Úvod do numerické matematiky

učební text k přednášce NMNM211

Vladimír Janovský

Obsah

Kapitola 1

Řešení soustav lineárních rovnic: eliminační techniky

Úloha je prostá: Je dána matice $A \in \mathbb{R}^{n\times n}$ a vektor $b \in \mathbb{R}^n$ (data úlohy). Cílem je najít vektor $x \in \mathbb{R}^n$ tak, aby

$$Ax = b\,. \tag{1.1}$$

Je dobře známo, že

Věta 1.1. *Nechť $A \in \mathbb{R}^{n\times n}$, $b \in \mathbb{R}^n$. Jestliže $\det(A) \neq 0$, potom existuje právě jediné řešení $x \in \mathbb{R}^n$ úlohy* (1.1).

Máme existenční větu. Zbývá jenom řešení $x \in \mathbb{R}^n$ "vypočítat".

1.1 Gaussova eliminace (idea)

Nechť $A = \begin{bmatrix} 2 & 5 \\ 3 & 7 \end{bmatrix}$, $b = \begin{bmatrix} 1 \\ 2 \end{bmatrix}$. Hledáme $x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$ tak, aby

$$\begin{aligned} 2x_1 + 5x_2 &= 1\,, & (1.2)\\ 3x_1 + 7x_2 &= 2\,. & (1.3) \end{aligned}$$

Gaussova eliminační idea: Předpokládejme, že složku x_2 známe. Chápeme ji jako parametr. Na základě x_2 lze z rovnice (1.2) vypočítat složku x_1:

$$x_1 = \frac{1}{2}\left(1 - 5x_2\right)\,.$$

Z rovnice (1.3) dostaneme podmínku pro x_2:

$$3\left(\frac{1}{2}\left(1 + 5x_2\right)\right) + 7x_2 = 2\,.$$

Tedy $\left(7 - \frac{3}{2}5\right) x_2 = 2 - \frac{2}{3}1$, $-\frac{1}{2}\, x_2 = \frac{1}{2}$ a tedy $x_2 = -1$. Teď je možno určit složku x_1: Z (1.2) plyne, že

$$x_1 = \frac{1}{2}\left(1 - 5x_2\right) = \frac{1}{2}\left(1 - 5\left(-1\right)\right) = 3\,.$$

Eliminační idea má dobře známou interpretaci: Hledáme *ekvivalentní úpravy* soustavy (1.2)&(1.3). První rovnici necháme beze změny, druhou rovnici nahradíme vhodnou lineární kombinací (1.2) a (1.3). Konkrétně, první rovnici vynásobíme faktorem $\frac{3}{2}$ a odečteme ji od druhé rovnice. Symbolicky, (1.3) – $\frac{3}{2}$ (1.2). Po této ekvivalentní úpravě dostaneme soustavu

$$\begin{aligned} 2x_1 + 5x_2 &= 1\,, && (1.4)\\ -\frac{1}{2}x_2 &= \frac{1}{2}\,. && (1.5) \end{aligned}$$

Podstatné je, rovnice (1.2)&(1.3) a (1.4)&(1.5) mají **stejné** řešení $x \in \mathbb{R}^2$, viz *ekvivalentní úpravy*. Označme U a z matici soustavy a vektor pravé strany soustavy (1.4)&(1.5): