & 3 \\ \boxed{0} & 1 & 2 \\ \boxed{1} & -2 & -2 \end{bmatrix}, \quad P = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix},$$

$$A^{(2)} = \begin{bmatrix} 1 & 2 & 3 \\ \boxed{1} & -2 & -2 \\ \boxed{0} & \boxed{-\frac{1}{2}} & 1 \end{bmatrix}, \quad P = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix}.$$

Uokvirjeni elementi spadajo v matriko L, na njihovih mestih v A pa so ničle. Dobili smo $L = \begin{bmatrix} 1 & & \\ 1 & 1 & \\ 0 & -\frac{1}{2} & 1 \end{bmatrix}$, $U = \begin{bmatrix} 1 & 2 & 3 \\ & -2 & -2 \\ & & 1 \end{bmatrix}$ *in* $PA = LU$. ■

Reševanje sistema $Ax = b$:

1) $PA = LU$,
2) $Ly = Pb =: b'$,
3) $Ux = y$.

Sistem $Ly = b'$ rešujemo s *premo substitucijo*. Iz

$$\begin{bmatrix} 1 & & & \\ l_{21} & 1 & & \\ \vdots & \ddots & \ddots & \\ l_{n1} & \cdots & l_{n,n-1} & 1 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} = \begin{bmatrix} b'_1 \\ b'_2 \\ \vdots \\ b'_n \end{bmatrix}$$

dobimo

$$l_{i1}y_1 + \cdots + l_{i,i-1}y_{i-1} + y_i = b'_i, \quad i = 1, \ldots, n,$$

od tod pa algoritem

$i = 1, \ldots, n$
  $y_i = b'_i - \sum_{j=1}^{i-1} l_{ij}y_j$

Število operacij je $\sum_{i=1}^{n}(1 + 2(i-1)) = n^2$.

Sistem $Ux = y$ rešujemo z *obratno substitucijo*. Iz

$$\begin{bmatrix} u_{11} & \cdots & u_{1n} \\ & \ddots & \vdots \\ & & u_{nn} \end{bmatrix} \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix}$$

dobimo

$$u_{ii}x_i + u_{i,i+1}x_{i+1} + \cdots + u_{in}x_n = y_i, \quad i = 1, \ldots, n,$$

od tod pa algoritem

$i = n, n-1, \ldots, 1$
$\quad x_i = \frac{1}{u_{ii}} \left(y_i - \sum_{j=i+1}^{n} u_{ij} x_j \right)$

Število operacij je $\sum_{i=1}^{n} (2 + 2(i-1)) = n^2 + n$ (še deljenja na diagonali).

Za LU razcep z delnim pivotiranjem torej porabimo $\frac{2}{3}n^3 + \mathcal{O}(n^2)$ operacij, ko pa L in U že poznamo, za reševanje $Ax = b$ porabimo še dodatnih $2n^2 + \mathcal{O}(n)$ operacij.

Za reševanje sistema $Ax = b$ nikoli ne uporabljamo inverzne matrike A^{-1}, saj:

- za množenje $A^{-1}b$ porabimo $2n^2$ operacij, kar ni ceneje od reševanja trikotnih L in U;
- za računanje A^{-1} potrebujemo $2n^3$ operacij, kar je trikrat toliko kot LU razcep;
- numerične napake so kvečjemu večje.

Poleg LU razcepa z delnim pivotiranjem poznamo še *LU razcep s kompletnim pivotiranjem*, kjer v j-tem stolpcu pivotni element izbiramo iz cele podmatrike $A(j:n, j:n)$, nato pa izvedemo zamenjavo vrstic in stolpcev. Na koncu dobimo razcep $PAQ = LU$, kjer sta P in Q permutacijski matriki za vrstice oziroma stolpce. Število operacij je enako kot pri osnovnem LU razcepu, število primerjanj pa je $\mathcal{O}(n^3)$.

Pri kompletnem pivotiranju sistem $Ax = b$ rešujemo:

1) $PAQ = LU$,
2) $Ly = Pb =: b'$,
3) $Ux' = y$,
4) $x = Qx'$.

Zgled 19 *Sistem $Ax = b$, kjer sta $A = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 2 & 3 \\ 1 & 1 & 1 \end{bmatrix}$ in $b = \begin{bmatrix} 14 \\ 6 \\ 5 \end{bmatrix}$, bomo rešili preko LU razcepa s kompletnim pivotiranjem.*

$$A^{(1)} = \begin{bmatrix} 3 & 2 & 1 \\ \boxed{\frac{1}{3}} & \frac{1}{3} & -\frac{1}{3} \\ \boxed{\frac{1}{3}} & \frac{1}{3} & \frac{2}{3} \end{bmatrix}, \quad P = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad Q = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix},$$

$$A^{(2)} = \begin{bmatrix} 3 & 1 & 2 \\ \boxed{\frac{1}{3}} & \frac{2}{3} & \frac{1}{3} \\ \boxed{\frac{1}{3}} & \boxed{-\frac{1}{2}} & \frac{1}{2} \end{bmatrix}, \quad P = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix}, \quad Q = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix}.$$

Uokvirjeni elementi so iz L, v A pa so na njihovih mestih ničle. Razcep je $PAQ = LU$, kjer sta $L = \begin{bmatrix} 1 & & \\ \frac{1}{3} & 1 & \\ \frac{1}{3} & -\frac{1}{2} & 1 \end{bmatrix}$ in $U = \begin{bmatrix} 3 & 1 & 2 \\ & \frac{2}{3} & \frac{1}{3} \\ & & \frac{1}{2} \end{bmatrix}$.

Dobimo $b' = Pb = \begin{bmatrix} 14 \\ 6 \\ 5 \end{bmatrix}$, $y = \begin{bmatrix} 14 \\ \frac{4}{3} \\ 1 \end{bmatrix}$, $x' = \begin{bmatrix} 3 \\ 1 \\ 2 \end{bmatrix}$ *in* $x = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}$. ■

3.6 Analiza zaokrožitvenih napak pri LU razcepu

Vemo, da za $\odot : +, -, *, /$ velja $fl(a \odot b) = (a \odot b)(1+\delta)$, kjer je $|\delta| \leq \epsilon$. Uporabljali bomo tudi $fl(a \odot b) = \frac{a \odot b}{1+\delta}$, kjer je $|\delta| \leq \epsilon$.

Pri reševanju $Ax = b$ preko LU razcepa dobimo $\widehat{x}$. Pri analizi obratne stabilnosti iščemo oceno za δA, da je $(A + \delta A)\widehat{x} = b$. Pri oceni predpostavimo, da med računanjem ne pride do prekoračitve oz. podkoračitve.

Oznaka $|A|$ pomeni matriko $|A| = \begin{bmatrix} |a_{11}| & \cdots & |a_{1n}| \\ \vdots & & \vdots \\ |a_{n1}| & \cdots & |a_{nn}| \end{bmatrix}$, $A \leq B$ pa, da je $a_{ij} \leq b_{ij}$ za vsak i, j.

Lema 8 *Naj bo L spodnja trikotna matrika velikosti $n \times n$. Sistem $Lx = b$ rešimo s premo substitucijo. Izračunana rešitev $\widehat{x}$ zadošča enačbi $(L + \delta L)\widehat{x} = b$, kjer je $|\delta L| \leq n\epsilon|L|$.*

Dokaz. Prema substitucija je

$x_1 = \frac{b_1}{l_{11}}$
$i = 2, \ldots, n$
$\quad x_i = \frac{1}{l_{ii}} \left(b_i - \sum_{k=1}^{i-1} l_{ik} x_k \right)$

Vemo, da za računanje skalarnega produkta $s = \sum_{k=1}^{n} a_k b_k$ velja $\widehat{s} = \sum_{k=1}^{n} a_k b_k (1+g_k)$, kjer je $|g_k| \leq n\epsilon$. Torej je

$$\widehat{x}_i = \frac{b_i - \sum_{k=1}^{i-1} l_{ik}\widehat{x}_k(1+g_{ik})}{l_{ii}(1+h_i)(1+h_i')}, \quad i = 2, \ldots, n,$$

kjer je $g_{ik} \leq (i-1)\epsilon$ in $|h_i|, |h_i'| \leq \epsilon$ (odštevanje in deljenje). Zapišemo lahko

$$\sum_{k=1}^{i} l_{ik}\widehat{x}_k(1+\delta_{ik}) = b_i, \quad i = 2, \ldots, n,$$

kjer je $|\delta_{ik}| \leq i\epsilon \leq n\epsilon$. To velja tudi za $i = 1$, saj je $\widehat{x}_1 = \frac{b_1}{l_{11}(1+\delta_{11})}$. Dobili smo $(L + \delta L)\widehat{x} = b$, kjer je $\delta l_{ij} = l_{ij} \cdot \delta_{ik}$ in $|\delta_{ik}| \leq n\epsilon$. ■

Podobno pri reševanju zgornje trikotnega sistema $Ux = y$ velja $(U + \delta U)\widehat{x} = y$, kjer je $|\delta U| \leq n\epsilon|U|$.

Lema 9 *Naj bo A matrika velikosti $n \times n$, pri kateri se izvede LU razcep brez pivotiranja. Za izračunana $\widehat{L}$ in $\widehat{U}$ velja $A = \widehat{L}\widehat{U} + E$, kjer je $|E| \leq n\epsilon|\widehat{L}| \cdot |\widehat{U}|$.*

Dokaz. Ločimo primera $j \le k$ in $j > k$ (zgornji in spodnji trikotnik), ostalo je podobno kot pri dokazu leme 8. ■

Posledica je, da za izračunana L, U velja $A = LU + E$, kjer je $\|E\| \le n\epsilon \||L|\| \cdot \||U|\|$ za ∞, 1 ali F normo (kjer velja $\||A|\| = \|A\|$), npr.

$$\|E\|_\infty \le n\epsilon \|L\|_\infty \|U\|_\infty.$$

Izrek 11 *Za izračunano rešitev $\widehat{x}$ sistema $Ax = b$ preko LU razcepa velja $(A+\delta A)\widehat{x} = b$, kjer je $|\delta A| \le 3n\epsilon |L| \cdot |U|$ oziroma $\|\delta A\|_\infty \le 3n\epsilon \|L\|_\infty \|U\|_\infty$.*

Dokaz. Vemo:

- $A = LU + E$, kjer $|E| \le n\epsilon |A|$,
- $(L + \delta L)\widehat{y} = b$, kjer $|\delta L| \le n\epsilon |L|$,
- $(U + \delta U)\widehat{x} = \widehat{y}$, kjer $|\delta U| \le n\epsilon |U|$.

Dobimo:

$$\begin{aligned} b &= (L + \delta L)\widehat{y} = (L + \delta L)(U + \delta U)\widehat{x} = \\ &= (LU + \delta LU + L\delta U + \delta L\delta U)\widehat{x} = \\ &= (A - E + \delta LU + L\delta U + \delta L\delta U)\widehat{x} = (A + \delta A)\widehat{x}, \end{aligned}$$

kjer je $\delta A = -E + L\delta U + U\delta L + \delta L\delta U$. Ocena je:

$$\begin{aligned} |\delta A| &\le |E| + |L| \cdot |\delta U| + |\delta L| \cdot |U| + |\delta L| \cdot |\delta U| \le \\ &\le n\epsilon |L| \cdot |U| + n\epsilon |L| \cdot |U| + n\epsilon |L| \cdot |U| + n^2\epsilon^2 |L| \cdot |U| \approx 3n\epsilon |L| \cdot |U|. \end{aligned}$$ ■

Zanima nas, kdaj je $3n\epsilon \|L\|_\infty \|U\|_\infty = \mathcal{O}(\epsilon)\|A\|_\infty$, saj je potem metoda obratno stabilna.

Količino $g := \frac{\max |u_{ij}|}{\max |a_{ij}|}$ imenujemo *pivotna rast*. Sledi $\|U\|_\infty \le ng\|A\|_\infty$.

Pri delnem in kompletnem pivotiranju velja $|l_{ij}| \le 1$, posledica pa je $\|L\|_\infty \le n$. Tako za delno in kompletno pivotiranje velja

$$\|\delta A\|_\infty \le 3gn^3\epsilon \|A\|_\infty.$$

Ocena je lahko zelo velika, tudi če je $g = 1$. Skoraj vedno je ocena slabša od dejanskih rezultatov. Vemo še:

a) delno pivotiranje:

- **Lema 10** *Pri delnem pivotiranju je pivotna rast omejena z 2^{n-1}.*
  Dokaz. Zaradi $a_{jk} = a_{jk} - l_{ji}a_{ik}$ in $|l_{ij}| \le 1$ se lahko vrednost največjega elementa v matriki v vsakem koraku podvoji. Ker se vsakega elementa dotaknemo največ $(n-1)$-krat, je $g \le 2^{n-1}$. ■
- Običajno se g obnaša kot $n^{2/3}$.

- LU razcep z delnim pivotiranjem je v praksi obratno stabilen.

b) kompletno pivotiranje:

- **Lema 11** *Pri kompletnem pivotiranju je pivotna rast omejena z*

$$g \leq \left(n \cdot 2 \cdot 3^{1/2} \cdot 4^{1/3} \cdots n^{1/(n-1)}\right)^{\frac{1}{2}} \approx n^{\frac{1}{2}+\frac{\ln n}{4}}.$$

- Običajno se g obnaša kot $n^{1/2}$.
- Domneva (Wilkinson, 1965) je bila, da je $g \leq n$, vendar so leta 1991 našli protiprimer 13×13, kjer je $g = 13.02\ldots$.
- LU razcep s kompletnim pivotiranjem je obratno stabilen, ker pa porabi preveč primerjanj, ga le redko uporabljamo.

LU razcep brez pivotiranja razen za posebne matrike (npr. diagonalno dominantne po stolpcih, simetrične pozitivno definitne) ni obratno stabilen.

Naj bo $\widehat{x}$ izračunana rešitev, x pa točna rešitev sistema $Ax = b$. Označimo ostanek $r := A\widehat{x} - b$. Velja $\widehat{x} - x = A^{-1}r$, od tod pa dobimo oceno

$$\frac{\|\widehat{x} - x\|_\infty}{\|\widehat{x}\|_\infty} \leq \|A^{-1}\|_\infty \frac{\|r\|_\infty}{\|\widehat{x}\|_\infty}.$$

Namesto točne vrednosti $\|A^{-1}\|$ uporabimo algoritme, ki z uporabo $\mathcal{O}(n^2)$ operacij ocenijo (ponavadi dokaj dobro) $\|A^{-1}\|_\infty$.

Še boljša je ocena

$$\frac{\|\widehat{x} - x\|_\infty}{\|\widehat{x}\|_\infty} \leq \frac{\||A^{-1}| \cdot |r|\|_\infty}{\|\widehat{x}\|_\infty}.$$

Spet obstajajo algoritmi, ki ekonomično ocenijo $\||A^{-1}| \cdot |r|\|_\infty$ brez računanja A^{-1}.

3.7 Posebni sistemi

Kadar ima A posebno obliko, lahko prihranimo tako pri številu operacij kot pri porabi pomnilnika.

3.7.1 Simetrične pozitivno definitne matrike

$A \in \mathbb{R}^{n\times n}$ je *simetrična pozitivno definitna* (s.p.d.), če je $A = A^T$ in $x^T Ax > 0$ za vsak $x \neq 0$.

Izrek 12 *Velja:*

1) Naj bo $\det Y \neq 0$. *Potem je* A *s.p.d.* $\iff$ $Y^T AY$ *s.p.d.*

2) A *s.p.d. in* $H = A(1:k, 1:k)$ *poljubna vodilna podmatrika,* $k \leq n$, $\implies$ H *s.p.d.*

3) A *s.p.d. in* $H = A([i_1\ i_2\ \cdots\ i_k], [i_1\ i_2\ \cdots\ i_k])$ *poljubna podmatrika simetrična na diagonalo* $\implies$ H *s.p.d.*

4) A s.p.d. $\iff A = A^T$ in vse lastne vrednosti A so pozitivne.

5) A s.p.d. $\implies a_{ii} > 0$ za $\forall i$ in $\max_{i,j} |a_{ij}| = \max_i |a_{ii}|$.

6) A s.p.d. $\implies LU$ razcep brez pivotiranja se izvede in $u_{ii} > 0$ za $\forall i$.

7) A s.p.d. $\iff$ obstaja taka nesingularna spodnja trikotna matrika V s pozitivnimi elementi na diagonali, da je $A = VV^T$.

Razcep $A = VV^T$ imenujemo *razcep Choleskega*, V pa *faktor Choleskega*.

Dokaz.

1) $\det Y \neq 0 \implies Yx \neq 0$ za $x \neq 0$. Sedaj dobimo $x^T Y^T A Y x = (Yx)^T A (Yx) > 0$ za $y \neq 0$. V drugo smer uporabimo matriko Y^{-1}.

2) $x \in \mathbb{R}^k$ dopolnimo v $y \in \mathbb{R}^n$ kot $y = \begin{bmatrix} x \\ 0 \end{bmatrix}$. Potem je $x \neq 0 \iff y \neq 0$ in za $x \neq 0$ je $y^T A y = x^T H x > 0$.

3) Obstaja permutacija P, ki podmatriko H prestavi v vodilno podmatriko $A(1:k, 1:k)$. Uporabimo 1) in 2).

4) ($\implies$:) A je simetrična. Naj bo $Ax = \lambda x$, $x \neq 0$. Potem je $x^T A x = \lambda \|x\|_2 > 0 \implies \lambda > 0$.
($\impliedby$:) $A = QDQ^T$, $Q^T Q = I$, $D = \operatorname{diag}(\lambda_1, \ldots, \lambda_n)$. Za $x \neq 0$ je $x^T (QDQ^T) x = (Q^T x)^T D (Q^T x) > 0$, saj je $z^T D z = \sum_{i=1}^n \lambda_i z_i^2$.

5) $a_{ii} = e_i^T A e_i > 0$.
Denimo, da $|a_{pq}| = \max_{i,j} |a_{ij}|$ in $p \neq q$. Matrika $H := A([p\ q], [p\ q]) = \begin{bmatrix} a_{pp} & a_{pq} \\ a_{pq} & a_{qq} \end{bmatrix}$ mora biti s.p.d., torej $\det H > 0$. Toda $\det H = a_{pp} a_{qq} - a_{pq}^2 \leq 0$, to pa je protislovje.

6) Vse vodilne podmatrike A so nesingularne, torej obstaja LU razcep brez pivotiranja. Ker je $u_{11} u_{22} \cdots u_{kk} = \det(A(1:k, 1:k)) > 0$ za vsak k, mora biti $u_{ii} > 0$ za vsak i.

7) ($\impliedby$:) VV^T je očitno s.p.d. za nesingularno V.
($\implies$:) $A = LU = LDM$, kjer je $D = \operatorname{diag}(u_{11}, \ldots, u_{nn})$ in M zgornja trikotna z enicami na diagonali. Ker je $A = A^T$, sta $L(DM)$ in $M^T(DL^T)$ dva LU razcepa, ki pa je za nesingularno matriko enoličen. To pomeni $M = L^T$ in $A = LDL^T$. Če sedaj definiramo $V := LD^{1/2}$, kjer je $D^{1/2} = \operatorname{diag}(u_{11}^{1/2}, \ldots, u_{nn}^{1/2})$, dobimo iskani razcep $A = VV^T$. ■

Če iz $A = VV^T$ zapišemo enačbo za a_{jk}, $j \geq k$, dobimo

$$a_{jk} = \sum_{i=1}^{k} v_{ji} v_{ki} = \sum_{i-1}^{k-1} v_{ji} v_{ki} + v_{jk} v_{kk},$$

odtod pa algoritem za razcep Choleskega:

$k = 1, \ldots, n$
  $v_{kk} = \left(a_{kk} - \sum_{i=1}^{k-1} v_{ki}^2 \right)^{1/2}$

$j = k+1, \ldots, n$
$v_{jk} = \frac{1}{v_{kk}} \left(a_{jk} - \sum_{i=1}^{k-1} v_{ji} v_{ki} \right)$

Število operacij:

$$\sum_{k=1}^{n} (2k + 2(n-k)k) = \frac{1}{3} n^3 + \mathcal{O}(n^2).$$

Poleg polovice manj operacij porabimo tudi polovico manj prostora kot pri LU razcepu.

Če A ni s.p.d., se v algoritmu pod korenom pojavi nepozitivna vrednost. Računanje razcepa Choleskega je tako najcenejša metoda za ugotavljanje pozitivne definitnosti simetrične matrike.

Zgled 20 *Pri razcepu Choleskega za* $A = \begin{bmatrix} 4 & -2 & 4 & -2 & 4 \\ -2 & 10 & 1 & -5 & -5 \\ 4 & 1 & 9 & -2 & 1 \\ -2 & -5 & -2 & 22 & 7 \\ 4 & -5 & 1 & 7 & 14 \end{bmatrix}$ *dobimo*

$V = \begin{bmatrix} 2 & & & & \\ -1 & 3 & & & \\ 2 & 1 & 2 & & \\ -1 & -2 & 1 & 4 & \\ 2 & -1 & -1 & 2 & 2 \end{bmatrix}$. ■

Reševanje s.p.d. sistema $Ax = b$:

1) $A = VV^T$,
2) $Vy = b$,
3) $V^T x = y$.

Iz podobne analize, kot smo jo naredili za LU, sledi, da izračunana rešitev $\widetilde{x}$ zadošča $(A + \delta A)\widetilde{x} = b$, kjer je $|\delta A| \leq 3n\epsilon |V| \cdot |V^T|$. Ker pa je

$$(|V| \cdot |V^T|)_{ij} = \sum_k |v_{ik}||v_{jk}| \leq (\sum_k |v_{ik}|^2)^{1/2} (\sum_k |v_{jk}|^2)^{1/2} = \sqrt{a_{ii}}\sqrt{a_{jj}} \leq \max_{i,j} |a_{ij}|,$$

velja $\| |V| \cdot |V^T| \|_\infty \leq n \|A\|_\infty$ in

$$\|\delta A\|_\infty \leq 3n^2 \epsilon \|A\|_\infty.$$

To pomeni, da je razcep Choleskega numerično stabilen in da pri Choleskem ne potrebujemo pivotiranja.

3.7.2 Simetrične nedefinitne matrike

Pri simetrični matriki ne želimo uporabljati LU razcepa, saj ne ohranja simetrije. Za nesingularno A obstaja razcep $PAP^T = LDL^T$, kjer je L spodnja trikotna matrika z enicami na

diagonali, D pa bločno diagonalna matrika z bloki 1×1 ali 2×2. Število operacij za razcep je $\frac{n^3}{3} + \mathcal{O}(n^2)$.

Zgled za to, da potrebujemo 2×2 bloke v D je npr. $A = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$.

3.7.3 Tridiagonalne matrike

Če je $A = \begin{bmatrix} a_1 & b_1 & & & \\ c_2 & a_2 & b_2 & & \\ & \ddots & \ddots & \ddots & \\ & & c_{n-1} & a_{n-1} & b_{n-1} \\ & & & c_n & a_n \end{bmatrix}$, potem pri LU razcepu brez pivotiranja dobimo $L = \begin{bmatrix} 1 & & & \\ l_2 & 1 & & \\ & \ddots & \ddots & \\ & & l_n & 1 \end{bmatrix}$ in $U = \begin{bmatrix} u_1 & b_1 & & \\ & \ddots & \ddots & \\ & & u_{n-1} & b_n \\ & & & u_n \end{bmatrix}$. Za razcep in nadaljnje reševanje sistema $Ax = b$ potrebujemo $\mathcal{O}(n)$ operacij in $\mathcal{O}(n)$ prostora, saj shranimo le neničelne diagonale matrik A, L in U.

Pri delnem pivotiranju dobimo $U = \begin{bmatrix} u_1 & v_1 & w_1 & & \\ & \ddots & \ddots & \ddots & \\ & & u_{n-2} & v_{n-2} & w_{n-2} \\ & & & u_{n-1} & v_{n-1} \\ & & & & u_n \end{bmatrix}$, pivotna rast pa je omejena z 2. To pomeni, da je reševanje tridiagonalnega sistema preko LU razcepa z delnim pivotiranjem obratno stabilno.

Podobno velja za pasovne matrike, ki imajo poleg glavne še p diagonal nad in q diagonal pod glavno diagonalo.

3.7.4 Vandermondove matrike

Vandermondova matrika ima obliko

$$V = \begin{bmatrix} 1 & 1 & \cdots & 1 \\ x_0 & x_1 & \cdots & x_n \\ \vdots & \vdots & & \vdots \\ x_0^n & x_1^n & \cdots & x_n^n \end{bmatrix}.$$

Če rešujemo sistem $V^T a = y$, potem iščemo koeficiente polinoma $p(x) = a_0 + a_1 x + \cdots + a_n x^n$, za katerega velja $p(x_i) = y_i$, $i = 0, 1, \ldots, n$. Interpolacijski problem lahko ekonomično rešimo preko deljenih diferenc in porabimo $\mathcal{O}(n^2)$ namesto $\mathcal{O}(n^3)$ operacij. Podobno lahko ekonomično rešimo tudi sistem $Va = y$.

3.7.5 Razpršene matrike

Matrika je *razpršena*, če je večina njenih elementov enakih 0, ostali pa nimajo kakšne posebne strukture. Pri taki matriki shranimo le indekse in vrednosti neničelnih elementov.

Pri LU razcepu razpršene matrike oz. razcepu Choleskega za s.p.d. razpršeno matriko so lahko faktorji L, U oziroma V daleč od razpršenosti.

Pomaga lahko, če stolpce in vrstice predhodno tako preuredimo, da bo pri razcepu nastalo čim manj novih neničelnih elementov. Obstajajo različni algoritmi in pristopi, ki za različne tipe matrik dajejo različne rezultate.

Ponavadi se za razpršene matrike uporablja iterativne metode namesto direktnih.

IV. Reševanje predoločenih sistemov

4.1 Uvod

Če imamo več enačb kot neznank, rešujemo sistem $Ax = b$, kjer je A pravokotna matrika $m \times n$ in $m > n$. Tak sistem imenujemo *predoločen sistem*. V splošnem nima rešitve, lahko pa poiščemo x, pri katerem bo napaka $Ax - b$ najmanjša. Predpostavimo še, da je $\text{rang}(A) = n$.

Če želimo minimizirati $\|Ax - b\|_2$, potem govorimo o *rešitvi po metodi najmanjših kvadratov*.

Kje srečamo takšne probleme:

- Pri statistiki ocenjujemo parametre modela na podlagi opazovanj. Predpostavimo, da je uspeh b študenta v prvem letniku odvisen od
  - a_1: uspeha v srednji šoli,
  - a_2: uspeha na maturi,
  - a_3: uspeha na sprejemnem izpitu.

  Določiti moramo parametre x_1, x_2, x_3 v linearnem modelu $b = x_1a_1 + x_2a_2 + x_3a_3$. Če vzamemo podatke za m študentov, dobimo predoločeni sistem

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ \vdots & & \vdots \\ a_{n1} & a_{n2} & a_{n3} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{bmatrix}.$$

- Iščemo krivuljo oblike $y = ae^{bx}$, ki se najbolje prilega točkam (x_i, y_i), $i = 1, \ldots, m$. Ker model ni linearen, ga linearizirамo:

$$\ln y = \ln a + bx.$$

  Tako dobimo predoločeni sistem

$$\begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \vdots \\ 1 & x_m \end{bmatrix} \begin{bmatrix} \ln a \\ b \end{bmatrix} = \begin{bmatrix} \ln y_1 \\ \ln y_2 \\ \vdots \\ \ln y_m \end{bmatrix}.$$

4.2 Normalni sistem

Če sistem $Ax = b$ z leve pomnožimo z A^T, dobimo *normalni sistem*

$$A^TAx = A^Tb,$$

ki je nesingularen, saj je A polnega ranga.

Lema 12 *Rešitev normalnega sistema je rešitev po metodi najmanjših kvadratov.*

Dokaz. Naj bo $B = A^TA$ in $c = A^Tb$. Matrika B je s.p.d. Velja

$$\|Ax - b\|_2^2 = (Bx - c)^TB^{-1}(Bx - c) - c^TB^{-1}c + b^Tb,$$

to pa bo zaradi tega, ker je B^{-1} s.p.d., minimalno, ko bo $Bx = c$. ■

Matrika A^TA je s.p.d., zato za reševanje normalnega sistema uporabimo razcep Choleskega. Število operacij za izračun A^TA, razcep Choleskega in reševanje sistema je $n^2m+\frac{1}{3}n^3+\mathcal{O}(n^2)$, ker pa je ponavadi $m \gg n$, je najpomembnejši člen n^2m.

Normalni sistem je najpreprostejši način reševanja predoločenega sistema, ni pa najstabilnejši.

Zgled 21 *Denimo, da iščemo polinom $p(x) = a_0 + a_1x + \cdots + a_nx^n$ stopnje n, ki se najbolje prilega točkam (x_i, y_i), $i = 1, \ldots, m$. Matrika $B = A^TA$ ima elemente $b_{ij} = \sum_{k=1}^m x_k^{i+j-2}$. Če so točke x_i enakomerno porazdeljene po intervalu $(0, 1)$, torej $x_i = i/(m + 1)$, velja*

$$b_{ij} = \sum_{k=1}^{m} \left(\frac{k}{m+1}\right)^{i+j-2} \approx (m+1)\int_0^1 x^{i+j-2}dx = \frac{m+1}{i+j-1},$$

to pa pomeni, da je $B \approx (m + 1)H_{n+1}$. Ker pa so Hilbertove matrike zgled za zelo občutljive matrike, računanje aproksimacijskega polinoma visoke stopnje preko normalnega sistema ni stabilno. ■

4.3 QR razcep

Denimo, da poznamo razcep $A = QR$, kjer je Q pravokotna matrika $m \times n$ z ortonormiranimi stolpci, R zgornja trikotna matrika $n \times n$. Tak razcep imenujemo *QR razcep*. Potem iz normalnega sistema dobimo

$$x = (A^TA)^{-1}A^Tb = (R^TQ^TQR)^{-1}R^TQ^Tb = R^{-1}R^{-T}R^TQ^Tb = R^{-1}Q^Tb.$$

Rešitev po metodi najmanjših kvadratov torej dobimo, če rešimo zgornje trikotni sistem

$$Rx = Q^Tb.$$

Reševanje preko QR razcepa je stabilnejše od normalnega sistema.

Izrek 13 *Naj bo $A \in \mathbb{R}^{m\times n}$, $m \geq n$ in $\operatorname{rang}(A) = n$. Potem obstaja enolični QR razcep $A = QR$, kjer je Q pravokotna matrika $m \times n$ z ortonormiranimi stolpci, R pa zgornja trikotna matrika $n \times n$ s pozitivnimi diagonalnimi elementi.*

Dokaz. Za dokaz obstoja bomo QR razcep kar skonstruirali. Denimo, da je $A = [a_1 \ \cdots \ a_n]$ in $Q = [q_1 \ \cdots \ q_n]$. Potem iz $A = QR$ sledi

$$a_k = \sum_{i=1}^{k} r_{ik}q_i.$$

Vektorji q_i so ortonormirani in razpenjajo isti podprostor kot a_i. To pomeni, da lahko Q in R dobimo z Gram-Schmidtovo ortogonalizacijo stolpcev matrike A.

Algoritem je:

$k = 1, \ldots, n$
  $q_k = a_k$
  $i = 1, \ldots, k-1$
    $r_{ik} = q_i^T a_k$ (CGS) ali $r_{ik} = q_i^T q_k$ (MGS)
    $q_k = q_k - r_{ik} q_i$
  $r_{kk} = \|q_k\|_2$
  $q_k = \frac{q_k}{r_{kk}}$

CGS je *klasična Gram-Schmidtova metoda*, MGS pa *modificirana Gram-Schmidtova metoda*. Pri eksaktnem računanju vrneta CGS in MGS identične rezultate, numerično pa je MGS stabilnejši od CGS.

Enoličnost: denimo, da je $A = QR$. Potem velja

$$A^T A = R^T R,$$

kar je razcep Choleskega za s.p.d. matriko A. Ker je razcep Choleskega enoličen, je R enolična, prav tako pa potem tudi $Q = AR^{-1}$. ■

Zgled 22 *Če vzamemo $\epsilon = 10^{-10}$ in preko CGS in MGS v Matlabu ortogonaliziramo vektorje*

$$x_1 = \begin{bmatrix} 1+\epsilon \\ 1 \\ 1 \end{bmatrix}, \quad x_2 = \begin{bmatrix} 1 \\ 1+\epsilon \\ 1 \end{bmatrix}, \quad x_3 = \begin{bmatrix} 1 \\ 1 \\ 1+\epsilon \end{bmatrix},$$

dobimo pri CGS $q_2^T q_3 \approx 0.5$, kar je narobe, pri MGS pa $q_2^T q_3 = -1.1 \cdot 10^{-16}$. ■

Število operacij za QR razcep je

$$\sum_{k=1}^{n} \left(3m + \sum_{i=1}^{k-1} 4m \right) \approx 4m \sum_{k=1}^{n} k \approx 2mn^2,$$

kar je približno dvakrat toliko operacij kot pri normalnem sistemu (za $m \gg n$).

Poleg QR razcepa poznamo še *razširjeni QR razcep* $A = \widetilde{Q}\widetilde{R}$, kjer je $\widetilde{Q}$ ortogonalna matrika $m \times m$, $\widetilde{R}$ pa zgornja trapezna matrika $m \times n$. Prvih n stolpcev matrike $\widetilde{Q}$ in zgornji kvadrat matrike $\widetilde{R}$ tvori QR razcep matrike A. Tudi razširjeni QR razcep bomo ponavadi imenovali kar QR razcep, saj je vse razvidno iz dimenzij matrik.

Naj bo $A = \widetilde{Q}\widetilde{R}$, kjer je $\widetilde{Q} = [Q \; Q_1]$ in $\widetilde{R} = \begin{bmatrix} R \\ 0 \end{bmatrix}$. Potem velja

$$\|Ax - b\|_2 = \|\widetilde{Q}^T(Ax - b)\|_2 = \| \begin{bmatrix} R \\ 0 \end{bmatrix} x - \begin{bmatrix} Q^T b \\ Q_1^T b \end{bmatrix} \|_2 = \| \begin{bmatrix} Rx - Q^T b \\ -Q_1^T b \end{bmatrix} \|_2.$$

Zgornji del lahko uničimo, spodnjega pa ne. Velja torej

$$\min_{x \in \mathbb{R}^n} \|Ax - b\|_2 = \|Q_1^T b\|_2,$$

minimum pa je dosežen pri $Rx = Q^T b$.

4.4 Givensove rotacije

V ravnini vektor $x = [x_1\ x_2]^T$ zarotiramo za kot φ v neg. smeri tako, da ga pomnožimo z matriko

$$R^T = \begin{bmatrix} c & s \\ -s & c \end{bmatrix},$$

kjer sta $c = \cos\varphi$ in $s = \sin\varphi$. Če to posplošimo na rotacijo v ravnini (i,k) v $\mathbb{R}^n$, dobimo matriko R_{ik}^T, ki je enaka identiteti razen v i-ti in k-ti vrstici, kjer je

$$R_{ik}^T([i,k],[i,k]) = \begin{bmatrix} c & s \\ -s & c \end{bmatrix}.$$

Za vektor $y = R_{ik}^T x$ velja

$$\begin{aligned} y_j &= x_j, \quad j \neq i,k \\ y_i &= cx_i + sx_k \\ y_k &= -sx_i + cx_k \end{aligned}$$

in c, s lahko izberemo tako, da bo $y_k = 0$. Rešitev je

$$\begin{aligned} r &= \sqrt{x_i^2 + x_k^2} \\ c &= \frac{x_i}{r} \\ s &= \frac{x_k}{r}. \end{aligned}$$

Matriko R_{ik} imenujemo *Givensova rotacija*. Pri množenju matrike z R_{ik}^T se spremenita le i-ta in k-ta vrstica. Z ustreznimi rotacijami, ki jih uporabljamo v pravilnem vrstnem redu, lahko v matriki uničimo vse elemente pod diagonalo in tako izračunamo QR razcep.

Pri matriki 4×3 tako dobimo

$$A = \begin{bmatrix} \times & \times & \times \\ \times & \times & \times \\ \times & \times & \times \\ \times & \times & \times \end{bmatrix} \xrightarrow{R_{12}^T\cdot} \begin{bmatrix} \times & \times & \times \\ 0 & \times & \times \\ \times & \times & \times \\ \times & \times & \times \end{bmatrix} \xrightarrow{R_{13}^T\cdot} \begin{bmatrix} \times & \times & \times \\ 0 & \times & \times \\ 0 & \times & \times \\ \times & \times & \times \end{bmatrix} \xrightarrow{R_{14}^T\cdot} \begin{bmatrix} \times & \times & \times \\ 0 & \times & \times \\ 0 & \times & \times \\ 0 & \times & \times \end{bmatrix}$$

$$\xrightarrow{R_{23}^T\cdot} \begin{bmatrix} \times & \times & \times \\ 0 & \times & \times \\ 0 & 0 & \times \\ 0 & \times & \times \end{bmatrix} \xrightarrow{R_{24}^T\cdot} \begin{bmatrix} \times & \times & \times \\ 0 & \times & \times \\ 0 & 0 & \times \\ 0 & 0 & \times \end{bmatrix} \xrightarrow{R_{34}^T\cdot} \begin{bmatrix} \times & \times & \times \\ 0 & \times & \times \\ 0 & 0 & \times \\ 0 & 0 & 0 \end{bmatrix} = \widetilde{R}.$$

Matrika $\widetilde{R}$ je zgornja trapezna, v zgornjem kvadratu pa vsebuje matriko R. Produkt $\widetilde{Q} = R_{12}R_{13}R_{14}R_{23}R_{24}R_{34}$ je ortogonalna matrika, ki v prvih n stolpcih vsebuje matriko Q. Tako smo dobili razcep $A = QR$.

Pri množenju z R_{jk}^T se j-ta in k-ta vrstica spremenita v linearni kombinaciji j-te in k-te vrstice. Če je v nekem stolpcu v obeh vrsticah 0, se to ne more pokvariti z množenjem z R_{jk}^T.

Skica algoritma:

$Q = I_m$
$j = 1, \ldots, n$
  $k = j+1, \ldots, m$
    $r = (a_{jj}^2 + a_{jk}^2)^{1/2}$
    $c = a_{jj}/r$
    $s = a_{jk}/r$
    $A([j\ k], j:n) = \begin{bmatrix} c & s \\ -s & c \end{bmatrix} A([j\ k], j:n)$
    $b([j\ k]) = \begin{bmatrix} c & s \\ -s & c \end{bmatrix} b([j\ k])$ (če rešujemo predoločeni sistem $Ax = b$)
    $Q([j\ k], 1:m) = \begin{bmatrix} c & s \\ -s & c \end{bmatrix} Q([j\ k], 1:m)$ (če potrebujemo Q)
$Q = Q^T$.

Število operacij za reševanje $Ax = b$ je

$$\sum_{j=1}^{n} \left(\sum_{k=j+1}^{m} (6 + 6(n-j+1) + 6) \right) \approx 6 \sum_{j=1}^{n} (m-j)(n-j) \approx 3mn^2 - n^3.$$

Če potrebujemo Q, imamo še dodatnih $6m^2n - 3mn^2$ operacij. Matriko $n \times n$ tako po Givensu transformiramo v zgornjo trikotno z uporabo $2n^3$ operacij, za Q pa potrebujemo še dodatnih $3n^3$.

4.5 Householderjeva zrcaljenja

Za vektor $w \in \mathbb{R}^n$, kjer je $w \neq 0$, definiramo

$$P = I - \frac{2}{w^T w} w w^T.$$

P je simetrična in ortogonalna matrika, saj je $P = P^T$ in $P^2 = I$. Vsak vektor $x \in \mathbb{R}^n$ lahko zapišemo kot $x = \alpha w + u$, kjer je $u \perp w$. Dobimo $Px = -\alpha w + u$, kar pomeni, da je P zrcaljenje preko hiperravnine, ki je ortogonalna na w. Matriko P imenujemo *Householderjevo zrcaljenje*.

Množenje s P izvedemo tako, da izračunamo $Px = x - \frac{1}{m}(x^T w)w$, kjer je $m = \frac{1}{2} w^T w$.

Če imamo taka neničelna vektorja x in y, da je $\|x\|_2 = \|y\|_2$, potem velja $Px = y$, če izberemo $w = y - x$.

Naj bo $x \neq 0$. Iščemo zrcaljenje, ki v x uniči vse komponente razen prve, torej $Px = \pm k e_1$, kjer je $k = \|x\|_2$. Veljati mora $w = x \mp k e_1$, vprašanje je le, kateri predznak izbrati. Za m dobimo

$$m = \frac{1}{2} w^T w = \frac{1}{2}(k^2 \mp 2kx_1 + k^2) = k(k \mp x_1).$$

Zaradi deljenja z m želimo, da bo m čim večji, zato izberemo $m = k(k + |x_1|)$. Izbira je torej

$$w = \begin{bmatrix} x_1 + \operatorname{sign}(x_1)\|x\|_2 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}.$$

Število operacij za izračun produkta $Pz = z - \frac{1}{m}(z^T w)w$ je $4n + \mathcal{O}(1)$, pri čemer m izračunamo vnaprej. Za izračun w in m pa potrebujemo $2n + \mathcal{O}(1)$ operacij.

Z množenjem matrike z ustreznimi zrcaljenji jo lahko spremenimo v zgornjo trapezno obliko. Pri matriki 4×3 tako dobimo

$$A = \begin{bmatrix} \times & \times & \times \\ \times & \times & \times \\ \times & \times & \times \\ \times & \times & \times \end{bmatrix} \xrightarrow{\widetilde{P}_1 \cdot} \begin{bmatrix} \times & \times & \times \\ 0 & \times & \times \\ 0 & \times & \times \\ 0 & \times & \times \end{bmatrix} \xrightarrow{\widetilde{P}_2 \cdot} \begin{bmatrix} \times & \times & \times \\ 0 & \times & \times \\ 0 & 0 & \times \\ 0 & 0 & \times \end{bmatrix} \xrightarrow{\widetilde{P}_3 \cdot} \begin{bmatrix} \times & \times & \times \\ 0 & \times & \times \\ 0 & 0 & \times \\ 0 & 0 & 0 \end{bmatrix}.$$

Pri tem je

$$\widetilde{P}_i = \begin{array}{c} \\ i \\ m-i \end{array} \begin{array}{c} \begin{array}{cc} i & m-i \end{array} \\ \begin{pmatrix} I_i & 0 \\ 0 & P_i \end{pmatrix} \end{array}.$$

$Q^T = \widetilde{P}_3 \widetilde{P}_2 \widetilde{P}_1$, torej $Q = \widetilde{P}_1 \widetilde{P}_2 \widetilde{P}_3$. Namesto računanja Q raje shranimo vektorje w_i.

Skica algoritma:

$Q = I_m$
$i = 1, \ldots, n$
  določi $w_i \in \mathbb{R}^{m-i+1}$, ki prezrcali $A(i:m, i)$ v $\pm k e_1$.
  $A(i:m, i:n) = P_i \cdot A(i:m, i:n)$
  $b(i:m) = P_i \cdot b(i:m)$ (če rešujemo predoločeni sistem $Ax = b$)
  $Q(i:m, 1:n) = P_i \cdot Q(i:m, 1:n)$ (če potrebujemo Q)
$Q = Q^T$.

Število operacij za reševanje $Ax = b$ je

$$\sum_{i=1}^{n} [2(m-i+1) + 4(n-i+1)(m-i+1) + 4(m-i+1)] \approx 4 \sum_{i=1}^{n} (m-i)(n-i) \approx$$

$$\approx 2mn^2 - \frac{2}{3}n^3.$$

Za Q potrebujemo še dodatnih $4m^2n - 2mn^2$ operacij. Matriko $n \times n$ s Householderjevimi zrcaljenji transformiramo v zgornjo trikotno z uporabo $\frac{4}{3}n^3$ operacij, za Q pa potrebujemo še dodatnih $2n^3$.

Primerjava obstoječih metod:

- Reševanje predoločenega sistema $Ax = b$, $m \gg n$:
  - normalni sistem: mn^2,

  - MGS: $2mn^2$,
  - Givens: $3mn^2 - n^3$,
  - Householder: $2mn^2 - \frac{2}{3}n^3$.

- Reševanje kvadratnega sistema $Ax = b$, $m = n$:
  - LU razcep: $\frac{2}{3}n^3$,
  - Householder: $\frac{4}{3}n^3$,
  - Givens: $2n^3$.

4.6 Singularni razcep

Izrek 14 *Za $A \in \mathbb{R}^{m\times n}$, $m \geq n$, obstaja* singularni razcep

$$A = U\Sigma V^T,$$

kjer sta $U \in \mathbb{R}^{m\times m}$ in $V \in \mathbb{R}^{n\times n}$ ortogonalni matriki in $\Sigma \in \mathbb{R}^{m\times n}$ oblike

$$\Sigma = \begin{bmatrix} \sigma_1 & & \\ & \ddots & \\ & & \sigma_n \\ & & \end{bmatrix},$$

kjer so $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_n \geq 0$ singularne vrednosti A.

Stolpci $U = [u_1 \ \cdots \ u_m]$ so *levi*, stolpci $V = [v_1 \ \cdots \ v_n]$ pa *desni singularni vektorji*.

Dokaz. Ker je A^TA simetrična pozitivno semidefinitna matrika, so vse njene lastne vrednosti nenegativne:

$$\sigma_1^2 \geq \sigma_2^2 \geq \cdots \geq \sigma_n^2 \geq 0.$$

Ustrezni ortonormirani lastni vektorji naj bodo $A^TAv_i = \sigma_i^2 v_i$, $i = 1, \ldots, n$.

Naj bo $\sigma_r > 0$ in $\sigma_{r+1} = \cdots = \sigma_n = 0$. Označimo $V_1 := [v_1 \ \cdots \ v_r]$ in $V_2 := [v_{r+1} \ \cdots \ v_n]$. Iz

$$(AV_2)^T(AV_2) = V_2^TA^TAV_2 = V_2^T[0 \ \cdots \ 0] = 0$$

sledi $AV_2 = 0$.

Sedaj definiramo $u_i = \frac{1}{\sigma_i}Av_i$, $i = 1, \ldots, r$. Vektorji $u_1, \ldots, u_r$ so ortonormirani, saj je

$$u_i^Tu_j = \frac{1}{\sigma_i\sigma_j}v_i^TA^TAv_j = \frac{\sigma_j}{\sigma_i}v_i^Tv_j = \delta_{ij}, \quad i, j = 1, \ldots, r.$$

Označimo $U_1 := [u_1 \ \cdots \ u_r]$ in izberemo $U_2 := [u_{r+1} \ \cdots \ u_n]$ tako, da je $U = [U_1 \ U_2]$ ortogonalna matrika. Matrika U^TAV ima obliko

$$U^TAV = \begin{matrix} \\ r \\ m-r \end{matrix} \begin{matrix} \begin{matrix} r & n-r \end{matrix} \\ \begin{pmatrix} U_1^TAV_1 & U_1^TAV_2 \\ U_2^TAV_1 & U_2^TAV_2 \end{pmatrix} \end{matrix}.$$

Desna dva bloka sta zaradi $AV_2 = 0$ enaka 0. Za $i = 1, \ldots, r$ in $k = 1, \ldots, m$ velja

$$u_k A v_i = \sigma_i u_k u_i = \sigma_i \delta_{ik},$$

torej $U_2^T A V_1 = 0$ in $U_1^T A V_1 = \mathrm{diag}(\sigma_1, \ldots, \sigma_r)$. Tako smo dobili SV razcep $A = U \Sigma V^T$, kjer je $S = \mathrm{diag}(\sigma_1, \ldots, \sigma_r)$ in

$$\Sigma = \begin{array}{c} \\ r \\ m-r \end{array} \begin{array}{c} \begin{array}{cc} r & n-r \end{array} \\ \begin{pmatrix} S & 0 \\ 0 & 0 \end{pmatrix} \end{array}. \quad \blacksquare$$

V primeru $n < m$ dobimo SV razcep tako, da transponiramo SV razcep za A^T.

Pomen SV razcepa:

- r se ujema z $\mathrm{rang}(A)$,
- stolpci U_1 tvorijo bazo za $\mathrm{im} A$,
- stolpci V_2 tvorijo bazo za $\ker A$,
- stolpci U_2 tvorijo bazo za $\ker A^T$,
- stolpci V_1 tvorijo bazo za $\mathrm{im} A^T$.

Poleg omenjenega poznamo tudi SV razcep oblike $A = \widetilde{U} \widetilde{\Sigma} V^T$, kjer je $\widetilde{U}$ matrika $m \times n$ z ortonormiranimi stolpci, $\widetilde{\Sigma} = \mathrm{diag}(\sigma_1, \ldots, \sigma_r)$, V pa je $n \times n$ ortogonalna matrika. $\widetilde{U}$ se ujema s prvimi n stolpci matrike U, $\widetilde{\Sigma}$ pa je zgornji kvadrat Σ v SV razcepu $A = U \Sigma V^T$.

A predstavlja preslikavo iz $\mathbb{R}^n$ v $\mathbb{R}^m$. Geometrijski pomen SV razcepa je, da se z ortogonalnima transformacijama baz U v $\mathbb{R}^m$ in V v $\mathbb{R}^n$ A spremeni v diagonalno matriko, saj potem velja

$$A v_i = \sigma u_i, \quad i = 1, \ldots, n.$$

Lema 13 *Če je $A \in \mathbb{R}^{m \times n}$, $m \geq n$, $\mathrm{rang}(A) = n$, potem je minimum $\|Ax - b\|_2$ dosežen pri*

$$x = \sum_{i=1}^{n} \frac{u_i^T b}{\sigma_i} v_i.$$

Dokaz. Naj bo $A = U \Sigma V^T$ in

$$U = \begin{array}{c} \begin{array}{cc} n & m-n \end{array} \\ \begin{pmatrix} U_1 & U_2 \end{pmatrix} \end{array}, \quad \Sigma = \begin{array}{c} \\ n \\ m-n \end{array} \begin{array}{c} \\ \begin{pmatrix} S \\ 0 \end{pmatrix} \end{array}.$$

$$\|Ax - b\|_2 = \|U \Sigma V^T x - b\|_2 = \|\Sigma V^T x - U^T b\|_2 = \left\| \begin{bmatrix} S V^T x - U_1^T b \\ U_2^T b \end{bmatrix} \right\|_2.$$

Minimum je dosežen pri $S V^T x = U_1^T b$ oziroma

$$x = V S^{-1} U_1^T b = \sum_{i=1}^{n} \frac{u_i^T b}{\sigma_i} v_i. \quad \blacksquare$$

Definicija 4 *Za matriko* $A \in \mathbb{R}^{m\times n}$, $m \geq n$, $\operatorname{rang}(A) = n$, *definiramo* (Moore-Penroseov) psevdoinverz $A^+ \in \mathbb{R}^{n\times m}$ *kot*

$$A^+ = (A^TA)^{-1}A^T.$$

V primeru $m < n$ *in* $\operatorname{rang}(A) = m$ *definiramo* $A^+ = A^T(AA^T)^{-1}$.

Rešitev predoločenega sistema polnega ranga $Ax = b$ lahko zapišemo kot $x = A^+b$.

Če A ni polnega ranga, je psevdoinverz definiran preko SV razcepa. Naj bo $A \in \mathbb{R}^{m\times n}$, $m \geq n$, $\operatorname{rang}(A) = r$ in $A = U\Sigma V^T$, kjer je

$$U = \begin{matrix} \\ \end{matrix}\begin{matrix} r & m-r \end{matrix}\\ \begin{pmatrix} U_1 & U_2 \end{pmatrix}, \quad V = \begin{matrix} \\ \end{matrix}\begin{matrix} r & n-r \end{matrix}\\ \begin{pmatrix} V_1 & V_2 \end{pmatrix}, \quad \Sigma = \begin{matrix} r \\ m-r \end{matrix}\begin{matrix} \\ \end{matrix}\begin{matrix} r & n-r \end{matrix}\\ \begin{pmatrix} S & 0 \\ 0 & 0 \end{pmatrix}$$

in $S = \operatorname{diag}(\sigma_1, \ldots, \sigma_r)$. Potem je

$$A^+ = V\Sigma^+U^T,$$

kjer je

$$\Sigma^+ = \begin{matrix} \\ r \\ n-r \end{matrix}\begin{matrix} \begin{matrix} r & m-r \end{matrix} \\ \end{matrix}\\ \begin{pmatrix} S^{-1} & 0 \\ 0 & 0 \end{pmatrix}.$$

Lema 14 *Matrika* X *je psevdoinverz* A *natanko tedaj, ko izpolnjuje Moore-Penroseove pogoje:*

1) $AXA = A$,

2) $XAX = X$,

3) $(AX)^T = AX$,

4) $(XA)^T = XA$. ■

Če je $A = U\Sigma V^T$ SV razcep A in je $\operatorname{rang}(A) = r$, potem direktno iz razcepa sledi

$$A = \sum_{i=1}^{r} \sigma_i u_i v_i^T.$$

Izrek 15 *Naj bo* $A = U\Sigma V^T$ *SV razcep* A *in* $\operatorname{rang}(A) > k$. *Naj bo*

$$A_k = \sum_{i=1}^{k} \sigma_i u_i v_i^T$$

oziroma $A_k = U\Sigma_k V^T$, *kjer je*

$$\Sigma_k = \begin{bmatrix} \sigma_1 & & & & & \\ & \ddots & & & & \\ & & \sigma_k & & & \\ & & & 0 & & \\ & & & & \ddots & \\ & & & & & 0 \\ & & & & & \end{bmatrix}.$$

Potem velja

$$\min_{\text{rang}(B)=k} \|B - A\|_2 = \|A_k - A\|_2 = \sigma_{k+1}.$$

Dokaz. Če je $\text{rang}(B) = k$, potem je $\dim \ker B = n - k$. Naj bo $V_{k+1} = [v_1 \ \cdots \ v_{k+1}]$. Ker je $\dim \text{im} V_{k+1} + \dim \ker B = n + 1$, obstaja $0 \neq z \in \text{im} V_{k+1} \cap \ker B$ in $\|z\|_2 = 1$. Dobimo

$$\|A - B\|_2^2 \geq \|(A - B)z\|_2^2 = \|Az\|_2^2 = \|U\Sigma V^T z\|_2^2 = \|\Sigma V^T z\|_2^2 \geq \sigma_{k+1}^2 \|V^T z\| = \sigma_{k+1}^2.$$

Po drugi strani je očitno, da je $\|A_{k+1} - A\|_2 = \sigma_{k+1}$, saj je maksimum dosežen pri v_{k+1}. ■

To pomeni, da je A_k najboljša aproksimacija matrike A z matriko ranga k, σ_{k+1} pa nam pove, kako daleč je A od prostora matrik ranga k.

Zgled 23 *SV razcep lahko uporabimo za kompresijo slik. Sliko lahko predstavimo z matriko A, katere elementi predstavljajo nivo sivine. Namesto A vzamemo najboljšo aproksimacijo z matriko ranga k. Pri tem namesto mn podatkov potrebujemo le $(m + n)k$ podatkov za $[u_1 \ \cdots \ u_k]$ in $[\sigma_1 v_1 \ \cdots \ \sigma_k v_k]$.*

4.7 Teorija motenj

Za matriko A, ki je ranga r, definiramo

$$\kappa_2(A) = \|A\|_2 \|A^+\|_2 = \frac{\sigma_1(A)}{\sigma_r(A)}.$$

Izrek 16 *Naj bo $A \in \mathbb{R}^{m \times n}$, $m \geq n$, $\text{rang}(A) = n$, $x = A^+ b$ rešitev predoločenega sistema in $r = Ax - b$. Naj bo $\widetilde{x} = (A + \delta A)^+ (b + \delta b)$, kjer je*

$$\epsilon = \max\left(\frac{\|\delta A\|_2}{\|A\|_2}, \frac{\|\delta b\|_2}{\|b\|_2}\right) < \frac{1}{\kappa_2(A)}.$$

Potem je $(A + \delta A)$ ranga k in velja

$$\frac{\|\widetilde{x} - x\|_2}{\|x\|_2} \leq \frac{\epsilon \kappa_2(A)}{1 - \epsilon \kappa_2(A)} \left(2 + (\kappa_2(A) + 1)\frac{\|r\|_2}{\|A\|_2 \|x\|_2}\right). \quad ■$$

V primeru, ko je $\|r\|_2$ majhna, je občutljivost reda $\mathcal{O}(\kappa_2(A))$, če pa $\|r\|_2$ ni zanemarljiva, je občutljivost predoločenega sistema reda $\mathcal{O}(\kappa_2^2(A))$, V primeru $r = 0$ se ocena ujema z oceno občutljivosti linearnega sistema.

4.8 Podobni problemi

Če imamo sistem $Ax = b$, kjer je $A \in \mathbb{R}^{m\times n}$, $m < n$ in $\text{rang}(A) = m$, potem je to *poddoločen sistem*. Ker je $\dim \ker A = n - m$, rešitev ni enolična, saj ji lahko prištejemo poljuben $z \in \ker A$. Zaradi tega iščemo tisto rešitev x, ki ima minimalno normo $\|x\|_2$. Iskana rešitev je $x = A^+b$ oziroma

$$x = \sum_{i=1}^{m} \frac{u_i^T b}{\sigma_i} v_i.$$

Za splošni sistem $Ax = b$ lahko rečemo, da v primeru, ko matrika A ni polnega ranga, izmed vseh rešitev x, ki minimizirajo $\|Ax - b\|_2$ vzamemo tisto z minimalno normo $\|x\|_2$. Če je $\text{rang}(A) = r$, potem je iskana rešitev kar

$$x = A^+b = \sum_{i=1}^{r} \frac{u_i^T b}{\sigma_i} v_i.$$

Pri numeričnem računanju je težko ugotoviti točen rang matrike, zato ponavadi tiste singularne vrednosti, ki so blizu 0, proglasimo za 0 in nato preko SV razcepa poiščemo rešitev.

V. Nesimetrični lastni problem

5.1 Uvod

Za $A \in \mathbb{R}^{n\times n}$ rešujemo $Ax = \lambda x$, kjer $x \in \mathbb{C}^n$, $x \neq 0$ in $\lambda \in \mathbb{C}$. λ je *lastna vrednost*, ustrezni vektor x pa *(desni) lastni vektor*. Vektor $y \neq 0$, pri katerem je $y^*A = \lambda y^*$, je *levi lastni vektor*.

Matrika A se da diagonalizirati, če obstajata nesingularna matrika $X = [x_1 \ \cdots \ x_n]$ in diagonalna matrika $\Lambda = \mathrm{diag}(\lambda_1, \ldots, \lambda_n)$, da je $A = X\Lambda X^{-1}$. V tem primeru je $Ax_i = \lambda_i x_i$ za $i = 1, \ldots, n$.

Lastne vrednosti so ničle karakterističnega polinoma $p(\lambda) = \det(A - \lambda I)$, ker pa so ničle polinoma lahko zelo občutljive na motnje koeficientov (Wilkinsonov primer), to ni najprimernejša pot za računanje lastnih vrednosti.

Če je S nesingularna matrika, imata A in $S^{-1}AS$ enake lastne vrednosti, saj sta podobni matriki.

5.2 Jordanova in Schurova forma

Za vsako matriko A obstaja taka nesingularna matrika X, da je $X^{-1}AX = J$, kjer je $J = \mathrm{diag}(J_1, \ldots, J_k)$ in je

$$J_i = \begin{bmatrix} \lambda_i & 1 & & \\ & \ddots & \ddots & \\ & & \ddots & 1 \\ & & & \lambda_i \end{bmatrix}$$

Jordanova kletka. Matrika J je *Jordanova forma*.

Jordanova forma pove veliko o matriki A, na žalost pa ni stabilna. Ker ni zvezna funkcija elementov matrike A, jo lahko majhne motnje popolnoma spremenijo.

Zgled 24

$$A = \begin{bmatrix} 0 & 1 & & \\ & \ddots & \ddots & \\ & & \ddots & 1 \\ & & & 0 \end{bmatrix}$$

je kar Jordanova forma z eno samo kletko $n \times n$,

$$A(\epsilon) = \begin{bmatrix} 0 & 1 & & \\ & \ddots & \ddots & \\ & & \ddots & 1 \\ \epsilon & & & 0 \end{bmatrix}$$

pa ima n *kletk* 1×1 *z lastnimi vrednostmi* $\sqrt[n]{\epsilon}$. ■

Namesto Jordanove forme uporabljamo stabilno *Schurovo formo*, kjer je podobnostna transformacija ortogonalna.

Izrek 17 *Za vsako matriko A obstajata unitarna matrika Q in zgornja trikotna matrika T, da je $Q^*AQ = T$ (Schurova forma).*

Dokaz. Naredimo indukcijo po n. Za $n = 1$ izrek očitno velja.

$n-1 \to n$: Naj bo λ lastna vrednost A in x normiran lastni vektor. Obstaja taka unitarna matrika U, da je $Ue_1 = x$ (Givens, Householder, ...). Matrika $B = U^*AU$ ima obliko

$$B = \begin{array}{c} \\ 1 \\ n-1 \end{array} \begin{array}{c} \begin{array}{cc} 1 & n-1 \end{array} \\ \begin{pmatrix} \lambda & \times \cdots \times \\ 0 & C \end{pmatrix} \end{array},$$

saj je $Be_1 = U^*AUe_1 = U^*Ax = U^*\lambda x = \lambda e_1$. Po indukcijski predpostavki obstaja Schurova forma za C. Torej obstaja unitarna V_1, da je $V_1^*CV_1 = T_1$ zgornja trikotna. Sedaj je

$$\underbrace{\begin{bmatrix} 1 & 0 \\ 0 & V_1^* \end{bmatrix}}_{V^*} B \underbrace{\begin{bmatrix} 1 & 0 \\ 0 & V_1 \end{bmatrix}}_{V} = \begin{bmatrix} \lambda & \times \cdots \times \\ 0 & T_1 \end{bmatrix}$$

in $\underbrace{V^*U^*}_{Q^*} A \underbrace{UV}_{Q}$ je zgornja trikotna matrika, torej Schurova forma za A. ∎

Schurova forma ni enolična, saj je npr. vrstni red λ_i poljuben.

V primeru realne matrike obstaja *realna Schurova forma*. Obstaja taka ortogonalna matrika Q, da je $Q^TAQ = T$, kjer je T *kvazi zgornja trikotna matrika* (na diagonali ima lahko bloke 2×2). V diagonalnih blokih 2×2 so skriti konjugirani pari kompleksnih lastnih vrednosti.

5.3 Teorija motenj

Zanima nas, kaj se dogaja z lastnimi vrednostmi (in vektorji), ko zmotimo elemente matrike. Lastne vrednosti so zvezne funkcije elementov matrike (saj so ničle karakterističnega polinoma), a spremembe so lahko kljub temu zelo velike.

Zgled 25 *Za $n \times n$ matriko $A = \begin{bmatrix} 0 & 1 & & \\ & \ddots & \ddots & \\ & & 0 & 1 \\ \epsilon & & & 0 \end{bmatrix}$ je karakteristični polinom $p(\lambda) = \lambda^n - \epsilon$ in $\lambda = \sqrt[n]{\epsilon}$. Pri $n = 16$ in $\epsilon = 10^{-16}$ je $|\lambda| = 0.1$.* ∎

V primeru, ko so lastne vrednosti blizu večkratnosti, lahko pričakujemo velike spremembe.

Izrek 18 *Naj bo λ_i enostavna lastna vrednost A, x_i ustrezni desni, y_i levi lastni vektor in $\|x\|_2 = \|y\|_2 = 1$. Če je $\lambda_i + \delta\lambda_i$ ustrezna lastna vrednost $A + \delta A$ in $x_i + \delta x_i$ ustrezni lastni vektor, potem velja*

$$\lambda_i + \delta\lambda_i = \lambda_i + \frac{y_i^*\delta A x_i}{y_i^* x_i} + \mathcal{O}(\|\delta A\|^2).$$

in

$$x_i + \delta x_i = x_i + \sum_{\substack{j=1 \\ j\neq i}}^{n} \frac{y_j^* \delta A x_i}{(\lambda_i - \lambda_j) s_j} x_j + \mathcal{O}(\|\delta A\|^2).$$

Dokaz. Velja $Ax_i = \lambda_i x_i$ in $(A+\delta A)(x_i+\delta x_i) = (\lambda_i + \delta\lambda_i)(x_i + \delta x_i)$. Če zanemarimo kvadratne δ člene in pomnožimo enačbo z leve z y_i^*, dobimo

$$\delta\lambda_i = \frac{y_i^* \delta A x_i}{y_i^* x_i}. \quad \blacksquare$$

Definicija 5 *Če definiramo* $s_i := \frac{y_i^* x_i}{\|x_i\|_2 \|y_i\|_2}$, *je* s_i^{-1} *občutljivost enostavne lastne vrednosti* λ_i. *Če je* λ_i *večkratna lastna vrednost, je občutljivost* ∞.

Če je $A = A^T$, potem je $s_i = 1$, saj so levi lastni vektorji enaki desnim.

Izrek 19 (Bauer-Fike) *Če je* $A = X\Lambda X^{-1}$, *kjer je* $\Lambda = \mathrm{diag}(\lambda_1, \ldots, \lambda_n)$, *potem vse lastne vrednosti* $\lambda(\epsilon)$ *matrike* $A + \epsilon E$ *ležijo v uniji* n *krogov*

$$|z - \lambda_i| \leq \epsilon \|X\| \cdot \|X^{-1}\| \cdot \|E\| = \epsilon \kappa(X) \|E\|, \quad i = 1, \ldots, n.$$

Dokaz. Naj bo $\lambda(\epsilon)$ lastna vrednost $A + \epsilon E$. Predpostavimo lahko $\lambda(\epsilon) \neq \lambda_i$, $i = 1, \ldots, n$, saj sicer nimamo kaj dokazovati.

Matrika $A + \epsilon E - \lambda(\epsilon) I$ je singularna. Dobimo

$$X^{-1}(A + \epsilon E - \lambda(\epsilon) I) X = \Lambda - \lambda(\epsilon) I + \epsilon X^{-1} E X = (\Lambda - \lambda(\epsilon) I)\Big(I + \epsilon (\Lambda - \lambda(\epsilon) I)^{-1} X^{-1} E X\Big).$$

Ker je $\Lambda - \lambda(\epsilon) I$ nesingularna matrika, mora biti $I + \epsilon(\Lambda - \lambda(\epsilon) I)^{-1} X^{-1} E X$ singularna matrika. To pa pomeni, da je

$$1 \leq \|\epsilon (\Lambda - \lambda(\epsilon) I)^{-1} X^{-1} E X\| \leq \epsilon \|\Lambda - \lambda(\epsilon) I)^{-1}\| \|X^{-1}\| \|E\| \|X\|.$$

Iz

$$\|\Lambda - \lambda(\epsilon) I)^{-1}\| = \frac{1}{\min_{i=1,\ldots,n} |\lambda_i - \lambda(\epsilon)|}$$

sledi

$$\min_{i=1,\ldots,n} |\lambda_i - \lambda(\epsilon)| \leq \epsilon \kappa(X) \|E\|. \quad \blacksquare$$

Opomba. Če unija krogov razpade na povezane komponente, potem vsaka komponenta vsebuje toliko lastnih vrednosti, kot je v njej krogov (zaradi zveznosti $\lambda_i(\epsilon)$).

Posledica 2 *Če je* $A = A^T$ *in* $E = E^T$, *potem je*

$$\min_{i=1,\ldots,n} |\lambda(\epsilon) - \lambda_i| \leq \epsilon \|E\|.$$

Dokaz. Za simetrično matriko velja, da je X ortogonalna matrika, torej $\kappa(X) = 1$. ■

Lema 15 *Če je λ_i enostavna lastna vrednost, potem je $s_i \neq 0$.*

Dokaz. Naj bo $s_1 = 0$ (privzamemo lahko $i = 1$), $\|x_1\| = \|y_1\| = 1$ pa naj bosta ustrezni levi in desni lastni vektor. U naj bo taka unitarna matrika, da je $Ue_1 = x_1$. Potem je matrika $B = U^*AU$ oblike

$$B = \begin{array}{c} \\ 1 \\ n-1 \end{array} \begin{array}{c} \begin{array}{cc} 1 & n-1 \end{array} \\ \begin{pmatrix} \lambda & \times \cdots \times \\ 0 & C \end{pmatrix} \end{array},$$

saj je $Be_1 = U^*AUe_1 = U^*Ax = U^*\lambda x = \lambda e_1$. Iz enakosti $Ax_1 = \lambda_1 x_1$, $y_1^*A = \lambda_1 y_1^*$ in $y_1^*x_1 = 0$ dobimo

$$\begin{aligned} U^*AUe_1 &= U^*\lambda_1 Ue_1 = \lambda_1 e_1 \\ (y_1^*U)U^*AU &= \lambda_1(y_1^*U) \qquad (5.6) \\ y_1^*Pe_1 &= 0 \qquad (5.7) \end{aligned}$$

Iz (5.7) sledi, da je y_1U oblike $[0\ z_1^*]$, ko pa to vstavimo v (5.6), dobimo $[0\ z_1^*]B = \lambda_1[0\ z_1^*]$ oziroma $z_1^*B = \lambda_1 z_1^*$. Ker je λ_1 lastna vrednost C, ima A vsaj dvojno lastno vrednost λ_1. ■

Opomba. V primeru večkratne lastne vrednosti lahko vektorja x_i in y_i določimo tako, da bo $s_i = 0$, ni pa to nujno res za poljubne lastne vektorje večkratne lastne vrednosti.

Izrek 20 *Naj bo $A = X\Lambda X^{-1}$, kjer je $\Lambda = \operatorname{diag}(\lambda_1, \ldots, \lambda_n)$, $X = [x_1\ \cdots\ x_n]$ in naj bo $Y = [y_1\ \cdots\ y_n]$ matrika levih lastnih vektorjev. Potem velja*

$$X^{-1} = \begin{bmatrix} \frac{1}{s_1}y_1^* \\ \vdots \\ \frac{1}{s_n}y_n^* \end{bmatrix}.$$

Dokaz. $Y^*A = \Lambda Y^*$, po drugi strani pa $X^{-1}A = \Lambda X^{-1}$. Od tod sledi, da so stolpci X^{-T} levi lastni vektorji. Torej je

$$X^{-1} = \begin{bmatrix} c_1 y_1^* \\ \vdots \\ c_n y_n^* \end{bmatrix}$$

za neke konstante $c_1, \ldots, c_n$. Če želimo $XX^{-1} = I$, mora veljati $c_i = \frac{1}{s_i}$. ■

Lema 16 *Matrika ne more imeti natanko ene zelo občutljive lastne vrednosti.*

Dokaz. Naj bodo vse lastne vrednosti različne, sicer že imamo občutljiv par. Naj bo $AX = X\Lambda$, $X = [x_1\ \cdots\ x_n]$ in $\|x_i\| = 1$. Naj bodo y_i normirani levi lastni vektorji. Po izreku 20 je

$$X \begin{bmatrix} \frac{1}{s_1}y_1^* \\ \vdots \\ \frac{1}{s_n}y_n^* \end{bmatrix} = I$$

oziroma

$$\sum_{i=1}^{n} \frac{x_i y_i^*}{s_i} = I.$$

Denimo, da je $|s_1|$ po absolutni vrednosti najmanjša. Ocenimo lahko

$$\frac{1}{|s_1|} \leq 1 + \sum_{j=2}^{n} \frac{1}{|s_j|}.$$

Za $j \neq 1$ dobimo

$$\frac{1}{|s_j|} \geq \frac{\frac{1}{|s_1|} - 1}{n-1},$$

od tod pa je očitno, da $|s_1|$ ne more biti sama zelo blizu 0. ■

Opomba. Velika občutljivost lastne vrednosti pomeni, da je blizu večkratne lastne vrednosti, to pa seveda pomeni, da obstaja vsaj še ena taka lastna vrednost,

5.4 Potenčna metoda

Definicija 6 *Naj* maxel(x) *pomeni maksimalni element po absolutni vrednosti vektorja* x.

Tako je npr. maxel$([1 \quad -2 \quad 3 \quad -4]^T) = -4$.

Algoritem za potenčno metodo je:

izberi $\widetilde{z}_0 \neq 0$, $z_0 = \frac{1}{\text{maxel}(\widetilde{z}_0)}\widetilde{z}_0$
$k = 0, 1, \ldots :$
  $\widetilde{z}_{k+1} = Az_k$
  $z_{k+1} = \frac{1}{\text{maxel}(\widetilde{z}_{k+1})}\widetilde{z}_{k+1}$

Izrek 21 *Naj bodo lastne vrednosti matrike* A *take, da velja*

$$|\lambda_1| > |\lambda_2| \geq |\lambda_3| \geq \cdots \geq |\lambda_n|.$$

Potem, ko gre $k \to \infty$, maxel$(\widetilde{z}_k)$ *konvegira proti* λ_1, z_k *pa po smeri konvergira proti lastnemu vektorju za* λ_1.

Dokaz. Izrek dokažemo za primer, ko se da matrika diagonalizirati. Naj velja $A = X\Lambda X^{-1}$, kjer je $X = [x_1 \ \cdots \ x_n]$ in $\Lambda = \text{diag}(\lambda_1, \ldots, \lambda_n)$. Začetni vektor z_0 lahko razvijemo po lastnih vektorjih kot

$$z_0 = \sum_{i=1}^{n} \alpha_i x_i.$$

Potem pri pogoju $\alpha_1 \neq 0$ velja

$$z_k = \frac{A^k z_0}{\text{maxel}(A^k z_0)} = \frac{\alpha_1\lambda_1^k x_1 + \alpha_2\lambda_2^k x_2 + \cdots + \alpha_n\lambda_n^k x_n}{\text{maxel}(\alpha_1\lambda_1^k x_1 + \alpha_2\lambda_2^k x_2 + \cdots + \alpha_n\lambda_n^k x_n)} =$$

$$= \frac{\alpha_1 x_1 + \alpha_2 \left(\frac{\lambda_2}{\lambda_1}\right)^k x_2 + \cdots + \alpha_n \left(\frac{\lambda_n}{\lambda_1}\right)^k x_n}{\text{maxel}\left(\alpha_1 x_1 + \alpha_2 \left(\frac{\lambda_2}{\lambda_1}\right)^k x_2 + \cdots + \alpha_n \left(\frac{\lambda_n}{\lambda_1}\right)^k x_n\right)} \to \pm x_1. \quad \blacksquare$$

Pri dokazu smo predpostavili:

a) $\alpha_1 \neq 0$,

b) A ima enostavno dominantno lastno vrednost.

Numerično je predpostavka *a)* vedno izpolnjena, saj zaokrožitvene napake povzročijo, da je $\alpha_1 \neq 0$. Pri b) se da pokazati, da izrek velja tudi, ko je λ_1 večkratna lastna vrednost. Metoda se da ustrezno predelati tudi za primera:

- $|\lambda_1| = |\lambda_2| > |\lambda_3| \geq \cdots$ in $\lambda_1 = -\lambda_2$,
- $|\lambda_1| = |\lambda_2| > |\lambda_3| \geq \cdots$ in $\lambda_1 = \overline{\lambda_2}$.

Tako dobimo prvo dominantno lastno vrednost. Za ostale lahko naredimo redukcijo:

a) *Hotelingova redukcija* za $A = A^T$. Definiramo

$$B = A - \lambda_1 x_1 x_1^T.$$

Velja $Bx_1 = 0$ in $Bx_k = \lambda_k x_k$ za $k \neq i$.

b) *Househoderjeva redukcija* za splošno matriko. Poiščemo ortogonalno Q, da je $Qx_1 = e_1$, Potem ima $B = QAQ^T$ obliko

$$B = \begin{bmatrix} \lambda_1 & b^T \\ 0 & C \end{bmatrix},$$

saj je

$$Be_1 = QAQ^T e_1 = QAx_1 = Q\lambda_1 x_1 = \lambda_1 e_1.$$

Preostale lastne vrednosti dobimo v matriki C.

Hitrost konvergence je odvisna od razmerja $\left|\frac{\lambda_1}{\lambda_2}\right|$. Če je blizu 1, bo konvergenca počasna, blizu 0 pa hitrejša. Pomagamo si lahko s premiki. Matrika $A - \sigma I$ ima enake lastne vektorje kot A, lastne vrednosti pa so $\lambda_i - \sigma$, $i = 1, \ldots, n$. Sedaj je konvergenca lahko hitrejša.

Če iščemo lastno vrednost nesingularne matrike A, ki je najmanjša po absolutni vrednosti, delamo potenčno metodo za A^{-1}, saj ima A^{-1} lastne vrednosti $\lambda^{-1}, \ldots, \lambda_n^{-1}$. V algoritmu namesto množenja $\widetilde{z}_{k+1} = A^{-1} z_k$ rešujemo sistem $A\widetilde{z}_{k+1} = z_k$.

5.5 Inverzna iteracija

Naj bo σ tak približek, da je $|\lambda_i - \sigma| \ll |\lambda_j - \sigma|$ za $j \neq i$. Algoritem za inverzno iteracijo je:

izberi $\widetilde{z}_0 \neq 0$, $z_0 = \frac{1}{\|\widetilde{z}_0\|_\infty}\widetilde{z}_0$
$k = 0, 1, \ldots :$
  reši $(A - \sigma I)\widetilde{z}_{k+1} = z_k$
  $z_{k+1} = \frac{1}{\|\widetilde{z}_{k+1}\|_\infty}\widetilde{z}_{k+1}$

Vektor z_k konvergira po smeri proti lastnemu vektorju za λ_i. Inverzna iteracija ni nič drugega kot potenčna metoda za $(A - \sigma I)^{-1}$, zato jo imenujemo tudi *inverzna potenčna metoda*. Lastne vrednosti $(A - \sigma I)^{-1}$ so $(\lambda_j - \sigma)^{-1}$, $j = 1, \ldots, n$, in $(\lambda_i - \sigma)^{-1}$ je zelo dominantna lastna vrednost, zato ponavadi potrebujemo le dva koraka inverzne iteracije za poljubni začetni vektor.

Inverzno iteracijo ponavadi uporabljamo zato, da dobimo lastni vektor za numerično izračunano lastno vrednost. V tem primeru za σ vzamemo kar izračunano lastno vrednost, zaokrožitvene napake pa povzročijo, da nimamo težav s singularnostjo matrike $A - \sigma I$.

5.6 Ortogonalna iteracija

Namesto dominantnega vektorja, bi radi izračunali dominantni invariantni podprostor dimenzije p za matriko A.

Algoritem za ortogonalno iteracijo je:

izberi matriko Z_0 velikosti $n \times p$ z ortonormiranimi stolpci
$k = 0, 1, \ldots :$
  $Y_{k+1} = AZ_k$
  izračunaj QR razcep $Y_{k+1} = QR$ in vzemi $Z_{k+1} = Q$

Opomba. Pri $p = 1$ je to kar potenčna metoda.

Izrek 22 *Naj velja $A = X\Lambda X^{-1}$, kjer je $X = [x_1 \ \cdots \ x_n]$ in $\Lambda = \operatorname{diag}(\lambda_1, \ldots, \lambda_n)$. Lastne vrednosti naj bodo urejene po absolutni vrednosti in naj velja $|\lambda_p| > |\lambda_{p+1}|$. Potem matrika Z_k iz ortogonalne iteracije konvergira proti ortonormirani bazi za invariantni podprostor* $\operatorname{Lin}(\{x_1, \ldots, x_p\})$.

Dokaz. Očitno je $\operatorname{Lin}(Z_{k+1}) = \operatorname{Lin}(Y_{k+1}) = \operatorname{Lin}(AZ_k)$, od tod pa sledi $\operatorname{Lin}(Z_k) = \operatorname{Lin}(A^k Z_0)$. Ker je $A^k = X\Lambda^k X^{-1}$, velja

$$A^k Z_0 = X\Lambda^k X^{-1} Z_0 = \lambda_p^k X \begin{bmatrix} (\lambda_1/\lambda_p)^k & & & & \\ & \ddots & & & \\ & & 1 & & \\ & & & \ddots & \\ & & & & (\lambda_n/\lambda_p)^k \end{bmatrix} X^{-1} Z_0.$$

Ko gre $k \to \infty$, gre $A^k Z_0$ proti $X \cdot \begin{matrix} p \\ n-p \end{matrix} \begin{pmatrix} \times \\ 0 \end{pmatrix}$, to pa pomeni, da $\mathrm{Lin}(Z_k)$ konvergira proti $\mathrm{Lin}(\{x_1, \ldots, x_p\})$. ■

Podobno za prvih $r < p$ stolpcev velja, da $\mathrm{Lin}(Z_k(:, 1:r))$ konvergira proti $\mathrm{Lin}(\{x_1, \ldots, x_r\})$.

Vzemimo kar $p = n$ in $Z_0 = I$. Pri predpostavki, da so absolutne vrednosti λ_i paroma različne (to hkrati pomeni, da so vse realne), lahko pokažemo, da $A_k := Z_k^T A Z_k$ konvergira proti Schurovi formi.

A_k in $Z_k^T A Z_k$ sta podobni matriki, saj je Z_k ortogonalna matrika. Naj bo $Z_k = [Z_{k1}\ Z_{k2}]$, kjer ima Z_{k1} p stolpcev. Potem je

$$Z_k^T A Z_k = \begin{bmatrix} Z_{k1}^T A Z_{k1} & Z_{k1}^T A Z_{k2} \\ Z_{k2}^T A Z_{k1} & Z_{k2}^T A Z_{k2} \end{bmatrix}.$$

Ker $\mathrm{Lin}(Z_{k1})$ konvergira proti invariantnemu podprostoru $\mathrm{Lin}(\{x_1, \ldots, x_p\})$, enako velja za $\mathrm{Lin}(AZ_{k1})$, to pa pomeni, da $Z_{k2}^T A Z_{k1} \to 0$, saj je $Z_{k2}^T Z_{k1} = 0$. Poddiagonalni elementi A_k konvergirajo proti 0, konvergenca v p-tem stolpcu pa je odvisna od razmerja $\left|\frac{\lambda_{p+1}}{\lambda_p}\right|$.

5.7 QR iteracija

Osnovna varianta QR algoritma je:

$A_0 = A$
$k = 0, 1, \ldots:$
  $A_k = Q_k R_k$ (izračunamo QR razcep)
  $A_{k+1} = R_k Q_k$

Ker je $A_{k+1} = R_k Q_k = Q_k^T A_k Q_k$, sta A_{k+1} in A_k ortogonalno podobni, torej je A_k ortogonalno podobna A.

Lema 17 *Matrika A_k pri QR iteraciji je enaka $A_k = Z_k^T A Z_k$, kjer je Z_k matrika, ki jo dobimo v ortogonalni iteraciji iz $Z_0 = I$. Torej v primeru, ko imajo lastne vrednosti paroma različne absolutne vrednosti, A_k konvergira proti Schurovi formi.*

Dokaz. Naredimo indukcijo po k. Denimo, da je $A_k = Z_k^T A Z_k$. Potem je

$$Z_k^T A Z_k = Z_k^T \underbrace{(\underbrace{Z_{k+1}}_{\text{ort.}}\ \underbrace{S_{k+1}}_{\text{zg. trik.}})}_{\text{QR razcep } AZ_k} = \underbrace{Z_k^T Z_{k+1}}_{\text{ort.}}\ \underbrace{S_{k+1}}_{\text{zg. trik.}} = Q_k R_k,$$

zaradi enoličnosti pa je to kar QR razcep matrike A_k. Sledi

$$A_{k+1} = R_k Q_k = S_{k+1} Z_k^T Z_{k+1} = \underbrace{Z_{k+1}^T A Z_k}_{S_{k+1}} Z_k^T Z_{k+1} = Z_{k+1}^T A Z_{k+1}.$$ ■

V primeru, ko imamo tudi kompleksne lastne vrednosti, dobimo realno Schurovo formo.

Poglejmo, kako lahko QR iteracijo še izboljšamo.

5.7.1 Redukcija na Hessenbergovo obliko

En korak osnovne QR iteracije porabi $\mathcal{O}(n^3)$ operacij, kar ni najbolj ekonomično, saj pričakujemo najmanj $\mathcal{O}(n)$ potrebnih korakov. Na srečo gre hitreje, če matriko A prej reduciramo v zgornjo Hessenbergovo obliko.

Za $A \in \mathbb{R}^{n\times n}$ lahko poiščemo ortogonalno matriko Q, da je QAQ^T zgornja Hessenbergova matrika. To lahko naredimo npr. s Householderjevimi zrcaljenji.

Zgled 26 *Naj bo*

$$A = \begin{bmatrix} \times & \times & \times & \times & \times \\ (\times) & \times & \times & \times & \times \\ (\times) & \times & \times & \times & \times \\ (\times) & \times & \times & \times & \times \\ (\times) & \times & \times & \times & \times \end{bmatrix}.$$

Poiščemo ortogonalno Q_1, da je

$$Q_1A = \begin{bmatrix} \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times \end{bmatrix}, \quad A_1 = Q_1AQ_1^T = \begin{bmatrix} \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ 0 & (\times) & \times & \times & \times \\ 0 & (\times) & \times & \times & \times \\ 0 & (\times) & \times & \times & \times \end{bmatrix}.$$

Nato poiščemo ortogonalno Q_2, da je

$$Q_2A_1 = \begin{bmatrix} \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times \\ 0 & 0 & \times & \times & \times \\ 0 & 0 & \times & \times & \times \end{bmatrix}, \quad A_2 = Q_2A_1Q_2^T = \begin{bmatrix} \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times \\ 0 & 0 & (\times) & \times & \times \\ 0 & 0 & (\times) & \times & \times \end{bmatrix},$$

na koncu pa še ortogonalno Q_3, da je

$$Q_3A_2 = \begin{bmatrix} \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times \\ 0 & 0 & \times & \times & \times \\ 0 & 0 & 0 & \times & \times \end{bmatrix}, \quad H = Q_3A_2Q_3^T = \begin{bmatrix} \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times \\ 0 & 0 & \times & \times & \times \\ 0 & 0 & 0 & \times & \times \end{bmatrix}$$

Tako smo dobili $H = Q_3Q_2Q_1A(Q_3Q_2Q_1)^T$. ■

Algoritem za redukcijo na Hessenbergovo obliko preko Householderjevih zrcaljenj je:

$Q = I$ (*)
$i = 1, \ldots, n-2$
  določi $w_i \in \mathbb{R}^{n-i}$, ki prezrcali $A(i+1:n-i)$ v $\pm ke_1$
  $A(i+1:n, i:n) = P_iA(i+1:n, i:n)$
  $A(1:n, i+1:n) = A(1:n, i+1:n)P_i$
  $Q(i+1:n, i:n) = P_iQ(i+1:n, i:n)$ (*)

(*): če potrebujemo tudi Q

Število operacij je $\frac{10}{3}n^3 + \mathcal{O}(n^2)$ oziroma $\frac{14}{3}n^3 + \mathcal{O}(n^2)$ če potrebujemo tudi Q.

Trditev 1 *Če je A zgornja Hessenbergova, se oblika med QR iteracijo ohranja.*

Dokaz. Pri QR razcepu A dobimo zgornjo Hessenbergovo matriko Q in zgornjo trikotno R. Pri Q je to razvidno iz dejstva, da je q_i linearna kombinacija stolpcev $a_1, \ldots, a_i$. Hitro se da preveriti, da je produkt zgornje trikotne in zgornje Hessenbergove matrike spet zgornja Hessenbergova matrika. ■

Če na začetku A reduciramo na Hessenbergovo obliko, porabimo za en korak QR iteracije le še $\mathcal{O}(n^2)$ namesto $\mathcal{O}(n^3)$ operacij.

Definicija 7 *Hessenbergova matrika H je ireducibilna, če so vsi njeni subdiagonalni elementi $h_{i+1,i}$ neničelni.*

Če je H reducibilna, je npr.

$$H = \begin{bmatrix} \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ 0 & 0 & \times & \times & \times \\ 0 & 0 & \times & \times & \times \\ 0 & 0 & 0 & \times & \times \end{bmatrix}$$

in problem lastnih vrednosti razpade na dva ločena problema. Zaradi tega lahko vedno predpostavimo, da je H ireducibilna.

5.7.2 Premiki

Konvergenco lahko pospešimo s premiki:

$A_0 = A$
$k = 0, 1, \ldots :$
  izberi premik σ_k
  $A_k - \sigma_k I = Q_k R_k$ (izračunamo QR razcep)
  $A_{k+1} = R_k Q_k + \sigma_k I$

Lema 18 *Matriki A_k in A_{k+1} pri QR iteraciji s premikom sta ortogonalno podobni.*

Dokaz.

$$\begin{aligned} A_{k+1} &= R_k Q_k + \sigma_k I = Q_k^T Q_k R_k Q_k + \sigma_k Q_k^T Q_k = \\ &= Q_k^T (Q_k R_k + \sigma_k I) Q_k = Q_k^T A_k Q_k. \end{aligned}$$ ■

Kako izberemo premik? Čim bližje lastni vrednosti je, tem bolje.

Lema 19 *Če je σ lastna vrednost ireducibilne Hessenbergove matrike A in $A - \sigma I = QR$, $B = RQ + \sigma I$, potem je $b_{n,n-1} = 0$ in $b_{nn} = \sigma$.*

Dokaz. Ker je A ireducibilna, je prvih $n-1$ stolpcev $A - \sigma I$ linearno neodvisnih. V razcepu $A - \sigma I = QR$ zato velja $r_{ii} \neq 0$ za $i = 1, \ldots, n-1$. Ker pa je $A - \sigma I$ singularna, mora biti $r_{nn} = 0$. To pa pomeni, da je zadnja vrstica v matriki RQ enaka 0, torej v matriki $B = RQ + \sigma I$ velja $b_{n,n-1} = 0$ in $b_{nn} = \sigma$. ■

Ko najdemo eno lastno vrednost, nadaljujemo računanje z matriko $B(1 : n-1, 1 : n-1)$.

Potrebujemo čim boljši približek za lastno vrednost A_k, zato imamo na voljo različne premike:

a) *enojni premik*: Za σ_k izberemo $(A_k)_{nn}$. V tem primeru imamo kvadratično konvergenco v bližini lastne vrednosti, vendar premik ni dober za kompleksne lastne vrednosti.

b) *dvojni* oz. *Francisov premik*: Vzamemo podmatriko

$$A_k(n-1 : n, n-1 : n) = \begin{bmatrix} a_{n-1,n-1}^{(k)} & a_{n-1,n}^{(k)} \\ a_{n,n-1}^{(k)} & a_{nn}^{(k)} \end{bmatrix},$$

ki ima lastni vrednosti $\sigma_1^{(k)}, \sigma_2^{(k)}$ (lahko sta tudi kompleksni). Sedaj naredimo dva premika v enem koraku:

$$\begin{aligned} &A_k - \sigma_1^{(k)} I = Q_k R_k \text{ (izračunamo QR razcep)} \\ &A_k' = R_k Q_k + \sigma_1^{(k)} I \\ &A_k' - \sigma_2^{(k)} I = Q_k' R_k' \text{ (izračunamo QR razcep)} \\ &A_{k+1} = R_k' Q_k' + \sigma_2^{(k)} I \end{aligned}$$

Kompleksna aritmetika ni potrebna, saj velja:

$$\begin{aligned} Q_k Q_k' R_k' R_k &= Q_k (A_k' - \sigma_2^{(k)} I) R_k = \\ &= Q_k Q_k^* (A_k - \sigma_2^{(k)} I) Q_k R_k = (A_k - \sigma_2^{(k)} I) Q_k R_k = \\ &= (A_k - \sigma_2^{(k)} I)(A_k - \sigma_1^{(k)} I) = \\ &= A_k^2 - (\sigma_1^{(k)} + \sigma_2^{(k)}) A_k + \sigma_1^{(k)} \sigma_2^{(k)} I =: N \end{aligned}$$

in $(Q_k Q_k')(R_k' R_k)$ je QR razcep realne matrike N. Ker velja tudi:

$$\begin{aligned} A_{k+1} &= R_k' Q_k' + \sigma_2^{(k)} I = Q_k'^* (A_k' - \sigma_2^{(k)} I) Q_k' + \sigma_2^{(k)} I = \\ &= Q_k'^* (R_k Q_k + (\sigma_1^{(k)} - \sigma_2^{(k)} I) Q_k' + \sigma_2^{(k)} I = \\ &= Q_k'^* (Q_k^* (A_k - \sigma_1^{(k)} I) Q_k + (\sigma_1^{(k)} - \sigma_2^{(k)} I) Q_k' + \sigma_2^{(k)} I = \\ &= Q_k'^* Q_k A_k Q_k Q_k', \end{aligned}$$

potrebujemo le realni QR razcep realne matrike N.

5.8 Implicitna QR metoda

Izrek 23 (Implicitni Q) *Če je $Q^T A Q = H$ ireducibilna Hessenbergova matrika in $Q = [q_1 \ \cdots \ q_n]$, so stolpci $q_2, \ldots, q_n$ do predznaka natančno določeni s q_1.*

Dokaz. Denimo, da je $Q^TAQ = H$ in $V^TAV = G$, kjer sta $Q = [q_1 \ \cdots \ q_n]$, $V = [v_1 \ \cdots \ v_n]$ ortogonalni matriki, G, H Hessenbergovi matriki in $q_1 = v_1$.

Če definiramo $W := V^TQ$, $W = [w_1 \ \cdots \ w_n]$, je W ortogonalna matrika in $w_1 = e_1$. Velja

$$GW = GV^TQ = V^TAQ = V^TQH = WH.$$

Iz te zveze sledi $Gw_i = \sum_{j=1}^{i+1} h_{ji}w_j$ oziroma

$$h_{i+1,i}w_{i+1} = Gw_i - \sum_{j=1}^{i} h_{ji}w_j.$$

Ker je $w_1 = e_1$ in ima Gw_i en neničelni element več od w_i, sledi $w_i \in \mathrm{Lin}(\{e_1, \ldots, e_i\})$. To pomeni, da je W zgornja trikotna matrika. Ker pa je W hkrati ortogonalna, je edina možnost $W = \mathrm{diag}(\pm 1, \ldots, \pm 1)$, torej $v_i = \pm q_i$, $i = 2, \ldots, n$. ■

Posledica je, da če v QR algoritmu $A_k = Q_kR_k$, $A_{k+1} = R_kQ_k = Q_k^TA_kQ_k$ poznamo prvi stolpec Q_k, potem lahko A_{k+1} izračunamo brez tega, da bi računali QR razcep matrike A_k. Tako dobimo *implicitni QR razcep*.

Poglejmo si implicitni QR razcep z enojnim premikom. Vemo, da je prvi stolpec Q_k enak normiranemu prvemu stolpcu $A_k - \sigma_k I$. Če uspemo poiskati tako ortogonalno matriko Q_k, ki bo imela za prvi stolpec normirani prvi stolpec $A_k - \sigma_k I$ in bo $Q_k^TA_kQ_k$ zgornja Hessenbergova matrika, potem je po izreku o implicitnem Q matrika $Q_k^TA_kQ_k$ kar matrika iz naslednjega koraka QR metode.

Matriko Q_k poiščemo kot produkt Givensovih rotacij $Q_k = R_{12}R_{23}\cdots R_{n-1,n}$. Prva rotacija R_{12} je že določena s prvim stolpcem $A_k - \sigma_k I$, ostale pa določimo tako, da bo $Q_k^TA_kQ_k$ zgornja Hessenbergova matrika. Če je namreč

$$R_{12} = \begin{bmatrix} c_1 & s_1 & & & \\ s_1 & c_1 & & & \\ & & 1 & & \\ & & & \ddots & \\ & & & & 1 \end{bmatrix},$$

potem je

$$Q_k = R_{12}R_{23}\cdots R_{n-1,n} = \begin{bmatrix} c_1 & \times & \cdots & \cdots & \times \\ s_1 & \times & \cdots & \cdots & \times \\ & \times & & & \vdots \\ & & \ddots & & \vdots \\ & & & \times & \times \end{bmatrix}.$$

Algoritem najkrajše označimo kot *premikanje grbe*. Poglejmo si ga na primeru matrike 5×5. Po prvem koraku dobimo

$$R_{12}^TA_kR_{12} = \begin{bmatrix} \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ + & \times & \times & \times & \times \\ & & \times & \times & \times \\ & & & \times & \times \end{bmatrix}.$$

Novi neničelni element + je grba, ki jo z naslednjimi rotacijami pomikamo navzdol ob diagonali. Tako po vrsti poiščemo R_{23}, R_{34} in R_{45}, da je

$$R_{23}^T R_{12}^T A_k R_{12} R_{23} = \begin{bmatrix} \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ & \times & \times & \times & \times \\ & + & \times & \times & \times \\ & & & \times & \times \end{bmatrix},$$

$$R_{34}^T R_{23}^T R_{12}^T A_k R_{12} R_{23} R_{34} = \begin{bmatrix} \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ & \times & \times & \times & \times \\ & & \times & \times & \times \\ & & + & \times & \times \end{bmatrix}$$

in

$$R_{45}^T R_{34}^T R_{23}^T R_{12}^T A_k R_{12} R_{23} R_{34} R_{45} = \begin{bmatrix} \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ & \times & \times & \times & \times \\ & & \times & \times & \times \\ & & & \times & \times \end{bmatrix}.$$

Podobno lahko naredimo tudi za dvojni premik. Ker vemo $A_{k+1} = U_k A_k U_k$, kjer je U_k iz QR razcepa matrike $N = A_k^2 - (\sigma_1^{(k)} + \sigma_2^{(k)}) A_k + \sigma_1^{(k)} \sigma_2^{(k)} I$, je dovolj poznati le prvi stolpec matrike N. Le ta ima obliko $[\times \ \times \ \times \ 0 \ \cdots \ 0]^T$, za neničelne elemente pa lahko izpeljemo direktne formule.

Sedaj najprej poiščemo Householderjevo zrcaljenje oblike

$$P_1 = \begin{bmatrix} \times & \times & \times & & & \\ \times & \times & \times & & & \\ \times & \times & \times & & & \\ & & & 1 & & \\ & & & & \ddots & \\ & & & & & 1 \end{bmatrix},$$

ki ima prvi stolpec enak normiranemu prvemu stolpcu matrike N. Dobimo grbo 2×2, ki jo premikamo navzdol s Householderjevimi zrcaljenji. V primeru 6×6 imamo

$$P_1 A_k P_1 = \begin{bmatrix} \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \\ + & \times & \times & \times & \times & \times \\ + & + & \times & \times & \times & \times \\ & & & \times & \times & \times \\ & & & & \times & \times \end{bmatrix},$$

$$P_2 = \begin{bmatrix} 1 & & & & & \\ & \times & \times & \times & & \\ & \times & \times & \times & & \\ & \times & \times & \times & & \\ & & & & 1 & \\ & & & & & 1 \end{bmatrix}, \quad P_2 P_1 A_k P_1 P_2 = \begin{bmatrix} \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \\ & \times & \times & \times & \times & \times \\ & + & \times & \times & \times & \times \\ & + & + & \times & \times & \times \\ & & & & \times & \times \end{bmatrix},$$

$$P_3 = \begin{bmatrix} 1 & & & & & \\ & 1 & & & & \\ & & \times & \times & \times & \\ & & \times & \times & \times & \\ & & \times & \times & \times & \\ & & & & & 1 \end{bmatrix}, \quad P_3P_2P_1A_kP_1P_2P_3 = \begin{bmatrix} \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \\ & \times & \times & \times & \times & \times \\ & & \times & \times & \times & \times \\ & & + & \times & \times & \times \\ & & + & + & \times & \times \end{bmatrix},$$

$$P_4 = \begin{bmatrix} 1 & & & & & \\ & 1 & & & & \\ & & 1 & & & \\ & & & \times & \times & \times \\ & & & \times & \times & \times \\ & & & \times & \times & \times \end{bmatrix}, \quad P_4P_3P_2P_1A_kP_1P_2P_3P_4 = \begin{bmatrix} \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \\ & \times & \times & \times & \times & \times \\ & & \times & \times & \times & \times \\ & & & \times & \times & \times \\ & & & + & \times & \times \end{bmatrix},$$

$$P_5 = \begin{bmatrix} 1 & & & & & \\ & 1 & & & & \\ & & 1 & & & \\ & & & 1 & & \\ & & & & \times & \times \\ & & & & \times & \times \end{bmatrix}, \quad P_5P_4P_3P_2P_1A_kP_1P_2P_3P_4P_5 = \begin{bmatrix} \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \\ & \times & \times & \times & \times & \times \\ & & \times & \times & \times & \times \\ & & & \times & \times & \times \\ & & & & \times & \times \end{bmatrix}.$$

VI. Simetrični problem lastnih vrednosti

6.1 Uvod

Če je $A = A^T$, potem vemo, da so vse lastne vrednosti realne, matrika pa se da diagonalizirati. Schurova forma za simetrično matriko je diagonalna matrika.

Lastne vrednosti označimo tako, da velja

$$\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n.$$

Lastne vektorje $x_1, \ldots, x_n$ lahko izberemo tako, da tvorijo ortonormirano bazo. Za $A = A^T$ velja

$$\|A\|_2 = \max(|\lambda_1|, |\lambda_n|).$$

Za $x \neq 0$ in $A = A^T$ definiramo *Rayleighov kvocient*

$$\rho(x, A) = \frac{x^T A x}{x^T x}.$$

Nekaj lastnosti Rayleighovega kvocienta:

a) Za $\alpha \neq 0$ je $\rho(x, A) = \rho(\alpha x, A)$.

b) $\rho(x_i, A) = \lambda_i$.

c) Za vsak $x \neq 0$ velja $\lambda_1 \geq \rho(x, A) \geq \lambda_n$.
   To vidimo, če razvijemo $x = \sum_{i=1}^n \alpha_i x_i$. Potem dobimo

$$\rho(x, A) = \frac{\sum_{i=1}^n \alpha_i^2 \lambda_i}{\sum_{i=1}^n \alpha_i^2}.$$

d) Če si mislimo, da je x približek za lastni vektor, je $\rho(x, A)$ najboljša aproksimacija za lastno vrednost v smislu, da je $\min_\sigma \|Ax - \sigma x\|_2$ dosežen pri $\sigma = \rho(x, A)$.
   To vidimo iz predoločenega sistema $x\sigma = Ax$.

Izrek 24 (Courant-Fischerjev minimaks izrek)

$$\lambda_i = \min_{\substack{S \subset \mathbb{R}^n \\ \dim(S) = n-i+1}} \max_{\substack{x \in S \\ x \neq 0}} \rho(x, A) = \max_{\substack{R \subset \mathbb{R}^n \\ \dim(R) = i}} \min_{\substack{x \in R \\ x \neq 0}} \rho(x, A). \tag{6.8}$$

Dokaz. Za poljubna podprostora $S, R \subset \mathbb{R}^n$, $\dim(R) = i$ in $\dim(S) = n - i + 1$, obstaja $x_{RS} \in R \cap S$, $x_{RS} \neq 0$, saj je $\dim(R) + \dim(S) = n + 1$. Očitno velja

$$\min_{\substack{x \in R \\ x \neq 0}} \rho(x, A) \leq \rho(x_{RS}, A) \leq \max_{\substack{x \in S \\ x \neq 0}} \rho(x, A).$$

Ker to velja za vsak par R, S velja tudi za par $\widetilde{R}, \widetilde{S}$, kjer je dosežen minimum oz. maksimum izrazov. Torej

$$\max_{\substack{R\subset\mathbb{R}^n \\ \dim(R)=i}} \min_{\substack{x\in R \\ x\neq 0}} \rho(x,A) \leq \rho(x_{\widetilde{R}\widetilde{S}},A) \leq \min_{\substack{S\subset\mathbb{R}^n \\ \dim(S)=n-i+1}} \max_{\substack{x\in S \\ x\neq 0}} \rho(x,A). \tag{6.9}$$

Po drugi strani pa za par $\widehat{R} = \mathrm{Lin}(x_1,\ldots,x_i)$ in $\widehat{S} = \mathrm{Lin}(x_i,\ldots,x_n)$ velja

$$\min_{\substack{x\in \widehat{R} \\ x\neq 0}} \rho(x,A) = \lambda_i = \max_{\substack{x\in \widehat{S} \\ x\neq 0}} \rho(x,A).$$

Od tod sledi

$$\max_{\substack{R\subset\mathbb{R}^n \\ \dim(R)=i}} \min_{\substack{x\in R \\ x\neq 0}} \rho(x,A) \geq \min_{\substack{S\subset\mathbb{R}^n \\ \dim(S)=n-i+1}} \max_{\substack{x\in S \\ x\neq 0}} \rho(x,A), \tag{6.10}$$

saj je

$$\max_{\substack{R\subset\mathbb{R}^n \\ \dim(R)=i}} \min_{\substack{x\in R \\ x\neq 0}} \rho(x,A) \geq \min_{\substack{x\in \widehat{R} \\ x\neq 0}} \rho(x,A)$$

in podobno

$$\min_{\substack{S\subset\mathbb{R}^n \\ \dim(S)=n-i+1}} \max_{\substack{x\in S \\ x\neq 0}} \rho(x,A) \leq \max_{\substack{x\in \widehat{S} \\ x\neq 0}} \rho(x,A).$$

Iz (6.9) in (6.10) sledi (6.8). ■

Izrek 25 (Weylov izrek) *Če sta A, E simetrični matriki in so $\alpha_1 \geq \cdots \geq \alpha_n$ lastne vrednosti A, $\widehat{\alpha}_1 \geq \cdots \widehat{\alpha}_n$ pa lastne vrednosti $A+E$, potem za $i = 1,\ldots,n$ velja*

$$|\alpha_i - \widehat{\alpha}_i| \leq \|E\|_2.$$

Dokaz. Ocenimo lahko

$$\rho(x, A+E) = \rho(x,A) + \rho(x,E) \leq \rho(x,A) + \|E\|_2$$

in podobno

$$\rho(x, A+E) \geq \rho(x,A) - \|E\|_2.$$

Sedaj po minimaks izreku sledi $|\widehat{\alpha}_i - \alpha_i| \leq \|E\|_2$. ■

Izrek 26 (Cauchyjev izrek o prepletanju) *Če je A simetrična matrika in je A_r vodilna $r \times r$ podmatrika matrike A, potem velja*

$$\lambda_{r+1}(A_{r+1}) \leq \lambda_r(A_r) \leq \lambda_r(A_{r+1}) \leq \cdots \leq \lambda_2(A_{r+1}) \leq \lambda_1(A_r) \leq \lambda_1(A_{r+1}).$$

Dokaz. Dovolj je dokazati primer $r = n-1$. Če je $x' \in \mathbb{R}^{n-1}$ in $x = \begin{bmatrix} x' \\ 0 \end{bmatrix} \in \mathbb{R}^n$, potem je $\rho(x', A_{n-1}) = \rho(x, A)$.

Po minimaks izreku velja

$$\lambda_k(A_{n-1}) = \min_{\substack{S'\subset\mathbb{R}^{n-1} \\ \dim(S')=n-k}} \max_{\substack{x'\in S' \\ x'\neq 0}} \rho(x', A_{n-1}).$$

Vsak podprostor lahko podamo z njegovim ortogonalnim komplementom, zato lahko pišemo

$$\lambda_k(A_{n-1}) = \min_{p'_1,\ldots,p'_{k-1}\in\mathbb{R}^{n-1}} \max_{\substack{x'\in\mathbb{R}^{n-1},\ x'\neq 0\\ x'\perp p'_i,\ i=1,\ldots,k-1}} \rho(x', A_{n-1}).$$

Pogoj, da morajo biti vektorji $p'_1, \ldots, p'_{k-1}$ linearno neodvisni, lahko izpustimo, saj je minimum očitno dosežen pri linearno neodvisnih vektorjih. Sedaj lahko pišemo

$$\begin{aligned}\lambda_k(A_{n-1}) &= \min_{p_1,\ldots,p_{k-1}\in\mathbb{R}^n} \max_{\substack{x\in\mathbb{R}^n,\ x\neq 0,\ x\perp e_n\\ x\perp p_i,\ i=1,\ldots,k-1}} \rho(x, A) \leq \\ &\leq \min_{p_1,\ldots,p_{k-1}\in\mathbb{R}^n} \max_{\substack{x\in\mathbb{R}^n,\ x\neq 0\\ x\perp p_i,\ i=1,\ldots,k-1}} \rho(x, A) = \lambda_k(A).\end{aligned}$$

Tako smo pokazali $\lambda_k(A_{n-1}) \leq \lambda_k(A)$. Po drugi strani pa velja

$$\begin{aligned}\lambda_k(A_{n-1}) &= \min_{p_1,\ldots,p_{k-1}\in\mathbb{R}^n} \max_{\substack{x\in\mathbb{R}^n,\ x\neq 0,\ x\perp e_n\\ x\perp p_i,\ i=1,\ldots,k-1}} \rho(x, A) = \\ &= \min_{\substack{p_1,\ldots,p_{k-1},p_k\in\mathbb{R}^n\\ p_k=e_n}} \max_{\substack{x\in\mathbb{R}^n,\ x\neq 0\\ x\perp p_i,\ i=1,\ldots,k}} \rho(x, A) \geq \\ &\geq \min_{p_1,\ldots,p_{k-1},p_k\in\mathbb{R}^n} \max_{\substack{x\in\mathbb{R}^n,\ x\neq 0\\ x\perp p_i,\ i=1,\ldots,k}} \rho(x, A) = \lambda_{k-1}(A). \quad \blacksquare\end{aligned}$$

6.2 Rayleighova iteracija

Inverzno iteracijo kombiniramo z Rayleighovim kvocientom in dobimo:

izberi $\widetilde{z}_0 \neq 0$, $z_0 = \frac{1}{\|\widetilde{z}_0\|_\infty}\widetilde{z}_0$
$k = 0, 1, \ldots:$
  $\sigma_k = \rho(z_k, A)$
  reši $(A - \sigma_k I)\widetilde{z}_{k+1} = z_k$
  $z_{k+1} = \frac{1}{\|\widetilde{z}_{k+1}\|_\infty}\widetilde{z}_{k+1}$

Namesto fiksnega premika σ pri inverzni iteraciji uporabljamo Rayleighov kvocient, ki je najboljši približek za lastno vrednost danega vektorja.

Konvergenca Rayleighove iteracije v bližini enostavne lastne vrednosti je kubična.

Zgled 27 *Naj bo* $A = \begin{bmatrix}\lambda_1 & 0\\ 0 & \lambda_2\end{bmatrix}$, $\lambda_1 > \lambda_2$ *in* $z_r = \begin{bmatrix}c_r\\ s_r\end{bmatrix}$, $c_r^2 + s_r^2 = 1$. *Pri enem koraku Rayleighove iteracije dobimo*

$$\sigma_r = z_r^T A z_r = c_r^2\lambda_1 + \sigma_r^2\lambda_2.$$

Iz sistema

$$\begin{bmatrix}\lambda_1 - c_r^2\lambda_1 - s_r^2\lambda_2 & 0\\ 0 & \lambda_2 - c_r^2\lambda_1 - s_r^2\lambda_2\end{bmatrix}\widetilde{z}_{r+1} = z_r$$

dobimo

$$\widetilde{z}_{r+1} = \frac{1}{(\lambda_1 - \lambda_2)c_r^2 s_r^2}\begin{bmatrix}c_r^3\\ -s_r^3\end{bmatrix},$$

od tod pa sledi

$$z_{r+1} = \frac{1}{\sqrt{c_r^6 + s_r^6}} \begin{bmatrix} c_r^3 \\ -s_r^3 \end{bmatrix}.$$

Od tod v primeru $s_r \neq c_r$ sledi kubična konvergenca proti e_1 oziroma e_2. ■

6.3 QR iteracija

V primeru simetrične matrike je zgornja Hessenbergova matrika tridiagonalna. Za redukcijo na tridiagonalno obliko še vedno porabimo $\mathcal{O}(n^3)$, zaradi tridiagonalne oblike pa en korak QR iteracije sedaj lahko izvedemo v $\mathcal{O}(n)$ namesto v $\mathcal{O}(n^2)$ kot pri nesimetričnem problemu.

Pri QR iteraciji torej najprej poiščemo ortogonalno matriko Q, da je $T = Q^T A Q$ tridiagonalna, potem pa delamo QR z enojnim premikom.

Naj bo

$$T_k = \begin{bmatrix} a_1^{(k)} & b_1^{(k)} & & & \\ b_1^{(k)} & a_2^{(k)} & b_2^{(k)} & & \\ & \ddots & \ddots & \ddots & \\ & & b_{n-2}^{(k)} & a_{n-1}^{(k)} & b_{n-1}^{(k)} \\ & & & b_{n-1}^{(k)} & a_n^{(k)} \end{bmatrix}.$$

Kako izberemo premik:

- $\sigma_k = a_n^{(k)}$: V tem primeru imamo za skoraj vse matrike zagotovljeno kubično konvergenco, a vseeno obstajajo primeri, ko metoda ne konvergira.
- *Wilkinsonov premik*: Za σ_k vzamemo tisto lastno vrednost matrike $\begin{bmatrix} a_{n-1}^{(k)} & b_{n-1}^{(k)} \\ b_{n-1}^{(k)} & a_n^{(k)} \end{bmatrix}$, ki je bližja $a_n^{(k)}$. Sedaj imamo za vse matrike dokazano vsaj kvadratično konvergenco, v praksi pa imamo za skoraj vse matrike kubično konvergenco (a brez dokaza).

6.4 Sturmovo zaporedje

Naj bo

$$A_n = \begin{bmatrix} a_1 & b_1 & & & \\ b_1 & a_2 & b_2 & & \\ & \ddots & \ddots & \ddots & \\ & & b_{n-2} & a_{n-1} & b_{n-1} \\ & & & b_{n-1} & a_n \end{bmatrix}$$

ireducibilna matrika ($b_i \neq 0$ za vsak i). Če definiramo $f_n(\lambda) = \det(A_n - \lambda I)$, potem z razvijanjem po zadnji vrstici pridemo do rekurzivne formule

$$f_{r+1}(\lambda) = (a_{r+1} - \lambda) f_r(\lambda) - b_r^2 f_{r-1}(\lambda),$$

ki se začne z $f_0(\lambda) = 1$ in $f_1(\lambda) = a_1 - \lambda$.

Izrek 27 *Polinomi $f_0, \ldots, f_n$ tvorijo Sturmovo zaporedje oziroma velja:*

1. *$f_0(\lambda) \neq 0$ za vsak λ.*
2. *Če je $f_r(\lambda_0) = 0$ za $r < n$, potem je $f_{r-1}(\lambda_0) f_{r+1}(\lambda_0) < 0$.*
3. *Če je $f_n(\lambda_0) = 0$, potem je $f_{n-1}(\lambda_0) f_n'(\lambda_0) < 0$.*

Dokaz. Točka 1) je očitna. Pri točki 2) iz rekurzivne formule sledi $f_{r+1}(\lambda_0) = -b_r^2 f_{r-1}(\lambda)$. Obe vrednosti sta neničelni, saj bi sicer veljalo $f_i(\lambda_0) = 0$ za $i = 0, \ldots, n$, to pa je v protislovju s točko 1).

Pri točki 3) definiramo $\Delta_r(\lambda_0) = f_r(\lambda_0) f_{r-1}'(\lambda_0) - f_{r-1}(\lambda_0) f_r'(\lambda_0)$. Z računanjem lahko hitro preverimo, da velja

$$\Delta_{r+1}(\lambda_0) = f_r^2(\lambda_0) + b_r^2 \Delta_r(\lambda_0).$$

Ker je $\Delta_1(\lambda_0) = 1 > 0$, je $\Delta_r(\lambda_0) > 0$ za vsak r. Torej tudi $\Delta_n(\lambda_0) = -f_{n-1}(\lambda_0) f_n'(\lambda_0) > 0$. ■

Posledica 3 *Ireducibilna tridiagonalna simetrična matrika ima enostavne lastne vrednosti.*

Dokaz. To sledi iz točke 3) prejšnjega izreka, saj v primeru $f_n(\lambda_0) = 0$ velja $f_n'(\lambda_0) \neq 0$. ■

Označimo z $u(\lambda)$ število ujemanj predznaka v zaporedju $f_0(\lambda), \ldots, f_n(\lambda)$. Pri tem vsako notranjo ničlo štejemo za eno ujemanje, ničlo na koncu pa ne. Primeri: $u(++--+) = 2$, $u(++0-+) = 2$, $u(++0-+0) = 2$.

Lema 20 *Število $u(\lambda_0)$ je enako številu lastnih vrednosti, ki so strogo večje od λ_0.*

Dokaz. Naj λ teče od $-\infty$ do ∞. Pri $\lambda = -\infty$ imamo očitno zaporedje $+++\cdots$, pri $\lambda = \infty$ pa $+-+-\cdots$. Tako je $u(-\infty) = n$ in $u(\infty) = 0$.

Pokazali bomo, da se število $u(\lambda)$ lahko spremeni le, če prečkamo ničlo f_n, ne pa tudi, če prečkamo ničlo f_r, $r < n$.

Naj bo $f_r(\lambda_0) = 0$, $r < n$. Potem je iz tabele

	$\lambda_0 - \epsilon$	λ_0	$\lambda_0 + \epsilon$
f_{r-1}	$\pm$	$\pm$	$\pm$
f_r	$\pm$	0	$\mp$
f_{r+1}	$\mp$	$\mp$	$\mp$

razvidno, da pri zadosti majhnem $\epsilon > 0$ velja $u(\lambda_0 - \epsilon) = u(\lambda_0 + \epsilon)$.

V primeru $f_n(\lambda_0) = 0$ pa iz

	$\lambda_0 - \epsilon$	λ_0	$\lambda_0 + \epsilon$
f_{r-1}	$\pm$	$\pm$	$\pm$
f_n	$\pm$	0	$\mp$

sledi, da pri zadosti majhnem $\epsilon > 0$ velja $u(\lambda_0 - \epsilon) = u(\lambda_0 + \epsilon) + 1$. ■

Sedaj lahko z bisekcijo ali kakšno drugo metodo poiščemo k-to lastno vrednost. Iščemo točko λ_k, za katero velja $u(\lambda_k - \epsilon) = k$ in $u(\lambda_k + \epsilon) = k + 1$ za dovolj majhen $\epsilon > 0$.

Metoda je uporabna tudi, če nas zanimajo samo lastne vrednosti na določenem intervalu ali pa nekaj lastnih vrednosti.

6.5 Deli in vladaj

To je trenutno najhitrejša metoda za ireducibilno simetrično tridiagonalno matriko v široki uporabi, obstaja pa že hitrejša metoda RRR. Naj bo

$$T = \begin{bmatrix} a_1 & b_1 & & 0 \\ b_1 & \ddots & \ddots & \\ & \ddots & \ddots & b_{n-1} \\ 0 & & b_{n-1} & a_n \end{bmatrix}.$$

Za $m < n$ razdelimo T kot

$$T = \begin{bmatrix} T_1 & 0 \\ 0 & T_2 \end{bmatrix} + b_m v v^T,$$

kjer je

$$T_1 = \begin{bmatrix} a_1 & b_1 & & 0 \\ b_1 & \ddots & \ddots & \\ & \ddots & a_{m-1} & b_{m-1} \\ 0 & & b_{m-1} & a_m - b_m \end{bmatrix}, \quad T_2 = \begin{bmatrix} a_{m+1} - b_m & b_{m+1} & & 0 \\ b_{m+1} & a_{m+2} & \ddots & \\ & \ddots & \ddots & b_{n-1} \\ 0 & & b_{n-1} & a_n \end{bmatrix}$$

in

$$v = [0 \ \cdots \ 0 \ 1 \ 1 \ 0 \cdots \ 0]^T.$$

T_1 in T_2 sta simetrični tridiagonalni matriki, zato obstajata ortogonalni matriki Q_1, Q_2 in diagonalni D_1, D_2, da je $T_1 = Q_1 D_1 Q_1^T$ in $T_2 = Q_2 D_2 Q_2^T$. Potem je

$$T = \begin{bmatrix} Q_1 & 0 \\ 0 & Q_2 \end{bmatrix} \left(\begin{bmatrix} D_1 & 0 \\ 0 & D_2 \end{bmatrix} + b_m u u^T \right) \begin{bmatrix} Q_1^T & 0 \\ 0 & Q_2^T \end{bmatrix},$$

kjer je

$$u = \begin{bmatrix} Q_1^T & 0 \\ 0 & Q_2^T \end{bmatrix} v = \begin{bmatrix} Q_1(m,:)^T \\ Q_2(1,:)^T \end{bmatrix}.$$

Lastne vrednosti T so tako enake lastnim vrednostim $D + \rho u u^T$, kjer je $D = \begin{bmatrix} D_1 & 0 \\ 0 & D_2 \end{bmatrix}$ in $\rho = b_m$. Če izračunamo lastne vrednosti in vektorje $D + \rho u u^T = Q' D Q'^T$, potem so v D lastne vrednosti T, $Q = \begin{bmatrix} Q_1 & 0 \\ 0 & Q_2 \end{bmatrix} Q'$ pa so ustrezni lastni vektorji.

Algoritem v grobem je:

$[Q, D]$=deliinvladaj(T)
  če je T velikosti 1×1, potem vrni $Q = 1$, $D = T$

sicer:
razdeli $T = \begin{bmatrix} T_1 & 0 \\ 0 & T_2 \end{bmatrix} + b_m v v^T$.
$[Q_1, D_1]$=deliinvladaj(T_1)
$[Q_2, D_2]$=deliinvladaj(T_2)
iz Q_1, Q_2, D_1, D_2 izračunaj $D + \rho u u^T$
izračunaj lastne vrednosti D in vektorje Q' za $D + \rho u u^T$
$Q = \begin{bmatrix} Q_1 & 0 \\ 0 & Q_2 \end{bmatrix} Q'$.

Kako izračunamo lastne vrednosti in vektorje za $D + \rho u u^T$? Za samo računanje uredimo diagonalne elemente D tako, da je $d_1 \geq d_2 \geq \cdots \geq d_n$. Seveda moramo ustrezno preurediti tudi elemente vektorja u.

Naslednji dve lemi dokažemo na vajah:

Lema 21 *Naj bo $A = D + \rho u u^T$, $D = \mathrm{diag}(d_1, \ldots, d_n)$, $d_1 \geq d_2 \geq \cdots \geq d_n$, $u = [u_1 \ \cdots \ u_n]^T$.*

a) Če je $d_i = d_{i+1}$ je d_i lastna vrednost A, lastni vektor pa je $[0 \ \cdots \ 0 \ -u_{i+1} \ u_i \ 0 \ \cdots \ 0]^T$.

b) Če je $u_i = 0$ je d_i lastna vrednost A, lastni vektor pa je e_i. ■

Lema 22 *Za $x, y \in \mathbb{R}^n$ velja $\det(I + xy^T) = 1 + y^T x$.* ■

Opomba: Več kot dva d_i ne moreta biti enaka, saj bi potem T_1 ali T_2 morala imeti večkratno lastno vrednost.

Predpostavimo, da so vsi d_i različni in da je $u_i \neq 0$ za $i = 1, \ldots, n$. Predpostavimo še, da je matrika λ lastna vrednost $D + \rho u u^T$, matrika $D - \lambda I$ pa je nesingularna. Potem je

$$\det(D + \rho u u^T - \lambda I) = \det\Big((D - \lambda I)(I + \rho (D - \lambda I)^{-1} u u^T)\Big),$$

od tod pa sledi

$$\det(I + \rho (D - \lambda I)^{-1} u u^T) = 0.$$

Ker je $\det(I + xy^T) = 1 + y^T x$, so lastne vrednosti $D + \rho u u^T$ ničle *sekularne enačbe* $f(\lambda) = 0$, kjer je

$$\begin{aligned} f(\lambda) &= \det(I + \rho (D - \lambda I)^{-1} u u^T) = 1 + \rho u^T (D - \lambda I)^{-1} u = \\ &= 1 + \rho \sum_{i=1}^{n} \frac{u_i^2}{d_i - \lambda}. \end{aligned}$$

Kako zgleda graf $f(\lambda)$? Asimptota je $y = 1$. Ker je $f'(\lambda) = \rho \sum_{i=1}^{n} \frac{u_i^2}{(d_i - \lambda)^2}$, je za $\rho > 0$ funkcija strogo naraščajoča (med poli), sicer pa padajoča. Ničle ležijo med poli, ena pa desno od zadnjega pola (pri $\rho > 0$) ali levo od prvega pola (pri $\rho < 0$).

Denimo, da z neko numerično metodo, npr. Newtonovo, poiščemo lastne vrednosti. Za lastne vektorje potem velja:

Lema 23 *Če je α lastna vrednost $D+\rho uu^T$, je $(D-\alpha I)^{-1}u$ ustrezni lastni vektor.*

Dokaz.

$$\begin{aligned}(D+\rho uu^T)(D-\alpha I)^{-1}u &= (D-\alpha I+\alpha I+\rho uu^T)(D-\alpha I)^{-1}u = \\ &= u+\alpha(D-\alpha I)^{-1}u+u(\rho u^T(D-\alpha I)^{-1}u) = \\ &= u+\alpha(D-\alpha I)^{-1}u-u=\alpha(D-\alpha I)^{-1}u,\end{aligned}$$

saj iz $f(\alpha)=1+\rho u^T(D-\alpha I)^{-1}u=0$ sledi $\rho u^T(D-\alpha I)^{-1}u=-1$. ■

Ker rešujemo diagonalni sistem, lahko vsak lastni vektor izračunamo v $\mathcal{O}(n)$.

Za konec si podrobno poglejmo, kako rešujemo sekularno enačbo. Navadne tangentne metode ne moremo uporabiti, saj so ničle lahko zelo blizu polov in bi potrebovali zelo dobre približke. Namesto aproksimacije funkcije s tangento zato raje uporabimo preprosto racionalno funkcijo, ki se prilega funkciji f.

Denimo, da iščemo rešitev na intervalu (d_{i+1},d_i), začetni približek pa je x_r. Sedaj poiščemo racionalno funkcijo oblike

$$h(\lambda)=\frac{c_1}{d_i-\lambda}+\frac{c_2}{d_{i+1}-\lambda}+c_3,$$

za katero velja $h(x_r)=f(x_r)$ in $f'(x_r)=h'(x_r)$. Zaradi stabilnosti razdelimo f na dva dela kot

$$f(\lambda)=1+\sum_{k=1}^{i}\frac{u_k^2}{d_k-\lambda}+\sum_{k=i+1}^{n}\frac{u_k^2}{d_k-\lambda}=:1+\psi_1(\lambda)+\psi_2(\lambda).$$

To naredimo zato, da v vsoti za $\psi_1(\lambda)$ oziroma $\psi_2(\lambda)$ seštevamo enako predznačene člene. Sedaj določimo c_1,c_1' tako, da za

$$h_1(\lambda)=\frac{c_1}{d_i-\lambda}+c_1'$$

velja $h_1(x_r)=\psi_1(x_r)$ in $h_1'(x_r)=\psi_1'(x_r)$. Podobno določimo c_2,c_2' tako, da za

$$h_2(\lambda)=\frac{c_2}{d_{i+1}-\lambda}+c_2'$$

velja $h_2(x_r)=\psi_2(x_r)$ in $h_2'(x_r)=\psi_2'(x_r)$. Sedaj je $h(\lambda)=1+h_1(\lambda)+h_2(\lambda)$ iskana racionalna funkcija. Enačba $h(\lambda)=0$ ima dve rešitvi, za x_{r+1} pa vzamemo tisto, ki leži znotraj (d_{i+1},d_i). Komvergenca je zelo hitra, saj h zelo dobro aproksimira f na intervalu (d_{i+1},d_i).

Pri numeričnem računanju se izkaže, da so lastni vektorji, ki jih izračunamo preko leme 23 bolj slabo ortogonalni, zato je potrebno algoritem popraviti. Pri popravku vektor u nadomestimo z bližnjim $\widehat{u}$, za katerega je potem vse v redu, vse pa temelji na Löwnerjevem izreku.

Časovna zahtevnost algoritma za izračun vseh lastnih vrednosti in vektorjev je $\frac{4}{3}n^3+\mathcal{O}(n^2)$, v praksi pa je še manjša, saj lahko velikokrat lastne vektorje in vrednosti izračunamo preko leme 21.

6.6 Jacobijeva metoda

Pri tej metodi matrike ne reduciramo na tridiagonalno obliko. Ideja je, da matriko A z množenji z Givensovimi rotacijami z leve in desne poskusimo spraviti čim bližje diagonalni matriki.

Najprej poiščimo rotacijo $R = \begin{bmatrix} c & -s \\ s & c \end{bmatrix}$, da bo veljalo

$$R^T \begin{bmatrix} a_{pp} & a_{pq} \\ a_{pq} & a_{qq} \end{bmatrix} R = \begin{bmatrix} c & s \\ -s & c \end{bmatrix} \begin{bmatrix} a_{pp} & a_{pq} \\ a_{pq} & a_{qq} \end{bmatrix} \begin{bmatrix} c & -s \\ s & c \end{bmatrix} = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix}. \tag{6.11}$$

Iz zveze $(a_{qq} - a_{pp})sc + a_{pq}(c^2 - s^2)$ dobimo

$$\tau := \frac{\cos 2\varphi}{\sin 2\varphi} = \frac{c^2 - s^2}{2sc} = \frac{a_{pp} - a_{qq}}{2a_{pq}}.$$

Če definiramo $t := \frac{s}{c} = \tan\varphi$, potem velja (formule za dvojni kot tangensa)

$$t^2 + 2\tau t - 1 = 0.$$

Rešitev je

$$t = \frac{\operatorname{sign}(\tau)}{|\tau| + \sqrt{1+\tau^2}}, \quad c = \sqrt{\frac{1}{1+t^2}}, \quad s = ct.$$

Tako smo dobili formule za izračun Jacobijeve rotacije $\operatorname{jac}(A, p, q)$, ki v A uniči element a_{pq}.

Algoritem za eno rotacijo je

$A = \operatorname{jac}(A, p, q)$
  $\tau = \frac{a_{pp} - a_{qq}}{2a_{pq}}$
  $t = \frac{\operatorname{sign}(\tau)}{|\tau| + \sqrt{1+\tau^2}}$
  $c = \sqrt{\frac{1}{1+t^2}}$
  $s = ct$
  $A = R_{pq}(\varphi)^T A R_{pq}(\varphi)$
  $J = JR_{pq}(\varphi)$ (če potrebujemo tudi lastne vektorje)

Po rotaciji (p, q) se v A spremenita vrstici p in q ter stolpca p in q.

Definicija 8

$$\operatorname{off}(A) = \sqrt{\sum_{\substack{j,k=1 \\ j \neq k}}^{n} |a_{jk}|^2}.$$

Lema 24 *Če A' dobimo iz A z Jacobijevo rotacijo $\operatorname{jac}(A, p, q)$, potem velja*

$$\operatorname{off}(A')^2 = \operatorname{off}(A)^2 - 2a_{pq}^2.$$

Dokaz. Iz $A' = R_{pq}^T A R_{pq}$ sledi $\|A'\|_F = \|A\|_F$. Za diagonalne elemente A' velja $a'_{ii} = a_{ii}$ za $i \neq p, q$, za preostala dva elementa pa zaradi (6.11) velja $a'^2_{pp} + a'^2_{qq} = a_{pp}^2 + a_{qq}^2 + 2a_{pq}^2$. Torej mora za izvendiagonalne elemente veljati $\operatorname{off}(A')^2 = \operatorname{off}(A)^2 - 2a_{pq}^2$. ■

Z vsako Jacobijevo rotacijo se tako zmanjša $\operatorname{off}(A)$.

Kako uničujemo elemente:

- *klasična varianta*: v vsakem koraku poiščemo po absolutni vrednosti največji izvendiagonalni element in ga uničimo. Sicer imamo po številu rotacij res najhitrejšo konvergenco, a imamo veliko primerjanj, ki povečajo časovno zahtevnost metode.
- *ciklična varianta*: v vedno enakem vrstnem redu gremo skozi vse elemente. Tu nimamo primerjanj, lahko pa zaradi uničevanja elementov, ki so majhni po absolutni vrednosti porabimo veliko korakov.
- *pragovna varianta*: v vedno enakem vrstnem redu gremo skozi vse elemente, a uničimo le tiste elemente, ki so po absolutni vrednosti čez neko mejo, ki jo zmanjšamo v vsakem sprehodu.

Iteriramo, dokler ni $\mathrm{off}(A) \leq \epsilon$.

Jacobijeva metoda porabi več operacij kot QR ali deli in vladaj, njena prednost pa je, da lastne vrednosti z majhnimi absolutnimi vrednostmi izračuna relativno bolj natančno od ostalih metod.

VII. Računanje singularnega razcepa

7.1 Uvod

Za matriko $A \in \mathbb{R}^{m\times n}$ bi radi izračunali singularni razcep $A = U\Sigma V^T$. Ker velja

$$A^TA = V\Sigma^T\Sigma V^T,$$

dobimo preprosto idejo:

1) izračunamo A^TA,
2) rešimo lastni problem $A^TA = V\Lambda V^T$, odtod dobimo V,
3) za Σ vzamemo $m \times n$ matriko, ki ima v zgornjem bloku kvadratni koren Λ,
4) rešimo sistem $U\Sigma = AV$ za matriko U.

Zgled 28 *Računanje na ta način ni numerično stabilno. Če vzamemo npr.* $A = \begin{bmatrix} 1 & 1 \\ 0 & \sqrt{\eta} \end{bmatrix}$, *kjer je* $\eta = \frac{1}{2}u$, *potem je* $A^TA = \begin{bmatrix} 1 & 1 \\ 1 & 1+\eta \end{bmatrix}$, *v premični piki pa se to zaokroži v* $\begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}$. *Numerično izračunane singularne vrednosti so zato* $\sqrt{2}$ *in* 0, *točne pa so* $\sqrt{2}$ *in* $\sqrt{\eta}$. *Pri majhnih singularnih vrednostih lahko torej pride do velike relativne napake.* ■

Računanje A^TA tako ni najboljša ideja in se mu poskušamo izogniti. Pri računanju SVD za A (razen pri Jacobijevi metodi) matriko najprej reduciramo na bidiagonalno obliko. Postopek je:

1) poišči ortogonalni matriki U_1, V_1, da bo $A = U_1BV_1^T$ in B bidiagonalna,
2) izračunaj SVD za B: $B = U_2\Sigma V_2^T$,
3) SVD razcep za A je $A = (U_1U_2)\Sigma(V_1V_2)^T$.

Redukcijo na bidiagonalno obliko naredimo z množenji s Householderjevimi zrcaljenji z leve in z desne. V primeru matrike 6×4 dobimo:

$$A = \begin{bmatrix} \times & \times & \times & \times \\ \times & \times & \times & \times \\ \times & \times & \times & \times \\ \times & \times & \times & \times \\ \times & \times & \times & \times \\ \times & \times & \times & \times \end{bmatrix}, \quad P_1A = \begin{bmatrix} \times & \times & \times & \times \\ 0 & \times & \times & \times \\ 0 & \times & \times & \times \\ 0 & \times & \times & \times \\ 0 & \times & \times & \times \\ 0 & \times & \times & \times \end{bmatrix},$$

$$P_1AP_1' = \begin{bmatrix} \times & \times & 0 & 0 \\ 0 & \times & \times & \times \\ 0 & \times & \times & \times \\ 0 & \times & \times & \times \\ 0 & \times & \times & \times \\ 0 & \times & \times & \times \end{bmatrix}, \quad P_2P_1AP_1' = \begin{bmatrix} \times & \times & 0 & 0 \\ 0 & \times & \times & \times \\ 0 & 0 & \times & \times \\ 0 & 0 & \times & \times \\ 0 & 0 & \times & \times \\ 0 & 0 & \times & \times \end{bmatrix},$$

$$P_2P_1AP_1'P_2' = \begin{bmatrix} \times & \times & 0 & 0 \\ 0 & \times & \times & 0 \\ 0 & 0 & \times & \times \\ 0 & 0 & \times & \times \\ 0 & 0 & \times & \times \\ 0 & 0 & \times & \times \end{bmatrix}, \quad P_4P_3P_2P_1AP_1'P_2' = \begin{bmatrix} \times & \times & 0 & 0 \\ 0 & 0 & \times & \times \\ 0 & 0 & 0 & \times \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} = B.$$

Nadaljujemo z računanjem singularnega razcepa za B, pri čemer lahko predpostavimo, da je B kvadratna matrika $n \times n$.

Lema 25 *Naj bo B kvadratna matrike $n \times n$ s singularnimi vrednostmi in vektorji $Bv_i = \sigma_i u_i$, $i = 1, \ldots, n$. Lastne vrednosti matrike*

$$C = \begin{bmatrix} 0 & B^T \\ B & 0 \end{bmatrix}$$

so $\pm\sigma_i$, ustrezni lastni vektorji pa $\frac{1}{\sqrt{2}} \begin{bmatrix} v_i \\ \pm u_i \end{bmatrix}$, $i = 1, \ldots, n$.

Dokaz. Naj bo $B = U\Sigma V^T$ singularni razcep za B. Potem velja

$$\begin{bmatrix} 0 & B^T \\ B & 0 \end{bmatrix} \begin{bmatrix} V & V \\ U & -U \end{bmatrix} = \begin{bmatrix} V & V \\ U & -U \end{bmatrix} \begin{bmatrix} \Sigma & 0 \\ 0 & \Sigma \end{bmatrix}. \quad \blacksquare$$

Tako lahko računamo singularni razcep brez računanja B^TB a z dvakrat večjo matriko. Tudi tu pa imamo lahko še vedno težave z majhnimi singularnimi vrednostmi. Poglejmo si nekaj specialnih metod za računanje singularnega razcepa matrike B.

7.2 QR iteracija

Naj bo

$$B = \begin{bmatrix} a_1 & b_1 & & \\ & \ddots & \ddots & \\ & & a_{n-1} & b_{n-1} \\ & & & a_n \end{bmatrix}.$$

Mislimo si, da računamo implicitno QR iteracijo z Wilkinsonovimi premiki za B^TB, a B^TB nikoli ne izračunamo eksplicitno. Za Wilkinsonov premik potrebujemo lastne vrednosti desne spodnje 2×2 podmatrike B^TB

$$\begin{bmatrix} a_{n-1}^2 + b_{n-2}^2 & a_{n-1}b_{n-1} \\ a_{n-1}b_{n-1} & a_n^2 + b_{n-1}^2 \end{bmatrix},$$

za σ^2 pa vzamemo bližnjo lastno vrednost $a_n^2 + b_{n-1}^2$. Nato izračunamo Givensovo rotacijo $R = \begin{bmatrix} c & -s \\ s & c \end{bmatrix}$, da bo (glede na prvi stolpec B^TB)

$$R \begin{bmatrix} a_1^2 - \sigma^2 \\ a_1 b_2 \end{bmatrix} = \begin{bmatrix} \times \\ 0 \end{bmatrix}$$

in vzamemo $R'_{12} = \begin{bmatrix} R & 0 \\ 0 & I \end{bmatrix}$. Ko izračunamo BR'_{12} (to je ekvivalentno $R_{12}^T B^T B R_{12}$ pri QR za B^TB), dobimo (v primeru 5×5)

$$BR'_{12} = \begin{bmatrix} \times & \times & & & \\ + & \times & \times & & \\ & & \times & \times & \\ & & & \times & \times \\ & & & & \times \end{bmatrix}.$$

Dodatni neničelni element + uničimo tako, da ga z množenjem z ustreznimi rotacijami poganjamo navzdol z leve na desno. Tako dobimo

$$R_{12}BR'_{12} = \begin{bmatrix} \times & \times & + & & \\ 0 & \times & \times & & \\ & & \times & \times & \\ & & & \times & \times \\ & & & & \times \end{bmatrix}, \quad R_{12}BR'_{12}R'_{23} = \begin{bmatrix} \times & \times & 0 & & \\ & \times & \times & & \\ & + & \times & \times & \\ & & & \times & \times \\ & & & & \times \end{bmatrix}.$$

Na koncu je $(R_{45}R_{34}R_{23}R_{12})B(R'_{12}B'_{23}B'_{34}B'_{45})$ bidiagonalna matrika za naslednji korak QR metode.

7.3 Jacobijeva metoda

Spet delamo s polno matriko A. Mislimo si, da delamo Jacobijevo metodo za računanje lastnih vrednosti na A^TA, pri čemer spet ne računamo A^TA.

Če delamo jac(A^TA, p, q) to pomeni, da iz A^TA dobimo $R_{pq}^T A^T A R_{pq}$. Zato lahko delamo z A in v enem koraku iz A dobimo AR_{pq}.

Algoritem za enostransko Jacobijevo metodo je:

$A =$ oneside_jac(A, p, q)
  izračunaj $b_{pp} = (A^TA)_{pp}$, $b_{pq} = (A^TA)_{pq}$, $b_{qq} = (A^TA)_{qq}$
  če velja $|b_{pq}| > \epsilon\sqrt{b_{pp}b_{qq}}$, potem
    $\tau = \frac{b_{pp} - b_{qq}}{2b_{pq}}$
    $t = \frac{\text{sign}(\tau)}{|\tau| + \sqrt{1+\tau^2}}$
    $c = \frac{1}{1+t^2}$
    $s = ct$
    $A = AR_{pq}(\varphi)$
    $J = JR_{pq}(\varphi)$ (če potrebujemo tudi singularne vektorje).

Po rotaciji (p, q) se v A spremenita stolpca p in q.

V poštev pride le pragovna varianta, saj bi morali pri klasični izračunati vse elemente $A^T A$, da bi lahko poiskali ustrezno rotacijo. Končamo, ko velja $|b_{pq}| \leq \epsilon\sqrt{b_{pp}b_{qq}}$ za vse $p \neq q$.

Na koncu dobimo:

- $\sigma_i = \|A(:, i)\|_2$, (to je v bistvu $\sqrt{b_{ii}}$),
- $U = [u_1 \ \cdots \ u_n]$, kjer je $u_i = \frac{1}{\sigma_i} A(:, i)$,
- $V = J$ (če smo shranjevali produkte).

VIII. Iterativno reševanje linearnih sistemov

8.1 Uvod

Sistem $Ax = b$ zapišemo v ekvivalentni obliki $x = Rx + c$ in ga rešujemo iterativno

$$x^{(r+1)} = Rx^{(r)} + c.$$

Matriko R imenujemo *iteracijska matrika*. Upamo, da bo pri čim blažjih pogojih zaporedje $\{x^{(r)}\}$ konvergiralo proti rešitvi sistema $Ax = b$. Vedeti moramo:

a) kdaj končamo z iteracijami,

b) pri kakšnih pogojih zaporedje konvergira,

c) kako iz $Ax = b$ pridemo do ekvivalentne oblike $x = Rx + c$.

Najprej si poglejmo dva kriterija, ki ju uporabljamo za konec:

- $\|x^{(r+1)} - x^{(r)}\| \leq \epsilon \|x^{(r)}\|$,
- $\|Ax^{(r+1)} - b\| \leq \epsilon(\|A\|\|x^{(r+1)}\| + \|b\|)$.

Pri drugem kriteriju res testiramo, kako dober približek je $x^{(r+1)}$, a moramo zato množiti z matriko A. Prvi kriterij je cenejši, a je pri počasni konvergenci lahko izpolnjen tudi, ko smo še daleč od prave rešitve.

Izrek 28 *Zaporedje $x^{(r+1)} = Rx^{(r)} + c$, $r = 0, 1, \ldots$, za poljuben $x^{(0)}$ konvergira natanko tedaj, ko velja $\rho(R) < 1$ (za vse lastne vrednosti λ matrike R velja $|\lambda| < 1$).*

Dokaz. Naj bo $\widehat{x}$ točna rešitev. Potem iz $\widehat{x} = R\widehat{x} + c$ in $x^{(r+1)} = Rx^{(r)} + c$ sledi

$$\widehat{x} - x^{(r+1)} = R(\widehat{x} - x^{(r)})$$

in naprej

$$\widehat{x} - x^{(r+1)} = R^2(\widehat{x} - x^{(r-1)}) = \cdots = R^{r+1}(\widehat{x} - x^{(0)}).$$

Očitno je potreben in zadosten pogoj za konvergenco kar $\lim_{k\to\infty} R^k = 0$. R lahko zapišemo v Jordanovi formi v obliki $R = XJX^{-1}$, kjer je $J = \operatorname{diag}(J_1, \ldots, J_k)$. Za vsak Jordanov blok J_i velja, da je

$$J_i^k = \begin{bmatrix} \lambda_i^k & \binom{k}{1}\lambda_i^{k-1} & \binom{k}{2}\lambda_i^{k-1} & \cdots & \\ & \lambda_i^k & \binom{k}{1}\lambda_i^{k-1} & \ddots & \vdots \\ & & \ddots & \ddots & \binom{k}{2}\lambda_i^{k-1} \\ & & & \lambda_i^k & \binom{k}{1}\lambda_i^{k-1} \\ & & & & \lambda_i^k \end{bmatrix},$$

torej je $\lim_{k\to\infty} J_i^k = 0$ natanko tedaj, ko je $|\lambda_i| < 1$. ∎

Posledica 4 *Zadosten pogoj za konvergenco zaporedja $x^{(r+1)} = Rx^{(r)} + c$, $r = 0, 1, \ldots$, za poljuben $x^{(0)}$ je $\|R\| < 1$.* ■

Kako pridemo do R? En način je, da sistem $Ax = b$ zapišemo kot $Mx = -Nx + b$, kjer je $A = M + N$ in dobimo $R := -M^{-1}N$. Sistem seveda rešujemo v obliki

$$Mx^{(r+1)} = -Nx^{(r)} + b, \quad r = 0, 1, \ldots,$$

matriko M pa izberemo tako, da znamo sistem z matriko M rešiti hitreje od polnega sistema.

Iterativne metode pridejo še posebno v poštev, ko imamo velike razpršene sisteme, kjer je veliko elementov enakih 0, neničelni elementi pa nimajo kakšne posebne oblike (npr. pasovne). Pri direktnih metodah (LU, razcep Choleskega, QR) se razpršenost ponavadi izgubi, zato te metode niso primerne.

8.2 Jabobijeva in Gauss-Seidelova metoda

Sistem $Ax = b$ lahko pri pogoju $a_{ii} \neq 0$, $i = 1, \ldots, n$, zapišemo kot

$$\begin{aligned} x_1 &= \frac{1}{a_{11}}(b_1 - a_{12}x_2 - a_{13}x_3 - \cdots - a_{1n}x_n) \\ x_2 &= \frac{1}{a_{22}}(b_2 - a_{21}x_1 - a_{23}x_3 - \cdots - a_{2n}x_n) \\ &\vdots \\ x_n &= \frac{1}{a_{nn}}(b_n - a_{n1}x_1 - a_{n2}x_n - \cdots - a_{n,n-1}x_{n-1}) \end{aligned}$$

Od tod sledi *Jacobijeva metoda*

$$x_k^{(r+1)} = \frac{1}{a_{kk}}\left(b_k - \sum_{i=1,\ i\neq k}^{n} a_{ki}x_i^{(r)}\right), \quad k = 1, \ldots, n.$$

Če zapišemo $A = L + D + U$, kjer je L spodnji trikotnik matrike A (brez diagonale), D diagonala, U pa zgornji trikotnik matrike A, potem je $M = D$ in $N = L + U$. Jacobijeva iteracijska matrika je

$$R_J = -D^{-1}(L + U).$$

Ko po vrsti računamo $x_1^{(r+1)}, \ldots, x_n^{(r+1)}$, bi lahko pri računanju $x_k^{(r+1)}$ uporabili že izračunane vrednosti $x_1^{(r+1)}, \ldots, x_{k-1}^{(r+1)}$. Tako dobimo *Gauss-Seidelovo metodo*

$$x_k^{(r+1)} = \frac{1}{a_{kk}}\left(b_k - \sum_{i=1}^{k-1} a_{ki}x_i^{(r+1)} - \sum_{i=k+1}^{n} a_{ki}x_i^{(r)}\right), \quad k = 1, \ldots, n.$$

Sedaj je $M = L + D$, $N = U$ in $R_{GS} = -(L + D)^{-1}U$.

Pri Jacobijevi metodi lahko vse elemente vektorja $x^{(r+1)}$ računamo hkrati in je zato zelo primerna za paralelizacijo. Pri Gauss-Seidelovi metodi to ni možno.

Izrek 29 *Če je A strogo diagonalno dominantna po vrsticah, kar pomeni*

$$|a_{ii}| > \sum_{j=1}^{n} |a_{ij}|, \quad i = 1, \ldots, n,$$

potem Jacobijeva in Gauss-Seidelova metoda konvergirata.

Dokaz. Pri Jacobijevi metodi očitno iz diagonalne dominantnosti sledi $\|R_j\|_\infty < 1$.

Naj bo (λ, x) lastni par za R_{GS}, torej

$$R_{GS}x = \lambda x$$

in $x \neq 0$. Od tod sledi

$$-\sum_{j=i+1}^{n} a_{ij}x_j = \lambda \sum_{j=1}^{i} a_{ij}x_j$$

in

$$-\lambda a_{ii}x_i = \lambda \sum_{j=1}^{i-1} a_{ij}x_j + \sum_{j=i+1}^{n} a_{ij}x_j.$$

Izberemo indeks k, pri katerem je $|x_k| = \|x\|_\infty$. Sedaj lahko ocenimo

$$\begin{aligned} |\lambda||a_{kk}| &\leq |\lambda| \sum_{j=1}^{k-1} |a_{kj}| \frac{|x_j|}{|x_k|} + \sum_{j=k+1}^{n} |a_{kj}| \frac{|x_j|}{|x_k|} \\ &\leq |\lambda| \sum_{j=1}^{k-1} |a_{kj}| + \sum_{j=k+1}^{n} |a_{kj}|. \end{aligned}$$

Sledi

$$|\lambda| \leq \frac{\sum_{j=k+1}^{n} |a_{jk}|}{|a_{kk}| - \sum_{j=1}^{k-1} |a_{kj}|} < 1,$$

saj je

$$|a_{kk}| > \sum_{j=1}^{k-1} |a_{kj}| + \sum_{j=k+1}^{n} |a_{kj}|. \quad \blacksquare$$

Izrek 30 *Gauss-Seidelova metoda konvergira za hermitsko pozitivno definitno matriko A.*

Dokaz. $A = A^*$, torej $L = U^*$ in $R_{GS} = -(U^* + D)^{-1}U$. Naj bo (λ, x) lastni par za R_{GS}, torej $x \neq 0$ in

$$-Ux = (U^* + D)\lambda x.$$

Enačbo pomnožimo z x^* in dobimo

$$\lambda = -\frac{x^*Ux}{x^*U^*x + x^*Dx}.$$

Sedaj definiramo

$$\sigma := x^*Dx = \sum_{i=1}^{n} a_{ii}|x_i|^2 > 0$$

in

$$\alpha + \beta i := x^* U x,$$

torej $x^*U^*x = \alpha - \beta i$. Dobimo

$$\lambda = -\frac{\alpha + \beta i}{\sigma + \alpha - \beta i}$$

in

$$|\lambda|^2 = \frac{\alpha^2 + \beta^2}{(\sigma + \alpha)^2 + \beta^2}.$$

Ker je A pozitivno definitna je $x^*Ax = x^*U^*x + x^*Dx + x^*Ux = 2\alpha + \sigma > 0$ in

$$(\sigma + \alpha)^2 = \sigma(\sigma + 2\alpha) + \alpha^2 > \alpha^2,$$

to pa pomeni, da je $|\lambda| < 1$. ■

Zgled 29 *Za sistem*

$$\begin{aligned} 12x_1 - 3x_2 + x_3 &= 10 \\ -x_1 + 9x_2 + 2x_3 &= 10 \\ x_1 - x_2 + 10x_3 &= 10 \end{aligned}$$

in začetni približek $x^{(0)} = [1\ 0\ 1]^T$ *izračunaj dva korak po Jacobijevi in Gauss-Seidelovi metodi. Pri Jacobiju dobimo*

$$x^{(1)} = \begin{bmatrix} 0.75 \\ 1 \\ 0.9 \end{bmatrix}, \quad x^{(2)} = \begin{bmatrix} 1.00833.. \\ 0.99444.. \\ 1.025 \end{bmatrix},$$

pri Gauss-Seidelovi metodi pa

$$x^{(1)} = \begin{bmatrix} 0.75 \\ 0.9722.. \\ 1.022.. \end{bmatrix}, \quad x^{(2)} = \begin{bmatrix} 0.991203.. \\ 0.994084.. \\ 1.0002881 \end{bmatrix},$$

Točen rezultat je $\widehat{x} = [1\ 1\ 1]^T$.

8.3 SOR metoda

Pri metodi *SOR* računamo

$$x_k^{(r+1)SOR} = x_k^{(r)} + \omega \left(x_k^{(r+1)GS} - x_k^{(r)} \right),$$

kjer je ω *relaksacijski parameter*, za katerega se izkaže, da mora biti $0 < \omega < 2$. Pri $\omega = 1$ je SOR kar Gauss-Seidelova metoda.

Gre za pospešitev Gauss-Seidelove metode, ideja pa je, da bo, če je zaporedje monotono, za primerni parameter $\omega > 1$ približek $x^{(r+1)SOR}$ bližje rešitvi kot $x^{(r+1)GS}$, če pa zaporedje alternira, bo boljši približek pri $0 < \omega < 1$. Optimalni ω je težko oceniti, dokazali pa bomo, da mora biti $\omega \in (0, 2)$.

V matrični obliki velja

$$x_k^{(r+1)} = x_k^{(r)} + \omega \frac{1}{a_{kk}} \left(b_k - \sum_{i=1}^{k-1} a_{ki} x_i^{(r+1)} - \sum_{i=k}^{n} a_{ki} x_i^{(r)} \right), \quad k = 1, \ldots, n,$$

oziroma

$$x^{(r+1)} = x^{(r)} + \omega D^{-1} \left(b - L x^{(r+1)} - (U + D) x^{(r)} \right)$$

in

$$(\omega L + D) x^{(r+1)} = (D - \omega(D + U))\, x^{(r)} + \omega b,$$

kar pomeni

$$R_{SOR} = (\omega L + D)^{-1}((1 - \omega)D - \omega U).$$

Izrek 31 *SOR ne more konvergirati za poljubni začetni vektor v primeru $\omega \notin (0, 2)$.*

Dokaz. $R_{SOR} = (\omega L + D)^{-1}((1 - \omega)D - \omega U)$. $(\omega L + D)^{-1}$ je spodnje trikotna matrika, ki ima na diagonali elemente a_{ii}^{-1}, $i = 1, \ldots, n$, $(1 - \omega)D - \omega U$ pa je zgornja trikotna matrika, ki ima na diagonali elemente $(1 - \omega)a_{ii}$, $i = 1, \ldots, n$. Od tod sledi

$$\det R_{SOR} = (1 - \omega)^n.$$

Ker je determinanta enaka produktu lastnih vrednosti, velja $\rho(R_{SOR}) \geq |1 - \omega|$ in potreben pogoj za konvergenco je $\omega \in (0, 2)$. ■

Izrek 32 *Če je A hermitska pozitivno definitna matrika, potem SOR v primeru $\omega \in (0, 2)$ konvergira za poljubni začetni vektor.*

Dokaz. Podoben dokazu za Gauss-Seidelovo metodo. ■

Definicija 9 *Matrika $A = L + D + U$ je konstantno urejena, če so lastne vrednosti matrike*

$$C(\alpha) = -D^{-1}(\frac{1}{\alpha} L + \alpha U)$$

neodvisne od α.

Definicija 10 *Matrika A ima lastnost A, če obstaja taka permutacijska matrika P, da je*

$$PAP^T = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix},$$

kjer sta A_{11} in A_{22} diagonalni matriki.

Lastnost A pomeni tudi, da je matrika konsistentno urejena.

Zgled 30 *Matrika oblike*

$$A = \begin{bmatrix} T & -I & & \\ -I & T & \ddots & \\ & \ddots & \ddots & -I \\ & & -I & T \end{bmatrix},$$

kjer je

$$T = \begin{bmatrix} 4 & -1 & & \\ -1 & 4 & \ddots & \\ & \ddots & \ddots & -1 \\ & & -1 & 4 \end{bmatrix},$$

ima lastnost A. Takšno matriko srečamo pri reševanju Poissonove enačbe. ■

Izrek 33 *Če je A konsistentno urejena in* $\mu = \rho(R_J)$*, potem velja:*

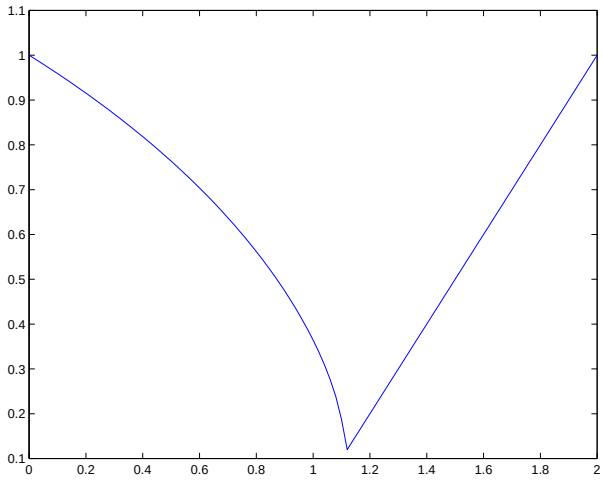
1) $\rho(R_{GS}) = \mu^2$,

2) $\omega_{\text{opt}} = \dfrac{2}{1+\sqrt{1-\mu^2}}, \quad \rho(R_{SOR}(\omega_{\text{opt}})) = \omega_{\text{opt}} - 1,$

3)

$$\rho(R_{SOR}(\omega)) = \begin{cases} \omega - 1, & \text{za } \omega_{\text{opt}} \le \omega < 2 \\ 1 - \omega + \frac{1}{2}\omega^2\mu^2 + \omega\mu\sqrt{1-\omega+\frac{1}{4}\omega^2\mu^2}, & \text{za } 0 < \omega \le \omega_{\text{opt}}. \end{cases}$$ ■

Zgled 31 *Za matriko* $A = \begin{bmatrix} 4 & -1 & & -1 & & \\ -1 & 4 & -1 & & -1 & \\ & -1 & 4 & & & -1 \\ -1 & & & 4 & -1 & \\ & -1 & & -1 & 4 & -1 \\ & & -1 & & -1 & 4 \end{bmatrix}$ *velja* $\rho(R_J) = 0.6036$, $\rho(R_{GS}) = 0.3634$, $\omega_{\text{opt}} = 1.1128$, *graf* $\rho(R_{SOR}(\omega))$ *pa je*



IX. Polinomska interpolacija

9.1 Uvod

Podane imamo vrednosti funkcije f v $n+1$ različnih točkah $x_0, \ldots, x_n$, iščemo pa polinom stopnje n ali manj, ki se v točkah x_i ujema s funkcijo f. Tak polinom imenujemo *interpolacijski polinom*.

Zakaj potrebujemo interpolacijo:

- včasih je bila glavna uporaba računanje vrednosti tabelirane funkcije,
- polinomska interpolacija je osnova za numerično odvajanje in integriranje.

Zakaj polinomi:

- namesto polinomov lahko uporabljamo tudi trigonometrične polinome ali racionalne funkcije,
- najbolj ustrezajo zahtevam za interpolacijsko funkcijo:
  - preprosta konstrukcija in računanje vrednosti,
  - preprosto odvajanje in integriranje.

Skoraj nikoli ne iščemo enačbe polinoma v standardni bazi. Pomembno je le, da znamo izračunati vrednost polinoma v dani točki.

9.2 Interpolacijski polinom

Denimo, da imamo paroma različne točke $x_0, \ldots, x_n$ in vrednosti $y_0, \ldots, y_n$. Iščemo polinom I_n stopnje n ali manj, za katerega velja $I_n(x_i) = y_i$, $i = 0, \ldots, n$.

Izrek 34 *Za paroma različne točke $x_0, \ldots, x_n$ in vrednosti $y_0, \ldots, y_n$ obstaja natanko en polinom I_n stopnje n ali manj, za katerega velja $I_n(x_i) = y_i$, $i = 0, \ldots, n$.*

Dokaz. Eksistenco pokažemo s konstrukcijo. Definirajmo polinome

$$L_{n,i}(x) = \frac{(x-x_0)\cdots(x-x_{i-1})(x-x_{i+1})\cdots(x-x_n)}{(x_i-x_0)\cdots(x_i-x_{i-1})(x_i-x_{i+1})\cdots(x_i-x_n)} = \prod_{\substack{k=0 \\ k\neq i}}^{n} \frac{x-x_k}{x_i-x_k},$$

ki so stopnje n, imenujemo pa jih *Lagrangevi koeficienti*. Očitno velja $L_{n,i}(x_j) = \delta_{ij}$. Če sedaj definiramo

$$I_n(x) = \sum_{k=0}^{n} y_k L_{n,k}(x),$$

je to očitno polinom stopnje n ali manj, prav tako pa velja $I_n(x_i) = y_i$, $i = 0, \ldots, n$.

Enoličnost sledi iz tega, da se dva različna polinoma stopnje n ali manj ne moreta ujemati v več kot n različnih točkah. Če bi se namreč ujemala, bi imela njuna razlika več kot n ničel. Ker pa je razlika spet polinom stopnje največ n, je to možno le, če je ta polinom kar enak 0. ■

Če vpeljemo polinom $\omega(x) = (x-x_0)(x-x_1)\cdots(x-x_n)$, potem velja

$$L_{n,i}(x) = \frac{\omega(x)}{(x-x_i)\omega'(x_i)}.$$

I_n se s funkcijo f ujema v točkah $x_0, \ldots, x_n$, drugje pa ne nujno. Lahko pa ocenimo razliko.

Izrek 35 *Če je f $(n+1)$-krat zvezno odvedljiva funkcija na $[a,b]$, ki vsebuje vse točke $x_0, \ldots, x_n$, potem za vsak $x \in [a,b]$ obstaja tak $\xi \in (a,b)$, da velja*

$$f(x) - I_n(x) = \frac{f^{(n+1)}(\xi)}{(n+1)!}\omega(x) \tag{9.12}$$

in $\min(x, x_0, \ldots, x_n) < \xi < \max(x, x_0, \ldots, x_n)$.

Dokaz. Za $x = x_i$ nimamo kaj dokazovati, saj je na obeh straneh 0. Za ostale pa definiramo funkcijo

$$F(z) := f(z) - I_n(z) - R \cdot \omega(z). \tag{9.13}$$

F je očitno $(n+1)$-krat zvezno odvedljiva in velja $F(x_i) = 0$, $i = 0, \ldots, n$. Za poljubno točko $x \in [a,b]$, ki je različna od $x_0, \ldots, x_n$ lahko konstanto R določimo tako, da bo $F(x) = 0$.

Pri tako izbranem R ima F vsaj $n+2$ različnih ničel na $[a,b]$. Po Rollovem izreku ima zato F' vsaj $n+1$ različnih ničel na (a,b), F'' vsaj n ničel, ..., in končno, $F^{(n+1)}$ ima vsaj eno ničlo na (a,b), ki jo označimo s ξ.

Iz enačbe (9.13) sledi $0 = F^{(n+1)}(\xi) = f^{(n+1)}(\xi) - R(n+1)!$, torej

$$R = \frac{f^{(n+1)}(\xi)}{(n+1)!}$$

in za izbrani x smo dokazali (9.12). ■

Lagrangeva oblika ni najprimernejša za konstrukcijo in računanje vrednosti interpolacijskega polinoma:

- veliko operacij,
- stopnjo moramo vnaprej predpisati.

Boljše so npr. zaporedne linearne interpolacije. Označimo z $I_{i,i+1,\ldots,i+k}$ interpolacijski polinom, ki se z f ujema v točkah $x_i, x_{i+1}, \ldots, x_{i+k}$. Hitro se lahko prepričamo, da velja

$$I_{i,i+1,\ldots,i+k}(x) = \frac{I_{i,i+1,\ldots,i+k-1}(x)(x - x_{i+k}) - I_{i+1,i+2,\ldots,i+k}(x)(x - x_i)}{x_i - x_{i+k}}$$

oziroma

$$I_{i,i+1,\ldots,i+k}(x) = \frac{1}{x_{i+k} - x_i} \begin{vmatrix} x - x_i & I_{i,i+1,\ldots,i+k-1}(x) \\ x - x_{i+k} & I_{i+1,i+2,\ldots,i+k}(x) \end{vmatrix}.$$

Z zaporednimi linearnimi interpolacijami lahko povečujemo stopnjo interpolacijskega polinoma. Obstaja več postopkov, pogledali pa si bomo *Nevilleovo shemo*, ki se jo da preprosto sprogramirati v računalniku. Predstavimo jo s trikotno shemo:

x_i	$x - x_i$	y_i	$I_{..}(x)$	$I_{...}(x)$	$I_{....}(x)$
x_0	$x - x_0$	y_0			
			$I_{01}(x)$		
x_1	$x - x_1$	y_1		$I_{012}(x)$	
			$I_{12}(x)$		$I_{0123}(x)$
x_2	$x - x_2$	y_2		$I_{123}(x)$	
			$I_{23}(x)$		
x_3	$x - x_3$	y_3			

Zgled 32 *Poišči vrednost interpolacijskega polinoma za podatke $x : 0, 2, 4$ in $f(x) : 2, 4, 8$ v točki $x = 1$.*

Uporabili bomo Nevillovo shemo:

x_i	$x - x_i$	y_i	$I_{..}(x)$	$I_{...}(x)$
$x_0 = 0$	$x - x_0 = 1$	$y_0 = 2$		
			$I_{01}(x) = 3$	
$x_1 = 2$	$x - x_1 = -1$	$y_1 = 4$		$I_{012}(x) = \frac{11}{4}$
			$I_{12}(x) = 2$	
$x_2 = 4$	$x - x_2 = -3$	$y_2 = 9$		

Do vrednosti smo prišli z naslednjimi računi:

$$I_{01}(x) = \frac{1}{x_1-x_0}\begin{vmatrix} x-x_0 & I_0(x) \\ x-x_1 & I_1(x) \end{vmatrix} = \frac{1}{2}\begin{vmatrix} 1 & 2 \\ -1 & 4 \end{vmatrix} = 3,$$

$$I_{12}(x) = \frac{1}{x_2-x_1}\begin{vmatrix} x-x_1 & I_1(x) \\ x-x_2 & I_2(x) \end{vmatrix} = \frac{1}{2}\begin{vmatrix} -1 & 4 \\ -3 & 8 \end{vmatrix} = 2,$$

$$I_{012}(x) = \frac{1}{x_2-x_0}\begin{vmatrix} x-x_0 & I_{01}(x) \\ x-x_2 & I_{12}(x) \end{vmatrix} = \frac{1}{4}\begin{vmatrix} 1 & 3 \\ -3 & 2 \end{vmatrix} = \frac{11}{4}. \quad \blacksquare$$

9.3 Deljene diference

Namesto $I_n(x) = a_0 + a_1x + \ldots + a_nx^n$ lahko interpolacijski polinom zapišemo v obliki

$$I_n(x) = c_0 + c_1(x-x_0) + c_2(x-x_0)(x-x_1) + \cdots + c_n(x-x_0)\cdots(x-x_{n-1}).$$

Definicija 11 *Deljena diferenca $f[x_0, x_1, \ldots, x_k]$ je vodilni koeficient interpolacijskega polinoma stopnje k, ki se ujema z f v paroma različnih točkah $x_0, x_1, \ldots, x_k$.*

Izrek 36 *Za deljene diference velja:*

a)

$$P_n(x) = f[x_0] + f[x_0,x_1](x-x_0) + \cdots + f[x_0,x_1,\ldots,x_n](x-x_0)\cdots(x-x_{n-1}) \quad (9.14)$$

je interpolacijski polinom, ki se v točkah $x_0, \ldots, x_n$ ujema z f.

b) $f[x_0, x_1, \ldots, x_k]$ je simetrična funkcija svojih argumentov.

c) $(\alpha f + \beta g)[x_0, \ldots, x_k] = \alpha f[x_0, \ldots, x_k] + \beta g[x_0, \ldots, x_k]$.

d) Velja rekurzivna formula

$$f[x_0, \ldots, x_k] = \frac{f[x_1, \ldots, x_k] - f[x_0, \ldots, x_{k-1}]}{x_k - x_0}. \quad (9.15)$$

Dokaz.

a) Uporabimo indukcijo. Očitno se $f[x_0]$ ujema z $f(x_0)$. Naj bo

$$P_i(x) = f[x_0] + f[x_0,x_1](x-x_0) + \cdots + f[x_0,x_1,\ldots,x_i](x-x_0)\cdots(x-x_{i-1}).$$

Polinom, ki interpolira f v $x_0, \ldots, x_{i+1}$ lahko zapišemo v obliki

$$P_{i+1} = P_i(x) + c(x-x_0)\cdots(x-x_i).$$

Po definiciji deljene diference se mora c ujemati z $f[x_0, \ldots, x_{i+1}]$.

b), c) Očitno.

d) Naj bo p_0 interpolacijski polinom, ki se ujema v točkah $x_0, \ldots, x_{k-1}$ in p_1 polinom, ki se ujema v točkah $x_1, \ldots, x_k$. Interpolacijski polinom p, ki se ujema v vseh točkah, ima obliko

$$p(x) = \frac{x - x_k}{x_0 - x_k} p_0(x) + \frac{x - x_0}{x_k - x_0} p_1(x).$$

Če primerjamo vodilne koeficiente, dobimo zvezo (9.15). ■

Obliko (9.14) imenujemo *Newtonov interpolacijski polinom.*

Nekaj preprostih formul:

- $f[x_0] = f(x_0)$,
- $f[x_0, x_1] = \frac{f(x_1) - f(x_0)}{x_1 - x_0}$.

Kaj se zgodi, ko v limiti pošljemo $x_1 \to x_0$? Iz

$$p(x) = f[x_0] + f[x_0, x_1](x - x_0) = f(x_0) + \frac{f(x_1) - f(x_0)}{x_1 - x_0}(x - x_0)$$

dobimo $p(x) = f(x_0) + f'(x_0)(x - x_0)$.

To pomeni, da lahko definicijo interpolacijskega polinoma in deljenih diferenc razširimo na polinome, ki se poleg vrednosti ujemajo tudi v odvodih. Če se točke ujemajo, potem to pomeni, da naj bo ujemanje tudi v odvodih. Tako npr. pri točkah $x_1, x_2, x_2, x_2, x_3, x_3$ iščemo polinom p za katerega velja $p(x_1) = f(x_1)$, $p(x_2) = f(x_2)$, $p'(x_2) = f'(x_2)$, $p''(x_2) = f''(x_2)$, $p(x_3) = f(x_3)$, $p'(x_3) = f'(x_3)$.

Definicija 12 *Če dopuščamo tudi ujemanje točk, je deljena diferenca definirana kot*

$$f[x_0, x_1, \ldots, x_k] = \begin{cases} \frac{f^k(x_0)}{k!} & x_0 = x_1 = \cdots = x_k \\ \frac{f[x_1, \ldots, x_k] - f[x_0, \ldots, x_{k-1}]}{x_k - x_0} & \text{sicer} \end{cases}.$$

Tudi deljene diference računamo v trikotni shemi iz katere na koncu hitro preberemo enačbo polinoma oz. izračunamo njegovo vrednost v iskani točki.

Zgled 33 *Zapiši enačbo polinoma stopnje 5, za katerega velja:* $p(0) = 1$, $p'(0) = 2$, $p''(0) = 3$, $p(1) = -1$, $p'(1) = 3$, $p(2) = 4$.

x_i	$f[.]$	$f[.,.]$	$f[.,.,.]$	$f[.,.,.,.]$	$f[.,.,.,.,.]$	$f[.,.,.,.,.,.]$
0	1					
		2				
0	1		$\frac{3}{2}$			
		2		$-\frac{11}{2}$		
0	1		-4		$\frac{29}{2}$	
		-2		9		$-\frac{79}{8}$
1	-1		5		$-\frac{21}{4}$	
		3		$-\frac{3}{2}$		
1	-1		2			
		5				
2	4					

$$p(x) = 1 + 2x + \frac{3}{2}x^2 - \frac{11}{2}x^3 + \frac{29}{2}x^3(x-1) - \frac{79}{8}x^3(x-1)^2. \quad \blacksquare$$

Izrek 37 (Hermite-Gennochijeva formula) *Za k-krat zvezno odvedljivo funkcijo f velja*

$$f[x_0, \ldots, x_k] = \int_0^1 dt_1 \int_0^{t_1} dt_2 \int \cdots \int_0^{t_{k-1}} f^{(k)}\Big(t_k(x_k - x_{k-1}) + \cdots + t_1(x_1 - x_0) + x_0\Big) dt_k.$$

Dokaz. Uporabimo indukcijo. Če je $x_0 \neq x_1$, dobimo

$$\begin{aligned} f[x_0, x_1] &= \int_0^1 f'(t_1(x_1 - x_0) + x_0) dt_1 = \frac{1}{x_1 - x_0} f'(t_1(x_1 - x_0) + x_0)\Big|_{t_1=0}^{t_1=1} = \\ &= \frac{f(x_1) - f(x_0)}{x_1 - x_0}, \end{aligned}$$

sicer pa

$$f[x_0, x_0] = \int_0^1 f'(x_0) dt_1 = f'(x_0).$$

Sedaj predpostavimo, da izrek velja za $1, \ldots, k-1$. Če definiramo $\xi = t_k(x_k - x_{k-1}) + \cdots + t_1(x_1 - x_0) + x_0$, lahko zapišemo integral kot

$$f[x_0, x_1, \ldots, x_k] = \int_0^1 dt_1 \int_0^{t_1} dt_2 \int \cdots \int_0^{t_{k-1}} f^{(k)}(\xi) dt_k.$$

. Sedaj s substitucijo spremenljivk dobimo:

$$d\xi = (x_k - x_0) dt_k,$$

$$\begin{aligned} t_k = 0 &\implies \xi_0 = t_{k-1}(x_{k-1} - x_{k-2}) + t_{k-2}(x_{k-2} - x_{k-3}) + \cdots + t_1(x_1 - x_0) + x_0 \\ t_k = t_{k-1} &\implies \xi_1 = t_{k-1}(x_k - x_{k-2}) + t_{k-2}(x_{k-2} - x_{k-3}) + \cdots + t_1(x_1 - x_0) + x_0 \end{aligned}$$

in

$$\int_0^{t_{k-1}} f^{(k)}(\xi) dt_k = \frac{1}{x_k - x_{k-1}} \int_{\xi_0}^{\xi_1} f^{(k)}(\xi) d\xi = \frac{f^{(k-1)}(\xi_1) - f^{(k-1)}(\xi_0)}{x_k - x_{k-1}}.$$

Po indukcijski predpostavki velja

$$\int_0^1 dt_1 \int_0^{t_1} dt_2 \int \cdots \int_0^{t_{k-2}} f^{(k-1)}(\xi_1) dt_{k-1} = f[x_0, x_1, \ldots, x_{k-2}, x_k]$$

in

$$\int_0^1 dt_1 \int_0^{t_1} dt_2 \int \cdots \int_0^{t_{k-2}} f^{(k-1)}(\xi_0) dt_{k-1} = f[x_0, x_1, \ldots, x_{k-2}, x_{k-1}],$$

ker pa vemo

$$f[x_0, x_1, \ldots, x_k] = \frac{f[x_0, x_1, \ldots, x_{k-2}, x_k] - f[x_0, x_1, \ldots, x_{k-2}, x_{k-1}]}{x_k - x_{k-1}},$$

je dokaz končan. $\blacksquare$

Posledica 5 *Za k-krat zvezno odvedljivo funkcijo f velja*

$$f[x_0, \ldots, x_k] = \frac{1}{k!} f^{(k)}(\xi),$$

kjer je

$$\min_{i=0,\ldots,k}(x_i) < \xi < \max_{i=0,\ldots,k}(x_i). \quad \blacksquare$$

Izrek 38 *Za $(n+1)$-krat zvezno odvedljivo funkcijo f in interpolacijski polinom I_n na točkah $x_0, \ldots, x_n$ velja*

$$f(x) = I_n(x) + f[x_0, \ldots, x_n, x](x - x_0) \cdots (x - x_n). \tag{9.16}$$

Dokaz. Naj bo q polinom, ki interpolira f v točkah $x_0, \ldots, x_n, t$. Veljati mora

$$q(x) = I_n(x) + f[x_0, \ldots, x_n, x](x - x_0) \cdots (x - x_n).$$

Ker to velja za vsak t, formula (9.16) drži. $\blacksquare$

Posledica 6 *Za $(n+1)$-krat zvezno odvedljivo funkcijo f in interpolacijski polinom I_n na točkah $x_0, \ldots, x_n$ velja*

$$f(x) - I_n(x) = \frac{f^{(n+1)}(\xi)}{(n+1)!}\omega(x),$$

kjer je $\min(x, x_0, \ldots, x_n) < \xi < \max(x, x_0, \ldots, x_n)$. $\blacksquare$

Prednost nove ocene je, da pride v poštev tudi, če vse točke niso medsebojno različne.

9.4 Končne diference

Če so vse točke *ekvidistantne*, kar pomeni $x_i = x_0 + ih$, kjer je h razmik med sosednjima točkama, lahko formule še poenostavimo. Naj bodo $y_i = f(x_i)$ vrednosti.

Definicija 13 *Končne diference so definirane rekurzivno kot*

$$\Delta^m y_k = \begin{cases} y_k, & m = 0 \\ \Delta^{m-1} y_{k+1} - \Delta^{m-1} y_k, & m > 0 \end{cases}.$$

Iz vrednosti y_i sestavimo *diferenčno tabelo*:

x_i	y_i	Δ	Δ^2	Δ^3
x_0	y_0			
		Δy_0		
x_1	y_1		$\Delta^2 y_0$	
		Δy_1		$\Delta^3 y_0$
x_2	y_2		$\Delta^2 y_1$	
		Δy_2		
x_3	y_3			

Za prve tri diference dobimo:

$$\begin{aligned}\Delta y_0 &= y_1 - y_0,\\ \Delta^2 y_0 &= \Delta y_1 - \Delta y_0 = (y_2 - y_1) - (y_1 - y_0) = y_2 - 2y_1 + y_0,\\ \Delta^3 y_0 &= \Delta^2 y_1 - \Delta^2 y_0 = (y_3 - 2y_2 + y_1) - (y_2 - 2y_1 + y_0) = y_3 - 3y_2 + 3y_1 - y_0.\end{aligned}$$

Obratno velja:

$$\begin{aligned}y_1 &= y_0 + \Delta y_0 = (I + \Delta) y_0\\ y_2 &= y_1 + \Delta y_1 = (I + \Delta) y_1 = (I + \Delta)^2 y_0\end{aligned}$$

Lema 26 *Veljata formuli:*

a)

$$\Delta^m y_0 = \sum_{k=0}^{m} (-1)^k \binom{m}{k} y_{m-k},$$

b)

$$y_m = (I + \Delta)^m y_0 = \sum_{k=0}^{m} \binom{m}{k} \Delta^k y_0. \quad \blacksquare$$

Lema 27 *Če je p polinom stopnje n z vodilnim koeficientom a_0, potem velja $\Delta^n p(x) = n! h^n a_0$ in $\Delta^m p(x) = 0$ za $m > n$.*

Dokaz.

$$\Delta p(x) = (a_0(x+h)^n + a_1(x+h)^{n-1} + \cdots + a_n) - (a_0 x^n + a_1 x^{n-1} + \cdots + a_n) = nh a_0 x^{n-1} + \cdots.$$

Pri vsaki diferenci se stopnja zmanjša za ena, vodilni koeficient pa pomnoži s stopnjo n in s h. $\blacksquare$

Ideja je, da za $x = x_0 + sh$ zapišemo

$$I_n(x_0 + sh) = (I + \Delta)^s y_0 = \sum_{k=0}^{\infty} \binom{s}{k} \Delta^k y_0.$$

Ker interpoliramo s polinomom, odpadejo vsi členi vsote kjer je $k > n$.

Lema 28

$$I_n(x_0 + sh) = (I + \Delta)^s y_0 = \sum_{k=0}^{\infty} \binom{s}{k} \Delta^k y_0 \tag{9.17}$$

je interpolacijski polinom stopnje kvečjemu n za točke $x_i = x_0 + ih$, $i = 0, \ldots, n$, in vrednosti $y_0, \ldots, y_n$.

Dokaz. Očitno je I_n stopnje kvečjemu n in očitno velja $I_n(x_i) = y_i$ za $i = 0, \ldots, n$. ■

Formulo (9.17) imenujemo *prema Newtonova interpolacijska formula.*

Zgled 34 *Sestavi diferenčno tabelo za podatke do tretje diference in izračunaj* $I_3(0.02)$.

x	0.0	0.1	0.2	0.3	0.4
y	0.00	0.11	0.28	0.57	1.04

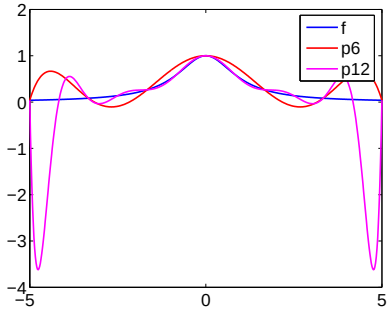
x_i	y_i	Δ	Δ^2	Δ^3
0.0	0.00			
		0.11		
0.1	0.11		0.06	
		0.17		0.06
0.2	0.28		0.12	
		0.29		0.06
0.3	0.57		0.18	
		047		
0.4	1.04			

$$h = 0.1, \quad s = \frac{x - x_0}{h} = 0.2$$

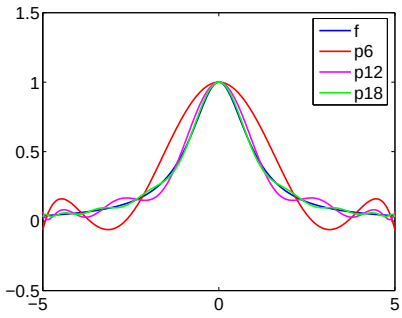
$$I_3(0.02) = 0.00 + 0.2 \cdot 0.11 + \frac{0.2 \cdot (-0.8)}{2} 0.06 + \frac{0.2 \cdot (-0.8) \cdot (-1.8)}{6} 0.06 = 0.02008. \quad ■$$

9.5 Rungejev protiprimer

Denimo, da interpoliramo $f(x) = \frac{1}{1+x^2}$ na $[-5, 5]$. Če uporabimo ekvidistantne točke, potem z večanjem stopnje interpolacijski polinom vedno slabše aproksimira f, kot kaže naslednja slika



Če namesto ekvidistantnih točk vzamemo točke Čebiševa, ki so v tem primeru definirane kot $x_i = 5\cos\left(\frac{i\pi}{n}\right)$, $i = 0, \ldots, n$, potem sedaj polinom dobro aproksimira f.



X. Numerično odvajanje

10.1 Uvod

Iščemo odvod funkcije, ki je podana s tabelo vrednosti v točkah $x_0, \ldots, x_n$. Ideja je, da za približek vzamemo odvod interpolacijskega polinoma.

Vemo, da je $f(x) = I_n(x) + \frac{\omega(x)}{(n+1)!} f^{(n+1)}(\xi)$, kjer je $\omega(x) = (x-x_0)\cdots(x-x_n)$. Z odvajanjem dobimo

$$f'(x) = I_n'(x) + \underbrace{\frac{\omega'(x)}{(n+1)!} f^{(n+1)}(\xi) + \frac{\omega(x)}{(n+1)!} \cdot \frac{df^{(n+1)}(\xi)}{dx}}_{\text{napaka}}.$$

Izraz za napako ni najlepši, saj ne poznamo odvisnosti ξ od x. Če pa računamo odvod v eni izmed točk $x_0, \ldots, x_n$, zadnji člen odpade in dobimo

$$f'(x_k) = I_n'(x_k) + \frac{\omega'(x_k)}{(n+1)!} f^{(n+1)}(\xi). \tag{10.18}$$

Tu je $I_n'(x_k)$ odvod interpolacijskega polinoma v točki x_k.

Formulo za $I_n'(x_k)$ izpeljemo preko Lagrangevih koeficientov. Iz $I_n(x) = \sum_{i=0}^n f(x_i) L_{n,i}(x)$ sledi $I_n'(x_k) = \sum_{i=0}^n f(x_i) L_{n,i}'(x_k)$. Z odvajanjem

$$L_{n,i}(x) = \frac{(x-x_0)\cdots(x-x_{i-1})(x-x_{i+1})\cdots(x-x_n)}{(x_i-x_0)\cdots(x_i-x_{i-1})(x_i-x_{i+1})\cdots(x_i-x_n)}$$

dobimo

a) $i \neq k$: $\quad L_{n,i}'(x_k) = \dfrac{\omega'(x_k)}{(x_k-x_i)\omega'(x_i)}$,

b) $i = k$: $\quad L_{n,k}'(x_k) = \displaystyle\sum_{\substack{j=0 \\ j\neq k}}^{n} \frac{1}{x_k - x_j}$.

S pomočjo teh formul lahko zapišemo

$$I_n'(x_k) = \omega'(x_k) \sum_{\substack{j=0 \\ j\neq k}}^{n} \frac{f(x_j)}{(x_k-x_j)\omega'(x_j)} + f(x_k) \sum_{\substack{j=0 \\ j\neq k}}^{n} \frac{1}{x_k - x_j}.$$

V primeru, ko so točke ekvidistantne in velja $x_i = x_0 + ih$, $i = 0, 1, \ldots, n$, se formula (10.18) še poenostavi. Izpeljavo pustimo za domačo nalogo, dobimo pa

$$f'(x_k) = \frac{1}{h}\left(\frac{(-1)^k}{\binom{n}{k}} \sum_{\substack{j=0 \\ j\neq k}}^{n} \frac{(-1)^j \binom{n}{j} f(x_j)}{k-j} + f(x_k) \sum_{\substack{j=0 \\ j\neq k}}^{n} \frac{1}{k-j} \right) + \frac{(-1)^{n-k} h^n}{(n+1)\binom{n}{k}} f^{(n+1)}(\xi).$$

Nekaj prvih formul, ki jih dobimo z vstavljanjem n in k:

- $n = 1$:

$$\begin{aligned} f'(x_0) &= \frac{1}{h}(f(x_1) - f(x_0)) - \frac{1}{2} h f''(\xi_0) \\ f'(x_1) &= \frac{1}{h}(f(x_1) - f(x_0)) + \frac{1}{2} h f''(\xi_1) \end{aligned}$$

- $n = 2$:

$$\begin{aligned} f'(x_0) &= \frac{1}{2h}(-3f(x_0) + 4f(x_1) - f(x_2)) + \frac{1}{3} h^2 f'''(\xi_0) \\ f'(x_1) &= \frac{1}{2h}(-f(x_0) + f(x_2)) - \frac{1}{6} h^2 f'''(\xi_1) \quad \text{(sim. diferenca)} \\ f'(x_2) &= \frac{1}{2h}(f(x_0) - 4f(x_1) + 3f(x_2)) + \frac{1}{3} h^2 f'''(\xi_2) \end{aligned}$$

Pri simetrični diferenci zaradi simetrije pridobimo red h^2 namesto h, trik pa je, da interpoliramo na treh točkah namesto na dveh.

10.2 Drugi načini izpeljave

Formule za numerično odvajanje lahko izpeljujemo tudi iz razvoja v Taylorjevo vrsto. Naj bodo točke ekvidistantne in $y_i = f(x_i)$. Iz razvojev

$$\begin{aligned} y_0 &= y_1 - h y_1' + \frac{1}{2} h^2 y_1'' - \frac{1}{6} h^3 y_1''' + \frac{1}{24} h^4 f^{(4)}(\xi_0) \\ y_1 &= y_1 \\ y_2 &= y_1 + h y_1' + \frac{1}{2} h^2 y_1'' + \frac{1}{6} h^3 y_1''' + \frac{1}{24} h^4 f^{(4)}(\xi_2) \end{aligned}$$

s seštevanjem dobimo formulo $f''(x_1) = \frac{1}{h^2}(y_0 - 2y_1 + y_2) - \frac{1}{24} h^2 (f^{(4)}(\xi_0) + f^{(4)}(\xi_2))$. Namesto $f^{(4)}(\xi_0) + f^{(4)}(\xi_2)$ lahko pišemo $2f^{(4)}(\xi)$ in dobimo končno formulo

$$f''(x_1) = \frac{1}{h^2}(y_0 - 2y_1 + y_2) - \frac{1}{12} h^2 f^{(4)}(\xi).$$

Alternativni način izpeljave je metoda nedoločenih koeficientov. Pri tej metodi nastavimo sistem $f'(x_k) = \sum_{j=0}^{n} \alpha_j f(x_j) + R(f)$ in določimo koeficiente α_j tako, da je formula točna za polinome čim višje stopnje. Napako dobimo iz polinoma višje stopnje.

Zgled 35 *Izpelji formulo* $f''(x_1) = af(x_0) + bf(x_1) + cf(x_2) + R(f)$.

Za bazo izberemo $1, x - x_1, (x - x_1)^2, \ldots$, *saj tako dobimo lepši sistem:*

$$\left.\begin{array}{rcccl} 1 & : & 0 & = & a+b+c \\ x-x_2 & : & 0 & = & -ha+hc \\ (x-x_2)^2 & : & 2 & = & h^2a+h^2c \end{array}\right\} \Rightarrow a = \frac{1}{h^2},\ b = \frac{-2}{h^2},\ c = \frac{1}{h^2}.$$

Napaka ima obliko $R(f) = Ch^p f^{(r)}(\xi)$, *kjer je* C *konstanta,* p *in* r *pa sta stopnji, ki ju moramo določiti. Odvod* r *ustreza najnižji stopnji polinoma, za katerega formula ni točna.*

Napako dobimo tako, da po vrsti vstavljamo $f(x) = x^r$ *in poiščemo prvo stopnjo, kjer formula ni točna. Ker imamo v formuli 3 točke, je formula točna za polinome stopnje 2 ali manj, zato najprej vstavimo* $r = 3$, *ker pa je formula točna, nadaljujemo z* $r = 4$.

$$\left.\begin{array}{rcccl} (x-x_2)^3 & : & 0 & = & -h^3a+h^3c \\ (x-x_2)^4 & : & 0 & \neq & h^4a+h^4c \end{array}\right\} \Rightarrow r = 4,\ p = 2.$$

Ker za $f(x) = (x - x_2)^4$ *velja* $f^{(4)}(\xi) = 4!$, *dobimo napako* $R(f) = -\frac{1}{12}h^2 f^{(4)}(\xi)$.

Pravilen način za ugotavljanje napake je Peanov izrek, ki ga bomo spoznali pozneje. ■

10.3 Celotna napaka

Zgled 36 *Če po formulah*

$$\begin{aligned} f'(0) &= \frac{1}{h}(f(h) - f(-h)) + \mathcal{O}(h^2) \\ f''(0) &= \frac{1}{h^2}(f(-h) - 2f(0) + f(h)) + \mathcal{O}(h^2) \end{aligned}$$

računamo $f'(0)$ *in* $f''(0)$ *za* $f(x) = e^x$, *dobimo:*

h	$f'(0)$	$f''(0)$
1	1.11752012	1.0861612
0.1	1.0016673	1.0008334
0.01	1.0000161	1.0000169
0.001	0.99999454	0.99837783
0.0001	0.99994244	1.4901161

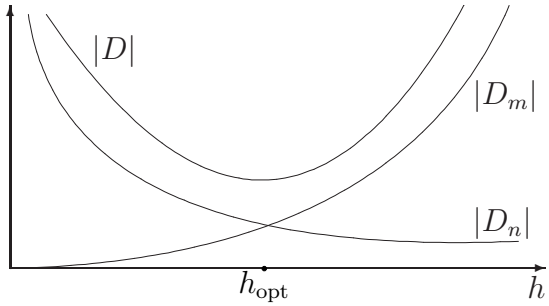
Čeprav je napake metode $\mathcal{O}(h^2)$, *dobimo pri premajhnem* h *slabše rezultate.* ■

Oglejmo si formulo $f''(x_1) = \frac{1}{h^2}(y_0 - 2y_1 + y_2) - \frac{1}{12}h^2 f^{(4)}(\xi)$. Napaka metode D_m je $-\frac{1}{12}h^2 f^{(4)}(\xi)$ in jo lahko ocenimo kot

$$|D_m| \leq \frac{h^2}{12}\|f^{(4)}\|.$$

Poleg napake metode pa se pojavi tudi neodstranljiva napaka D_n, saj namesto s točnimi vrednostmi $f(x_i)$ računamo s približki $\widetilde{y}_i$, za katere predpostavimo $|f(x_i) - \widetilde{y}_i| \leq \epsilon$. Ocenimo lahko

$$|D_n| \leq \frac{4\epsilon}{h^2}.$$



Pri računanju se pojavi še zaokrožitvena napaka D_z, ki pa jo za to analizo zanemarimo.

Ocena za celotno napako je

$$|D| \leq |D_m| + |D_n| \leq \frac{h^2}{12}\|f^{(4)}\| + \frac{4\epsilon}{h^2}.$$

Če poznamo oceno za $f^{(r)}$ in ϵ, lahko iz ocen za $|D_m|$ in $|D_n|$ določimo optimalni h, kjer bo ocena za skupno napako najmanjša.

Numerično odvajanje torej ni numerično dobro pogojen problem, saj zmanjšanje h napako poveča.

XI. Numerično integriranje

11.1 Kvadraturne formule

Računamo integral

$$\int_a^b f(x)\rho(x)dx,$$

kjer je ρ utež, ki mora biti nenegativna, ponavadi pa je $\rho(x) \equiv 1$. Integral aproksimiramo z vrednostmi funkcije f v točkah $x_0, \ldots, x_n$ s *kvadraturno formulo*

$$\int_a^b f(x)\rho(x)dx = \sum_{i=0}^{n} A_i f(x_i) + R(f),$$

pri čemer točke x_i imenujemo *vozli*, A_i so *koeficienti*, $R(f)$ pa je napaka.

Želimo, da bo formula točna za polinome čim višje stopnje, zato namesto f integriramo interpolacijski polinom. Če so vozli določeni, potem do formule pridemo tako, da:

a) integriramo Lagrangeve koeficiente: $A_i = \int_a^b L_{n,i}(x)\rho(x)dx$,

b) določimo $A_0, \ldots, A_n$ tako, da je formula točna za polinome čim višje stopnje (metoda nedoločenih koeficientov).

Za napako $R(f) = \int_a^b \frac{f^{(n+1)}(\xi)}{(n+1)!}\rho(x)dx$ uporabimo Peanov izrek.

11.2 Newton-Cotesova pravila

Pri *Newton-Cotesovih pravilih* so vozli ekvidistantni, $a = x_0, b = x_n$, utež pa je 1. Naj bo $y_k = f(x_k)$. Ločimo dva tipa N-C pravil:

a) *zaprti tip*: upoštevamo tudi krajišča: $\int_a^b f(x)dx = \sum_{k=0}^{n} A_k y_k + R_n(f)$.

b) *odprti tip*: brez krajišč: $\int_a^b f(x)dx = \sum_{k=1}^{n-1} B_k y_k + R_n(f)$.

Nekaj osnovnih zaprtih N-C pravil je:

- ($n = 1$) *Trapezno pravilo*: $\int_{x_0}^{x_1} f(x)dx = \frac{h}{2}(y_0 + y_1) - \frac{h^3}{12}f''(\xi)$.

  Kratka izpeljava:

  $$A_0 = \int_{x_0}^{x_1} \frac{x - x_1}{x_0 - x_1}dx = \frac{h}{2},$$

$$\begin{aligned} A_1 &= \int_{x_0}^{x_1} \frac{x-x_0}{x_1-x_0}dx = \frac{h}{2}, \\ R_1(f) &= \int_{x_0}^{x_1} \frac{f''(\xi_x)}{2}(x-x_0)(x-x_1)dx = \frac{f''(\xi)}{2}\int_{x_0}^{x_1}(x-x_0)(x-x_1)dx = -\frac{h^3}{12}f''(\xi). \end{aligned}$$

Pri izpeljavi napake smo uporabili izrek o povprečni vrednosti, saj je $(x-x_0)(x-x_1)$ konstantnega predznaka na $[x_0, x_1]$.

- $(n=2)$ *Simpsonovo pravilo*: $\int_{x_0}^{x_2} f(x)dx = \frac{h}{3}(y_0+4y_1+y_2) - \frac{h^5}{90}f^{(4)}(\xi)$.

  Formula je točna tudi za kubične polinome, saj za $f(x)=x^3$ velja

  $$\int_{x_0}^{x_2} \frac{f^{(3)}(\xi)}{6}(x-x_0)(x-x_1)(x-x_2)dx = 0.$$

  Za vse N-C formule z lihim številom točk (sodi n) zaradi simetrije velja, da so točne tudi za polinome stopnje $n+1$.

  Tu ne moremo več uporabiti izreka o povprečni vrednosti, na srečo pa za N-C pravila velja, da napako za zadostikrat odvedljive funkcije lahko ugotovimo iz napake pri x^{n+1} (x^{n+2} za sodi n).

- $(n=3)$ *3/8 pravilo*: $\int_{x_0}^{x_3} f(x)dx = \frac{3h}{8}(y_0+3y_1+3y_2+y_3) - \frac{3h^5}{80}f^{(4)}(\xi)$.

Nekaj osnovnih odprtih N-C pravil je:

- $(n=2)$ *Sredinsko pravilo*: $\int_{x_0}^{x_2} f(x)dx = 2hy_1 + \frac{h^3}{3}f''(\xi)$.
- $(n=3)$ $\int_{x_0}^{x_3} f(x)dx = \frac{3h}{2}(y_1+y_2) + \frac{3h^3}{4}f''(\xi)$.
- $(n=4)$ *Milnovo pravilo*: $\int_{x_0}^{x_4} f(x)dx = \frac{4h}{3}(2y_1-y_2+2y_3) + \frac{28h^5}{90}f^{(4)}(\xi)$.

Očitno je, da je vsota koeficientov enaka nh, a ker pri večjih n koeficienti postanejo tudi negativni, lahko pri velikem n pride do velikih neodstranljivih napak (čeprav sedaj h ni v imenovalcu), zato namesto večanja n sestavljamo pravila v sestavljene formule.

Osnovne sestavljene formule so:

- *Trapezna formula*:

  $$\int_{x_0}^{x_n} = \underbrace{h\left(\frac{1}{2}y_0 + y_1 + \cdots + y_{n-1} + \frac{1}{2}y_n\right)}_{T_h(f)} - \frac{h^2(x_n-x_0)}{12}f''(\xi).$$

  Kratka izpeljava:

  $$\int_{x_0}^{x_n} f(x)dx = \sum_{k=0}^{n-1}\int_{x_k}^{x_{k+1}} f(x)dx = \sum_{k=0}^{n-1}\left(\frac{h}{2}(y_k+y_{k+1}) - \frac{h^3}{12}f''(\xi_k)\right).$$

- *Simpsonova formula*:

$$\int_{x_0}^{x_n} f(x)dx = \underbrace{\frac{h}{3}\left(y_0 + 4y_1 + 2y_2 + 4y_3 + \cdots + 2y_{n-2} + 4y_{n-1} + y_n\right)}_{S_h(f)} - \frac{h^4(x_n - x_0)}{180} f^{(4)}(\xi).$$

- *Sredinska formula*:

$$\int_{x_0}^{x_n} f(x)dx = \underbrace{2h\left(y_1 + y_3 + \cdots + y_{n-1}\right)}_{U_h(f)} + \frac{h^2(x_n - x_0)}{6} f''(\xi).$$

Pri Simposonovi in sredinski formuli mora biti $n = 2m$.

11.3 Peanov izrek

Izrek 39 (Peano) *Naj bo L linearen funkcional oblike*

$$\begin{aligned} L(f) &= \int_a^b \left(a_0(x)f(x) + a_1(x)f'(x) + \cdots + a_n(x)f^{(n)}(x)\right)dx + \\ &+ \sum_{i=0}^{j_0} b_{i0}f(x_{i0}) + \sum_{i=0}^{j_1} b_{i1}f'(x_{i1}) + \cdots + \sum_{i=0}^{j_n} b_{in}f^{(n)}(x_{in}), \end{aligned}$$

kjer so funkcije a_i odsekoma zvezne na $[a,b]$, točke x_{ij} pa ležijo na intervalu $[a,b]$. Funkcional L naj bo za vse polinome stopnje n ali manj enak 0. Tedaj za vsako funkcijo f, ki je $(n+1)$-krat zvezno odvedljiva na $[a,b]$, velja

$$L(f) = \int_a^b f^{(n+1)}(t)K_n(t)dt,$$

kjer je K_n Peanovo jedro

$$K_n(t) = \frac{1}{n!}L((x-t)_+^n), \qquad (x-t)_+^n = \begin{cases} (x-t)^n, & x \geq t, \\ 0, & x < t. \end{cases}$$

Dokaz. Taylorjev izrek z ostankom v integralski obliki pravi

$$f(x) = f(a) + f'(a)(x-a) + \cdots + \frac{f^{(n)}(a)}{n!}(x-a)^n + \frac{1}{n!}\int_a^x f^{(n+1)}(t)(x-t)^n dt,$$

kar lahko zapišemo kot

$$f(x) = f(a) + f'(a)(x-a) + \cdots + \frac{f^{(n)}(a)}{n!}(x-a)^n + \frac{1}{n!}\int_a^b f^{(n+1)}(t)(x-t)_+^n dt. \qquad (11.19)$$

Če na obeh straneh (11.19) uporabimo L, dobimo

$$L(f) = \frac{1}{n!}L\left(\int_a^b f^{(n+1)}(t)(x-t)_+^n dt\right) = \frac{1}{n!}\int_a^b f^{(n+1)}(t)L((x-t)_+^n)dt,$$

saj lahko zamenjamo vrstni red L in integriranja. ■

Posledica 7 *Če je K_n konstantnega predznaka, potem za $(n+1)$-krat zvezno odvedljivo funkcijo f velja*

$$L(f) = \frac{f^{(n+1)}(\xi)}{(n+1)!} L(x^{n+1}).$$

Dokaz. Po Peanovem izreku velja $L(x^{n+1}) = \int_a^b (n+1)! K_n(t)dt$, torej je

$$\int_a^b K_n(t)dt = \frac{1}{(n+1)!} L(x^{n+1}).$$

Ker je K_n konstantnega predznaka, po izreku o povprečni vrednosti sledi

$$L(f) = \int_a^b f^{(n+1)}(t) K_n(t)dt = f^{(n+1)}(\xi) \int_a^b K_n(t)dt. \quad \blacksquare$$

Peanov izrek uporabimo za računanje napak kvadaraturnih formul in formul za numerično odvajanje.

Zgled 37 *Oceni napako trapeznega pravila pri integriranju enkrat zvezno odvedljive funkcije f. Standardne formule ne moremo uporabiti. ker f ni dvakrat zvezno odvedljiva. Uporabili bomo Peanov izrek in K_0.*

$$\begin{aligned} K_0(t) &= \int_{x_0}^{x_1} (x-t)_+^0 dx - \left(\frac{h}{2}((x_0-t)_+^0 + (x_1-t)_+^0) \right) = \\ &= \int_t^{x_1} (x-t)^0 dx - \left(\frac{h}{2}(0+1) \right) = \\ &= x_1 - t - \frac{h}{2} = \frac{x_0+x_1}{2} - t. \end{aligned}$$

Peanovo jedro ni konstantnega predznaka, zato lahko le ocenimo

$$|R(f)| \le \int_{x_0}^{x_1} |f'(t)| \left| \frac{x_0+x_1}{2} - t \right| dt \le \frac{h^2}{4} \|f'\|. \quad \blacksquare$$

11.4 Richardsonova ekstrapolacija

Z razpolavljanjem h pridemo pri sestavljenih formulah do točnejsih rezultatov. Iz približkov pri različnih h lahko ocenimo napako in ugotovimo, če je potrebno še nadaljne razpolavljanje.

Z ocenami se je prvi ukvarjal Richardson. Naj bo $S_h(f)$ Simpsonova formula s korakom h in $R_h(f)$ napaka Simpsonove formule pri koraku h. Vemo, da velja

$$I(f) = \int_a^b f(x)dx = S_h(f) + R_h(f) = S_{h/2}(f) + R_{h/2}(f).$$

Za napaki velja

$$R_h(f) = \frac{-(b-a)h^4}{180} f^{(4)}(\xi_1), \quad R_{h/2}(f) = \frac{-(b-a)h^4}{16 \cdot 180} f^{(4)}(\xi_2).$$

Če predpostavimo, da je $f^{(4)}(\xi_1) \approx f^{(4)}(\xi_2)$, dobimo

$$R_h(f) \approx 16 R_{h/2}(f).$$

Iz $R_{h/2}(f) = I(f) - S_{h/2}(f) = S_h(f) + R_h(f) - S_{h/2}(f) \approx S_h(f) + 16R_{h/2}(f) - S_{h/2}(f)$ dobimo

$$R_{h/2}(f) \approx \frac{S_{h/2}(f) - S_h(f)}{15}.$$

To formulo lahko uporabimo za oceno napake Simpsonove formule. Ocena pa ni preveč zanesljiva, saj smo izenačili odvode. V večini primerov vseeno dobimo spodobne rezultate. Poleg ocene lahko iz $S_{h/2}(f)$ in $S_h(f)$ z *ekstrapolacijo* dobimo še boljši približek, saj je

$$I(f) = S_{h/2}(f) + R_{h/2}(f) \approx \frac{16 S_{h/2}(f) - S_h(f)}{15}.$$

Podobno lahko naredimo pri trapezni in sredinski formuli.

11.5 Rombergova metoda

Zaporedna uporaba Richardsonove ekstrapolacije za trapezno formula je *Rombergova metoda.*

Definicija 14 *Bernoullijeva števila B_k so določena z razvojem*

$$\frac{x}{e^x - 1} = \sum_{k=0}^{\infty} \frac{B_k}{k!} x^k, \quad |x| < 2\pi.$$

Vsa Bernoullijeva števila so racionalna, vsa liha števila razen B_1 pa so enaka 0. Nakaj prvih Bernoullijevih števil je

$$B_0 = 1, \quad B_1 = -\frac{1}{2}, \quad B_2 = \frac{1}{6}, \quad B_4 = -\frac{1}{30}, \quad B_6 = \frac{1}{42}, \quad \ldots.$$

Izrek 40 *Za neskončnokrat zvezno odvedljivo funkcijo f velja* Euler-Maclaurinova sumacijska formula

$$I(f) = \int_a^b f(x)dx = T_h(f) - \sum_{k=1}^{\infty} \frac{B_{2k}}{(2k)!} h^{2k} \left(f^{(2k-1)}(b) - f^{(2k-1)}(a) \right). \tag{11.20}$$

Dokaz. Formulo bomo dokazali s simbolnim računanjem. Definiramo operatorje

$$\begin{aligned} Ef(x) &= f(x+h), \\ \Delta f(x) &= f(x+h) - f(x), \\ Df(x) &= f'(x). \end{aligned}$$

Veljajo naslednje formule:

$$I + \Delta = E$$

$$\begin{aligned} E &= e^{hD} \quad \text{(razvoj } f(x+h) \text{ v Taylorjevo vrsto)} \\ \Delta &= e^{hD} - I \\ \frac{1}{D} &= \int_a^x f(x)dt + C \quad \text{(nedoločeni integral)} \\ \frac{1}{\Delta} &= \frac{1}{E-I} = \frac{1}{e^{hD}-I} = \frac{1}{hD} - \frac{1}{2}I + \sum_{m=1}^{\infty} B_{2m}\frac{(hD)^{2m-1}}{(2m)!} \quad \left(\text{razvoj } \frac{z}{e^z-1}\right) \end{aligned}$$

Sedaj je

$$(E^n - I)\frac{1}{\Delta} = \frac{E^n - I}{E - I} = E^{n-1} + E^{n-2} + \cdots + E + I,$$

to pa pomeni

$$(E^n - I)\frac{1}{\Delta}f(x_0) = \sum_{k=0}^{n-1} f(x_k).$$

Po drugi strani je

$$(E^n - I)\frac{1}{\Delta}f(x_0) = \frac{1}{h}\int_{x_0}^{x_n} f(t)dt - \frac{1}{2}(f(x_n) - f(x_0)) + \sum_{m=1}^{\infty} B_{2m}\frac{h^{2m-1}}{(2m)!}\left(f^{(2m-1)}(x_n) - f^{(2m-1)}(x_0)\right),$$

od tod pa sledi formula (11.20). ■

Vrsta v (11.20) praviloma ni konvergentna, je pa asimptotska, ko gre $h \to 0$. Dobimo (ko $h \to 0$)

$$I(f) = T_h(f) - \frac{h^2}{12}(f'(b) - f'(a)) + \frac{h^4}{720}(f'''(b) - f'''(a)) + \cdots,$$

kar pomeni

$$I(f) = T_h(f) + \sum_{k=1}^{\infty} a_{k,0}h^{2k},$$

pri čemer so koeficienti $a_{k,0}$ neodvisni od h. Tako dobimo

$$\begin{aligned} I(f) &= T_h(f) + a_{1,0}h^2 + a_{2,0}h^4 + a_{3,0}h^6 + \cdots \\ I(f) &= T_{h/2}(f) + a_{1,0}\left(\frac{h}{2}\right)^2 + a_{2,0}\left(\frac{h}{2}\right)^4 + a_{3,0}\left(\frac{h}{2}\right)^6 + \cdots \\ I(f) &= T_{h/4}(f) + a_{1,0}\left(\frac{h}{4}\right)^2 + a_{2,0}\left(\frac{h}{4}\right)^2 + a_{3,0}\left(\frac{h}{4}\right)^6 + \cdots \end{aligned}$$

Če enačbo za $h/2$ pomnožimo s 4 in odštejemo od enačbe za h, se znebimo člena h^2 in dobimo točnejši približek:

$$\begin{aligned} I(f) &= T^{(1)}_{h/2}(f) + a_{2,1}h^4 + a_{3,1}h^6 + \cdots \\ I(f) &= T^{(1)}_{h/4}(f) + a_{2,1}\left(\frac{h}{2}\right)^4 + a_{3,1}\left(\frac{h}{2}\right)^6 + \cdots, \end{aligned}$$

kjer sta

$$T^{(1)}_{h/2}(f) = \frac{4T_{h/2}(f) - T_h(f)}{3}, \quad T^1_{h/4}(f) = \frac{4T_{h/4}(f) - T_{h/2}(f)}{3}.$$

Postopek sedaj nadaljujemo v

$$I(f) = T^{(2)}_{h/4}(f) + a_{3,2}h^6 + a_{4,2}h^8 + \cdots,$$

kjer je

$$T^{(2)}_{h/4}(f) = \frac{16T^{(1)}_{h/4}(f) - T^{(1)}_{h/2}(f)}{15}.$$

V splošnem postopku tvorimo shemo

napaka	$\mathcal{O}(h^2)$	$\mathcal{O}(h^4)$	$\mathcal{O}(h^6)$	$\mathcal{O}(h^8)$	$\cdots$
	$T^{(0)}_{h}(f)$				
	$T^{(0)}_{h/2}(f)$	$T^{(1)}_{h/2}(f)$			
	$T^{(0)}_{h/4}(f)$	$T^{(1)}_{h/4}(f)$	$T^{(2)}_{h/4}(f)$		
	$T^{(0)}_{h/8}(f)$	$T^{(1)}_{h/8}(f)$	$T^{(3)}_{h/8}(f)$	$T^{(4)}_{h/8}(f)$	

kjer je splošna formula

$$T^{(j)}_{h/2^k}(f) = \frac{4^j T^{(j-1)}_{h/2^k}(f) - T^{(j-1)}_{h/2^{k-1}}(f)}{4^j - 1}.$$

Zgled 38 *Z Rombergovo metodo izračunaj* $\int_1^{2.2} \ln x = 0.5346062$. *Začni s* $h = 0.6$ *in naredi dve razpolavljanji.*

Dobimo:

$$\begin{aligned}
T^{(0)}_{h} &= 0.6(\frac{1}{2}\ln 2.2 + \ln 1.6 + \frac{1}{2}\ln 2.2) = 0.5185394 \\
T^{(0)}_{h/2} &= \frac{1}{2}T^{(0)}_{h} + 0.3(\ln 1.3 + \ln 1.9) = 0.5305351 \\
T^{(0)}_{h/4} &= \frac{1}{2}T^{(0)}_{h/2} + 0.15(\ln 1.15 + \ln 1.45 + \ln 1.75 + \ln 2.05) = 0.5335847
\end{aligned}$$

Pomembno je, da $T_{h/2^k}$ *vedno računamo kot*

$$T_{h/2^k} = \frac{1}{2}T_{h/2^{k-1}} + \frac{h}{2^k}(y_1 + y_3 + \cdots + y_{2^k-1}).$$

Tako vsako funkcijsko vrednost izračunamo enkrat in imamo z Rombergom zanemarljivo dodatnega dela v primerjavi z računanjem $T^{(0)}_{h/2^k}$, *rezultat pa je lahko mnogo natančnejši.*

Sedaj z Rombergovo ekstrapolacijo dobimo

$$\begin{aligned}
T^{(1)}_{h/2} &= \frac{4T^{(0)}_{h/2} - T^{(0)}_{h}}{3} = 0.5345337 \\
T^{(1)}_{h/4} &= \frac{4T^{(0)}_{h/4} - T^{(0)}_{h/2}}{3} = 0.5346013 \\
T^{(2)}_{h/4} &= \frac{16T^{(1)}_{h/4} - T^{(1)}_{h/2}}{15} = 0.5346058. \quad \blacksquare
\end{aligned}$$

11.6 Gaussove kvadraturne formule

Integral $\int_a^b f(x)\rho(x)dx$, kjer je ρ nenegativna utež, aproksimiramo s kvadraturno formulo

$$\int_a^b f(x)\rho(x)dx = \sum_{i=0}^{n} A_i^{(n)} f(x_i^{(n)}) + R(f).$$

Koeficienti so določeni z vozli, saj velja $A_i^{(n)} = \int_a^b L_{n,i}(x)\rho(x)dx$, formula pa je točna za polinome stopnje vsaj n. S primerno izbiro vozlov lahko dosežemo, da bo formula točna za polinome stopnje vsaj $2n+1$, v ozadju pa so ortogonalni polinomi.

Za funkcije lahko na $[a,b]$ definiramo skalarni produkt kot

$$\langle f, g\rangle = \int_a^b f(x)g(x)\rho(x)dx.$$

Funkciji f in g sta ortogonalni, če je $\langle f, g\rangle = 0$. Iz standardne baze polinomov $1, x, x^2, \ldots$ z ortogonalizacijo dobimo ortonormirano bazo $P_0(x), P_1(x), P_2(x), \ldots$, kjer je P_i polinom stopnje i in velja

$$\langle P_i, P_k\rangle = \delta_{ik}.$$

Naj bo sedaj P_{n+1} tak normiran polinom, da je $\langle P_{n+1}, q\rangle = 0$ za vsak polinom q stopnje kvečjemu n in naj bo $P_{n+1} = k_{n+1}(x - x_0^{(n)})\cdots(x - x_n^{(n)})$. Pri *Gaussovi kvadraturni formuli* za vozle izberemo ničle $x_0^{(n)}, \ldots, x_n^{(n)}$, torej je $\omega(x) = (x - x_0^{(n)})\cdots(x - x_n^{(n)})$.

Poljuben polinom f stopnje $2n+1$ lahko zapišemo kot $f(x) = q(x)\omega(x) + r(x)$, kjer sta q, r polinoma stopnje kvečjemu n. To pomeni:

$$\begin{aligned}\int_a^b f(x)\rho(x)dx &= \int_a^b q(x)\omega(x)\rho(x)dx + \int_a^b r(x)\rho(x)dx \\ &= 0 + \sum_{i=0}^{n} A_i^{(n)} r(x_i^{(n)}) = \sum_{i=0}^{n} A_i^{(n)} f(x_i^{(n)}).\end{aligned}$$

Torej je pravilo točno za vse polinome stopnje $2n+1$ ali manj.

Lema 29 *Uteži Gaussovih kvadaturnih pravil so pozitivne.*

Dokaz. Vzemimo

$$P_i(x) = \frac{\omega^2(x)}{(x - x_i^{(n)})^2}$$

za $i = 0, \ldots, n$. P_i je polinom stopnje $2n$, torej velja

$$\int_a^b P_i(x)\rho(x)dx = \sum_{k=0}^{n} A_k P_i(x_k^{(n)}) = A_i P_i(x_i^{(n)}).$$

Ker je $P_i(x_i^{(n)}) > 0$ in je P_i nenegativna funkcija, mora biti $A_i > 0$. ∎

Koeficiente A_i lahko izračunamo z integriranjem Lagrangevih koeficientov, še bolje pa je, če uporabimo *Darboux-Cristoffelove formule*:

$$A_i^{(n)} = \frac{1}{\sum_{j=0}^{n} P_j^2(x_k^{(n)})}, \quad i = 0, \ldots, n. \tag{11.21}$$

Iz teorije ortogonalnih polinomov sledi, da so vse ničle $x_i^{(n)}$ enostavne, realne in leže na (a, b).

Izrek 41 *Za $f \in \mathcal{C}^{(2n+2)}[a, b]$ velja*

$$\int_a^b f(x)\rho(x) = \sum_{i=0}^{n} A_i^{(n)} f(x_i^{(n)}) + \frac{f^{(2n+2)}(\xi)}{(2n+2)!k_{k+1}^2}.$$

Dokaz. Vzamemo Hermiteov interpolacijski polinom H v točkah $x_0^{(n)}, \ldots, x_n^{(n)}$, za katerega velja $H(x_i^{(n)}) = f(x_i^{(n)})$, $H'(x_i^{(n)}) = f'(x_i^{(n)})$, $i = 0, \ldots, n$. Vemo

$$f(x) = H(x) + \frac{f^{(2n+2)}(\xi)}{(2n+2)!}\omega^2(x).$$

Sledi

$$\int_a^b f(x)\rho(x)dx = \int_a^b H(x)\rho(x)dx + \int_a^b \frac{f^{(2n+2)}(\xi)}{(2n+2)!}\omega^2(x)dx.$$

Ker je H polinom stopnje $2n+1$, velja

$$\int_a^b H(x)\rho(x)dx = \sum_{i=0}^{n} A_k H(x_k^{(n)}) = \sum_{i=0}^{n} A_k f(x_k^{(n)}),$$

za drugi integral pa velja

$$\int_a^b \frac{f^{(2n+2)}(\xi)}{(2n+2)!}\omega^2(x)dx = \int_a^b \frac{f^{(2n+2)}(\xi)}{(2n+2)!} \cdot \frac{P_{n+1}^2(x)}{k_{n+1}^2}dx = \frac{f^{(2n+2)}(\xi)}{(2n+2)!k_{n+1}^2}\int_a^b P_{n+1}^2(x)dx. \quad \blacksquare$$

Najpogostejši ortogonalni polinomi so:

$[-1, 1]$,	$\rho(x) = 1$,		(Legendre)
$[-1, 1]$,	$\rho(x) = (1-x^2)^{-\frac{1}{2}}$,		(Čebišev 1. vrste)
$[-1, 1]$,	$\rho(x) = (1-x^2)^{\frac{1}{2}}$,		(Čebišev 2. vrste)
$[-1, 1]$,	$\rho(x) = (1-x)^{\alpha}(1+x)^{\beta}$,	$\alpha, \beta > -1$,	(Jacobi)
$[-1, 1]$,	$\rho(x) = (1-x^2)^{\sigma-\frac{1}{2}}$,	$\sigma > \frac{1}{2}$,	(Gegenbauer)
$[0, \infty)$,	$\rho(x) = x^{\sigma}e^{-x}$,	$\sigma > -1$	(Laguerre)
$(-\infty, \infty)$,	$\rho(x) = e^{-x^2}$,		(Hermite).

Za te ortogonalne polinome imamo tabelirane ničle in Gaussove kvadraturne formule.

Zgled 39 *Gauss-Legendrovi kvadraturni formuli na dveh in treh točkah sta*

$$\int_{-1}^{1} f(x)dx = f\left(-\sqrt{\frac{1}{3}}\right) + f\left(\sqrt{\frac{1}{3}}\right) + \frac{1}{135}f^{(4)}(\xi),$$

$$\int_{-1}^{1} f(x)dx = \frac{5}{9}f\left(-\sqrt{\frac{3}{5}}\right) + \frac{8}{9}f(0) + \frac{5}{9}f\left(\sqrt{\frac{3}{5}}\right) + \frac{1}{15750}f^{(6)}(\xi).$$

Za primerjavo pri trapeznem in Simpsonovem pravilu dobimo

$$\int_{-1}^{1} f(x)dx = f(-1) + f(1) - \frac{2}{3}f^{(2)}(\xi),$$

$$\int_{-1}^{1} f(x)dx = \frac{1}{3}f(-1) + \frac{4}{3}f(0) + \frac{1}{3}f(1) + \frac{1}{90}f^{(4)}(\xi). \quad \blacksquare$$

11.7 Večdimenzionalni integrali

Denimo, da računamo integral funkcije f dveh spremenljivk po pravokotniku $\Omega = [a,b]\times[c,d]$. Pišemo lahko

$$\int_{\Omega} f(x,y)dxdy = \int_a^b dx \int_c^d f(x,y)dy.$$

Če $[a,b]$ razdelimo s točkami $x_i = a + ih$, $i = 0,\ldots,n$, $h = (b-a)/n$ in $[c,d]$ s točkami $y_j = c + jk$, $j = 0,\ldots,m$, $k = (d-c)/m$, in za integrale uporabimo trapezno pravilo, dobimo

$$\int_a^b dx \int_c^d f(x,y)dy = \frac{hk}{4}\sum_{i=0}^{n}\sum_{j=0}^{m} A_{ij}f(x_i,y_j) + \mathcal{O}(h^2+k^2),$$

kjer za koeficiente A_{ij} velja $A_{ij} = A_1(i)A_2(j)$ in $A_1(0) = A_1(n) = 1$, $A_1(i) = 2$ za $i \in \{1,\ldots,n-1\}$, ter $A_2(0) = A_2(m) = 1$, $A_2(j) = 2$ za $j \in \{1,\ldots,m-1\}$. V bistvu dobimo tenzorski produkt sestavljenega trapeznega pravila, podobno pa lahko naredimo tudi za ostala sestavljena pravila.

Če je $\Omega = [0,1]^d$ in v vsaki dimenziji porabimo n točk, uporabimo pa sestavljeno pravilo reda k, potem za dobljeno tenzorsko pravilo velja, da je napaka $\mathcal{O}(N^{-k/d})$, kjer je $N = n^d$ število vseh točk.

Težava je, da pri velikem d prvič potrebujemo zelo veliko točk (n^d), po drugi strani pa napaka prepočasi pada glede na število uporabljenih točk. Zaradi tega se pri velikih n bolje obnaša metoda Monte-Carlo, kjer lahko z znosnim številom točk ponavadi dobimo zadovoljiv približek.

11.8 Metoda Monte-Carlo

Integral $I(f) = \int_0^1 f(x)dx$ je enak povprečni vrednosti f na $[0,1]$. To je osnova za *metodo Monte-Carlo*.

Če je X slučajna spremenljivka, enakomerno porazdeljena po $[0,1]$, potem za matematično upanje velja $E(f(X)) = I(f)$. Približek za matematično upanje je

$$I_N(f) = \frac{1}{N}\sum_{i=1}^{N} f(X_i),$$

kjer so X_i neodvisne naključne vrednosti z $[0,1]$.

Za napako $\epsilon_N(f) = I(f) - I_N(f)$ velja

$$\epsilon \approx \sigma(f) N^{-1/2},$$

kjer je $\sigma(f) = \sqrt{D(f)}$ standardna deviacija,

$$D(f) = \int_0^1 (f(x) - I(f))^2 dx$$

pa varianca funkcije f.

Pri računanju integrala po $\Omega = [0,1]^d$ velja enako, le kot X_i sedaj izbiramo vektorje iz Ω, kjer je vsak element naključno število z $[0,1]$.

Pravih naključnih števil z računalnikom ne moremo generirati, zato govorimo o *psevdo-naključnih številih*. Eden izmed načinov za njihovo generiranje je *linearni kongruenčni model*

$$x_{r+1} = ax_r + c \pmod{m},$$

kjer so a, c, m naravna števila, ki jih izberemo tako, da je perioda čim večja.

XII. Diferencialne enačbe

12.1 Uvod

Rešujemo *začetni problem prvega reda* v obliki

$$\begin{aligned} y' &= f(x,y) \\ y(x_0) &= y_0, \end{aligned}$$

kjer je f dana dovolj gladka funkcija x in y, zanima pa nas rešitev na $[x_0, b]$. Želimo, da je problem dobro pogojen, torej da rešitev obstaja in da je enolična.

- *obstoj rešitve:* Npr. za $y' = 1 + y^2$, $y(0) = 0$, je analitična rešitev $y(x) = \tan x$. Rešitev strmo (vedno bolj) pada in pri končni vrednosti x (pri $\pm\pi/2$) pridemo do $\pm\infty$. Torej rešitev obstaja le na intervalu $(-\pi/2, \pi/2)$.
- *enoličnost:* Npr. za $y' = y^{2/3}$, $y(0) = 0$, je ena rešitev $y(x) = 0$, druga pa $y(x) = \frac{1}{27}x^3$, torej ni enoličnosti.

Pravio, da je f *Lipschitzova* na y (*zadošča L-pogoju* na y), če velja

$$|f(x,y) - f(x,\widetilde{y}| \leq L|y - \widetilde{y}|.$$

Izrek 42 *Če je $f(x,y)$ zvezna na $x \in [a,b]$, $y \in \mathbb{R}$, in zadošča L-pogoju, potem na $x \in [a,b]$ obstaja enolična rešitev začetnega problema.* ■

Računamo približke $y_1, y_2, \ldots$ za vrednosti $y(x_1), y(x_2), \ldots$ v točkah $x_0 < x_1 < x_2 < \cdots$. Označimo

$$\begin{aligned} y(x_i) &: \text{točna vrednost rešitve v } y, \\ y_i &: \text{izračunani približek za } y(x_i). \end{aligned}$$

Metode za reševanje začetnega problema delimo na:

- *enokoračne metode*: y_{n+1} izračunamo iz y_n,
- *večkoračne metode*: y_{n+1} izračunamo iz $y_n, y_{n-1}, \ldots, y_{n-k}$.

Metode ločimo še na:

- *eksplicitne metode*: imamo direktno formulo za y_{n+1},
- *implicitne metode*: y_{n+1} dobimo tako, da rešimo nelinearno enačbo.

12.2 Enokoračne metode

a) *Eulerjeva metoda*

- *eksplicitna*: $y_{n+1} = y_n + hf(x_n, y_n)$,
- *implicitna*: $y_{n+1} = y_n + hf(x_{n+1}, y_{n+1})$.

b) *Taylorjeva vrsta*

Če enačbo $y' = f(x, y)$ odvajamo, dobimo

$$\begin{aligned} y' &= f \\ y'' &= f_x + f_y y' = f_x + f_y f \\ y''' &= f_{xx} + 2f_{xy}f + f_{yy}f^2 + f_y(f_x + f_y f) \\ &\vdots \end{aligned}$$

Po Taylorjevi vrsti izračunamo $y(x_1) = y(x_0 + h) = y_0 + hy_0' + \frac{h^2}{2}y_0'' + \cdots$.

Zgled 40 $y' = xy + 1$, $y(0) = 0$, $y(0.2) = ?$

$$\begin{aligned} y' = xy + 1 &\implies y'(0) = 1 \\ y'' = xy' + y &\implies y''(0) = 0 \\ y''' = xy'' + 2y' &\implies y'''(0) = 2 \end{aligned}$$

$$y(h) = h + \frac{1}{3}h^3 + \cdots.$$

Pri $h = 0.2$ *dobimo* $y_1 = 0.2 + 0.00267 = 0.20267$. *Če iščemo* $y(0.4)$, *nadaljujemo s točko* $(0.2, 0.20267)$. ■

Definicija 15 *Pravimo, da je metoda reda* k, *če se pri točni vrednosti* $y_n = y(x_n)$ *izračunani* y_{n+1} *ujema z razvojem* $y(x_n + h)$ *v Taylorjevo vrsto okrog* x_n *do vključno člena* h^k.

Metoda v zgornjem zgledu ima red 3, sicer pa z razvijanjem v Taylorjevo vrsto lahko dobimo metodo poljubnega reda.

c) *Runge-Kutta metoda*

Najprej izračunamo

$$k_i = hf(x_n + \alpha_i h, y_n + \sum_{j=1}^{i} \beta_{ij} k_j), \quad i = 1, \ldots, m,$$

nato pa

$$y_{n+1} = y_n + \sum_{i=1}^{m} \gamma_i k_i.$$

Pri tem je m stopnja R-K metode, kar ne smemo zamenjevati z redom metode. Konstante α_i, β_{ij} in γ_i določimo tako, da se y_{n+1} čim bolj ujema z razvojem $y(x_n + h)$ v Taylorjevo vrsto. Pri tem ponavadi velja $\alpha_i = \sum_{j=1}^{i} \beta_{ij}$.

V primeru, ko je $\beta_{ii} = 0$ za $i = 1, \ldots, m$, je metoda eksplicitna, sicer pa implicitna.

Zgled 41 *Dvostopenjska eksplicitna R-K metoda ima obliko*

$$\begin{aligned} k_1 &= hf(x_n, y_n) \\ k_2 &= hf(x_n + \alpha h, y_n + \beta k_1) \\ y_{n+1} &= y_n + \gamma_1 k_1 + \gamma_2 k_2. \end{aligned}$$

Dobimo

$$\begin{aligned} k_1 &= hf \\ k_2 &= hf + \alpha h^2 f_x + \beta h k_1 f_y + \mathcal{O}(h^3) \end{aligned}$$

in

$$y_{n+1} = y_n + (\gamma_1 + \gamma_2) hf + \gamma_2 \alpha h^2 f_x + \gamma_2 \beta h^2 f f_y + \mathcal{O}(h^3),$$

kar primerjamo z

$$y(x_n + h) = y(x_n) + hf + \frac{1}{2} h^2 (f_x + f f_y) + \mathcal{O}(h^3).$$

Sledi

$$\begin{aligned} \gamma_1 + \gamma_2 &= 1 \\ \alpha \gamma_2 &= \tfrac{1}{2} \\ \beta \gamma_2 &= \tfrac{1}{2}. \end{aligned}$$

Sistem ima več rešitev, saj za poljubni $\gamma_2 \neq 0$ dobimo

$$\gamma_1 = 1 - \gamma_2, \quad \alpha = \frac{1}{2\gamma_2}, \quad \beta = \frac{1}{2\gamma_2}. \quad \blacksquare$$

Dva primera dvostopenjskih R-K metod drugega reda sta:

- *Heunova metoda*

$$\begin{aligned} k_1 &= hf(x_n, y_n) \\ k_2 &= hf(x_n + h, y_n + k_1) \\ y_{n+1} &= y_n + \tfrac{1}{2}(k_1 + k_2), \quad \text{napaka}: \ \mathcal{O}(h^3). \end{aligned}$$

- *modificirana Eulerjeva metoda*

$$\begin{aligned} k_1 &= hf(x_n, y_n) \\ k_2 &= hf(x_n + \tfrac{1}{2}h, y_n + \tfrac{1}{2}k_1) \\ y_{n+1} &= y_n + k_2, \quad \text{napaka}: \ \mathcal{O}(h^3). \end{aligned}$$

Runge-Kuttina 4 stopenjska metoda reda 4 je

$$\begin{aligned} k_1 &= hf(x_n, y_n) \\ k_2 &= hf(x_n + \tfrac{1}{2}h, y_n + \tfrac{1}{2}k_1) \\ k_3 &= hf(x_n + \tfrac{1}{2}h, y_n + \tfrac{1}{2}k_2) \\ k_4 &= hf(x_n + h, y_n + k_3) \\ y_{n+1} &= y_n + \tfrac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4), \quad \text{napaka}: \ \mathcal{O}(h^5). \end{aligned}$$

12.3 Adaptivna ocena koraka

Če imamo na voljo oceno lokalne napake, lahko h adaptivno prilagajamo. Denimo, da po 4 stopenjski R-K metodi reda 4 iz $y(x)$ izračunamo $y(x+2h)$ enkrat s korakom $2h$, drugič pa v dveh korakih s korakom h. Podobno kot pri Richardsonovi ekstrapolaciji dobimo

$$\begin{aligned} y(x+2h) &= y^{(1)} + (2h)^5 \cdot C_1 + \mathcal{O}(h^6) \\ y(x+2h) &= y^{(2)} + 2(h)^5 \cdot C_2 + \mathcal{O}(h^6) \end{aligned}$$

in

$$\Delta = \frac{y^{(2)} - y^{(1)}}{15}$$

je ocena za lokalno napako $y^{(1)}$. Ko je Δ velik, razpolovimo h, če je dovolj majhen pa lahko h podvojimo. Za izračun Δ in $y^{(1)}$ potrebujemo 11 izračunov f (4 izračune za vsako R-K, prvi pa se ponovi dvakrat).

Boljša je *Fehlbergova metoda*, kjer vzamemo 6 stopenjsko R-K metodo

$$k_i = hf(x_n + \alpha_i h, y_n + \sum_{j=1}^{i-1} \beta_{ij} k_j), \quad i = 1, \ldots, 6,$$

potem pa iz istih $k_1, \ldots, k_6$ sestavimo metodo reda 5

$$y_{n+1} = y_n + \sum_{i=1}^{6} \gamma_i k_i$$

in metodo reda 4

$$y^*_{n+1} = y_n + \sum_{i=1}^{6} \gamma^*_i k_i.$$

Ocena za napako y^*_{n+1} je potem kar $y_{n+1} - y^*_{n+1} = \sum_{i=1}^{6} (\gamma_i - \gamma^*_i) k_i$.

12.4 Stabilnost in konvergenca enokoračnih metod

Vsako enokoračno metodo lahko zapišemo v obliki

$$y_{n+1} = y_n + h\phi(x_n, y_n, h),$$

kjer je ϕ *funkcija prirastka*. Pravimo, da je metoda *konsistentna*, če velja

$$\lim_{h \to 0} \phi(x, y, h) = f(x, y).$$

Zgled 42 *Funkcija prirastka za modificirano Eulerjevo metodo*

$$y_{n+1} = y_n + hf(x_n + \frac{h}{2}, y_n + \frac{1}{2} hf(x_n, y_n))$$

je $\phi(x_n, y_n, h) = f(x_n + \frac{h}{2}, y_n + \frac{1}{2} hf(x_n, y_n))$. *Metoda je očitno konsistentna.* ■

Lokalno napako pri izračunu y_{n+1} lahko zapišemo kot $T(x_{n+1}) = y(x_{n+1}) - y(x_n) - h\phi(x_n, y_n, h)$. Če velja $T(x_{n+1}) = \mathcal{O}(h^{p+1})$, potem je metoda reda p.

Pravimo, da je numerična metoda *stabilna*, če za vsako diferencialno enačbo, ki zadošča L-pogoju obstajata taka $h_0 > 0$ in $K > 0$, da za poljubni dve rešitvi y_n, $\widetilde{y}_n$ velja $\|y_n - \widetilde{y}_n\| \leq K\|y_0 - \widetilde{y}_0\|$ za $h \leq h_0$.

Izrek 43 *Če funkcija prirastka ϕ zadošča L-pogoju na y, potem je metoda stabilna.*

Izrek 44 *Če je funkcija prirastka ϕ zvezna v x, y, h in zadošča L-pogoju na y, potem je metoda konvergentna natanko tedaj, ko je metoda konsistentna.*

Izrek 45 (O globalni napaki) *Če funkcija prirastka ϕ zadošča pogojem za konvergenco in za lokalno napako velja $\|T(x)\| \leq Dh^{p+1}$, potem za globalno napako velja*

$$\|y_n - y(x_n)\| \leq Dh^p \frac{e^{L(x_n - x_0)} - 1}{L} + e^{L(x_n - x_0)}\|y_0 - y(x_0)\|.$$

12.5 Večkoračne metode

Splošni nastavek je

$$\sum_{i=0}^{k} \alpha_i y_{n-i} + h \sum_{i=0}^{k} \beta_i f_{n-i} = 0,$$

kjer je $f_i = f(x_i, y_i)$, privzamemo pa še $\alpha_0 = 1$. Če je $\beta_0 = 0$, je metoda eksplicitna, sicer pa implicitna. Metodo določata *rodovna polinoma*

$$\begin{aligned} \rho(z) &= \sum_{i=0}^{k} \alpha_{k-i} z^i, \\ \sigma(z) &= \sum_{i=0}^{k} \beta_{k-i} z^i. \end{aligned}$$

Adamsove metode dobimo tako, da enačbo $y' = f(x, y)$ integriramo na $[x_n, x_{n+1}]$

$$y_{n+1} - y_n = \int_{x_n}^{x_{n+1}} y' dx = \int_{x_n}^{x_{n+1}} f(x, y) dx,$$

f pa nadomestimo z interpolacijskim polinomom na točkah:

a) $x_n, x_{n-1}, \ldots, x_{n-k+1}$: eksplicitne *Adams-Bashworthove formule*

$$\begin{aligned} y_{n+1} &= y_n + hf_n, \quad \text{napaka } \mathcal{O}(h^2), \\ y_{n+1} &= y_n + \frac{h}{2}(3f_n - f_{n-1}), \quad \text{napaka } \mathcal{O}(h^3), \\ y_{n+1} &= y_n + \frac{h}{12}(23f_n - 16f_{n-1} + 5f_{n-2}), \quad \text{napaka } \mathcal{O}(h^4). \end{aligned}$$

b) $x_{n+1}, x_n, \ldots, x_{n-k+2}$: implicitne *Adams-Moultonove formule*

$$\begin{aligned} y_{n+1} &= y_n + hf_{n+1}, \quad \text{napaka } \mathcal{O}(h^2), \\ y_{n+1} &= y_n + \frac{h}{2}(f_{n+1} + f_n), \quad \text{napaka } \mathcal{O}(h^3), \\ y_{n+1} &= y_n + \frac{h}{12}(5f_{n+1} + 8f_n - f_{n-1}), \quad \text{napaka } \mathcal{O}(h^4). \end{aligned}$$

Pri predpostavki, da je $k = n$ in $x_0 = 0$, definiramo linearni funkcional

$$L(y) = \sum_{i=0}^{k} \Big(\alpha_i y((k-i)h) + h\beta_i y'((k-i)h)\Big) = \sum_{j=0}^{k} \Big(\alpha_{k-j} y(jh) + h\beta_{k-j} y'(jh)\Big).$$

Če y in y' razvijemo v Taylorjevo vrsto okrog 0, dobimo

$$L(y) = d_0 y(0) + d_1 h y'(0) + d_2 h^2 y''(0) + \cdots.$$

Metoda je reda p, če je $L(y) = \mathcal{O}(n^{p+1})$ za vsako $(p+1)$-krat zvezno odvedljivo funkcijo y.

Izrek 46 *Ekvivalentno je:*

a) $d_0 = d_1 = \cdots = d_m = 0$,

b) $L(q) = 0$ *za poljuben polinom* q *stopnje kvečjemu* m,

c) $L(y) = \mathcal{O}(h^{m+1})$ *za vsako* $(m+1)$*-krat zvezno odvedljivo funkcijo* y.

Red metode je p, če velja $d_0 = d_1 = \cdots = d_p = 0 \neq d_{p+1}$.

Če v Taylorjevo vrsto razvijemo y in y', dobimo

$$\begin{aligned} y(jh) &= \sum_{r=0}^{\infty} \frac{(jh)^r}{r!} y^{(r)}(0), \\ y'(jh) &= \sum_{r=0}^{\infty} \frac{(jh)^r}{r!} y^{(r+1)}(0), \end{aligned}$$

Sedaj dobimo naslednje formule za $d_0, d_1, d_2, \ldots$:

$$\begin{aligned} d_0 &= \sum_{j=0}^{k} \alpha_{k-j}, \\ d_1 &= \sum_{j=0}^{k} (j\alpha_{k-j} + \beta_{k-j}), \\ d_2 &= \sum_{j=0}^{k} \Big(\frac{j^2}{2}\alpha_{k-j} + j\beta_{k-j}\Big), \\ &\vdots \\ d_q &= \sum_{j=0}^{k} \Big(\frac{j^q}{q!}\alpha_{k-j} + \frac{j^{q-1}}{(q-1)!}\beta_{k-j}\Big). \end{aligned}$$

Zgled 43 *Določi red večkoračne metode* $y_n - y_{n-2} = \frac{h}{3}(f_n + 4f_{n-1} + f_{n-2})$.

Koeficienti so $\alpha_0 = -1$, $\alpha_1 = 0$, $\alpha_2 = 1$, $\beta_0 = \frac{1}{3}$, $\beta_1 = \frac{4}{3}$, $\beta_2 = \frac{1}{3}$. *Po zgornjih formulah lahko izračunamo* $d_0 = d_1 = \cdots = d_4 = 0$ *in* $d_5 = -\frac{1}{90}$, *kar pomeni, da je red metode enak* 4. ■

Ponavadi pri večkoračnih metodah uporabljamo *prediktor-korektor metode*, kjer za izračun prediktorja y_{n+1}^P vzamemo eksplicitno metodo, za korektor pa vzamemo implicitno metodo, katere enačbo rešujemo iterativno tako, da na desno stran vstavimo približek y_{n+1}^P, na levi pa dobimo y_{n+1}^K. Tako se izognemu reševanju nelinearne enačbe.

Primer so *Milnove metode*, kjer $y' = f(x, y)$ integriramo na $[x_{n-k}, x_{n+1}]$:

$$y_{n+1} - y_{n-k} = \int_{x_{n-k}}^{x_{n+1}} f(x, y)dx,$$

integral pa računamo preko Newton-Cotesovih formul. Pri odprti dobimo eksplicitno, pri zaprti pa implicitno metodo. Primer je Milnov par

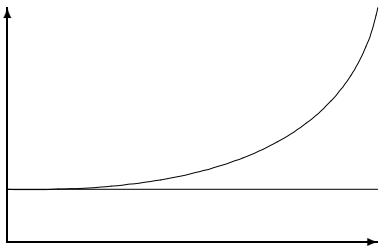
$$\begin{aligned} y_{n+1} &= y_{n-3} + \frac{4h}{3}(2f_n - f_{n-1} + 2f_{n-2}), \quad \text{napaka } \mathcal{O}(h^5) \quad : \text{ prediktor} \\ y_{n+1} &= y_{n-1} + \frac{h}{3}(f_{n+1} + 4f_n + f_{n-1}), \quad \text{napaka } \mathcal{O}(h^5) \quad : \text{ korektor} \end{aligned}$$

12.6 Stabilnost

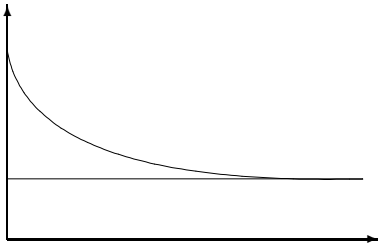
Inherentna nestabilnost je odvisna od problema in neodvisna od numerične metode.

Zgledi so:

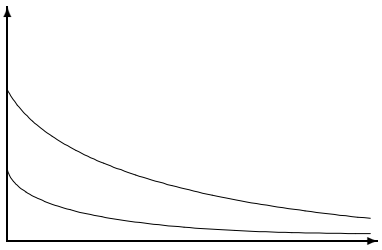
a) $y' = y - 1$, $y(0) = 1$, splošna rešitev je $y(x) = Ce^x + 1$ in $C = 0$, numerično pa dobimo $C \neq 0$. Problem je inherentno absolutno in relativno nestabilen.



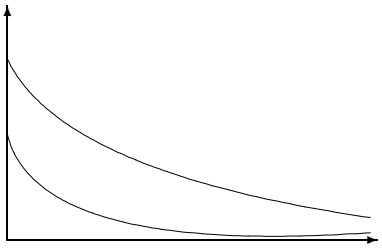
b) $y' = -y + 1$, $y(0) = 1$, splošna rešitev je $y(x) = Ce^{-x} + 1$ in $C = 0$, numerično pa dobimo $C \neq 0$. Problem je inherentno absolutno in relativno stabilen.



c) $y' = y + e^{2x}$, $y(0) = 1$, splošna rešitev je $y(x) = Ce^x + e^{2x}$ in $C = 0$, numerično pa dobimo $C \neq 0$. Problem je inherentno absolutno nestabilen in relativno stabilen.



d) $y' = -y - e^{2x}$, $y(0) = 1$, splošna rešitev je $y(x) = Ce^{-x} + e^{-2x}$ in $C = 0$, numerično pa dobimo $C \neq 0$. Problem je inherentno absolutno stabilen in relativno nestabilen.



Pri relativni nestabilnosti včasih pomaga, če obrnemo interval:

a) $y' = y - 1$, $y(0) = 1$. Vzamemo $y(x_n) = 1$ in računamo nazaj.

b) $y' = -y - e^{2x}$, $y(0) = 1$. Vzamemo $y(x_n) = 0$ in računamo nazaj.

Inducirana nestabilnost se pojavi pri večkoračnih metodah. Metoda

$$\sum_{i=0}^{k} \alpha_i y_{n-i} + h \sum_{i=0}^{k} \beta_i f_{n-i} = 0$$

mora biti stabilna za enačbo $y'(0) = 0$ (temu pravimo *ničelna stabilnost*).

Dobimo diferenčno enačbo

$$\sum_{i=0}^{k} \alpha_i y_{n-i} = 0.$$

Njen karakteristični polinom $\rho(\xi)$ ima k ničel $\xi_1, \ldots, \xi_k$. Splošna rešitev je (pri enostavnih ničlah)

$$y_n = \sum_{j=1}^{k} A_j \xi_j^n.$$

Za stabilnost mora veljati (Dahlquistov izrek)

a) $|\xi_i| \leq 1$ za $i = 1, \ldots, k$,

b) če je $|\xi_j| = 1$, mora biti ξ_j enostavna ničla.

Večkoračna metoda je konsistentna natanko tedaj, ko velja $\rho(1) = 0$ in $\rho'(1) + \sigma(1) = 0$, ta dva pogoja pa sta ekvivalentna temu, da je red vsaj 1.

Izrek 47 *Večkoračna metoda je konvergentna natanko tedaj, ko je ničelno stabilna in konsistentna.*

12.7 Začetni problemi drugega reda

Rešujemo

$$\begin{aligned} y'' &= f(x, y, y') \\ y(x_0) &= y_0 \\ y'(x_0) &= y'_0 \end{aligned}$$

To lahko prevedemo na sistem enačb prvega reda

$$\begin{aligned} y' &= p, & y(x_0) &= y_0, \\ p' &= f(x, y, p), & p(x_0) &= y'_0. \end{aligned}$$

V posebnem primeru, ko je $y'' = f(x, y)$, obstajajo posebne R-K ali večkoračne metode, kot npr. *metoda Numerova*

$$y_{n+1} - 2y_n + y_{n-1} = \frac{h^2}{12}(f_{n+1} + 10f_n + f_{n-1}) + \mathcal{O}(h^6).$$

12.8 Robni problemi drugega reda

Rešujemo

$$\begin{aligned} y'' &= f(x, y, y') \\ y(a) &= \alpha \\ y(b) &= \beta \end{aligned}$$

Najpreprostejši je *linearni robni problem drugega reda*

$$-y''(x) + p(x)y'(x) + q(x)y(x) = r(x), \tag{12.22}$$

$$y(a) = \alpha, \quad y(b) = \beta.$$

Numerični načini reševanja linearnega robnega problema so:

a) kombinacija dveh začetnih problemov

Izberemo $\xi_1 \neq \xi_2$ in rešimo začetna problema, ki ustrezata enačbi (12.22) in

$$\begin{aligned} y_1(a) &= \alpha, \quad y_1'(a) = \xi_1, \\ y_2(a) &= \alpha, \quad y_2'(a) = \xi_2. \end{aligned}$$

Za poljuben λ je $y = \lambda y_1 + (1-\lambda)y_2$ tudi rešitev enačbe (12.22) in ustreza pogoju $y(a) = \alpha$, Sedaj λ določimo tako, da bo $y(b) = \beta$. Veljati mora

$$\lambda y_1(b) + (1-\lambda)y_2(b) = \beta,$$

torej

$$\lambda = \frac{\beta - y_2(b)}{y_1(b) - y_2(b)}.$$

b) *diferenčna metoda*

Interval $[a,b]$ ekvidistantno razdelimo na $n+1$ delov s točkami $x_0 = a, x_1, \ldots, x_n, x_{n+1} = b$, kjer je $x_i = x_0 + ih$ in $h = \frac{b-a}{n+1}$. Odvode aproksimiramo s simetričnimi diferencami

$$\begin{aligned} y_i' &= \frac{y_{i+1} - y_{i-1}}{2h} + \mathcal{O}(h^2), \\ y_i'' &= \frac{y_{i+1} - 2y_i + y_{i-1}}{h^2} + \mathcal{O}(h^2). \end{aligned}$$

Dobimo sistem enačb

$$\frac{-y_{i+1} + 2y_i - y_{i-1}}{h^2} - p_i \frac{y_{i+1} - y_{i-1}}{2h} + q_i y_i = r_i, \quad i = 1, \ldots, n,$$

pri čemer je $y_0 = \alpha$ in $y_{n+1} = \beta$. Sistem je linearen in tridiagonalen, zato ga lahko rešimo na preprost način.

Diferenčno metodo lahko uporabljamo tudi pri robnih pogojih, v katerih nastopajo tudi odvodi (zgled na vajah).

Če v diferencialni enačbi ne nastopa y', lahko uporabimo metodo Numerova, saj je potem napaka reda $\mathcal{O}(h^6)$ namesto $\mathcal{O}(h^2)$. V primeru

$$-y''(x) + q(x)y(x) = r(x)$$

tako dobimo

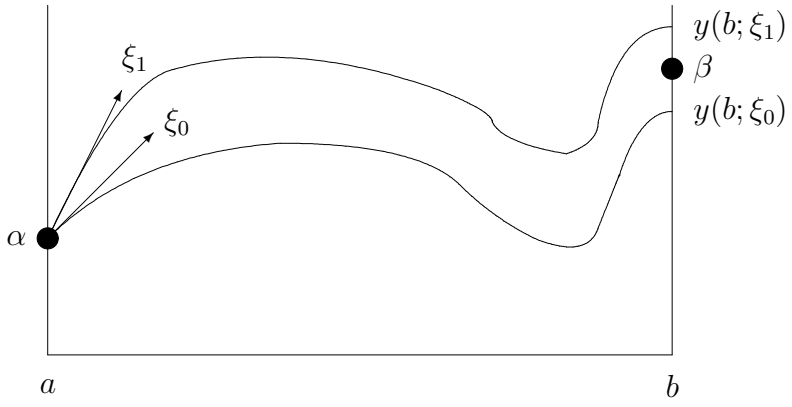
$$y_{i+1} - 2y_i + y_{i-1} = \frac{h^2}{12}(q_{n+1}y_{n+1} - r_{n+1} + 10q_n y_n - r_n - q_{n-1}y_{n-1} + r_{n-1}), \quad i = 1, \ldots, n.$$

Pri *nelinearnem robnem problemu drugega reda* je f nelinearna funkcija y in y'. Zaradi nelinearnosti ne moremo uporabiti kombinacije dveh začetnih problemov. Uporabimo lahko diferenčno metodo, a dobimo nelinearni sistem z veliko neznankami.

Na voljo pa imamo *strelsko metodo*. Pri strelski metodi rešujemo začetni problem

$$\begin{aligned} y'' &= f(x, y, y'), \\ y(a) &= \alpha, \\ y'(a) &= \xi. \end{aligned} \tag{12.23}$$

Rešitev je odvisna od ξ, parameter ξ pa je potrebno izbrati tako, da bo $y(b;\xi) = \beta$.



Če definiramo $E(\xi) := y(b;\xi) - \beta$, potem iščemo tak ξ, da bo $E(\xi) = 0$. Uporabimo lahko katerokoli metodo za reševanje nelinearne enačbe. Če npr. želimo uporabiti tangentno metodo, potem potrebujemo tudi $E'(\xi) = \frac{\partial y(b;\xi)}{\partial \xi}$. Definiramo $z := \frac{\partial y}{\partial \xi}$ in z odvajanjem

$$y'' = f(x, y, y'), \quad y(a) = \alpha,\ y'(a) = \xi$$

dobimo

$$z'' = f_y(x, y, y')z + f_{y'}(x, y, y')z', \quad z(a) = 0,\ z'(a) = 1. \tag{12.24}$$

Če rešujemo sistem (12.23) in (12.24), potem dobimo tudi vrednost odvoda funkcije E.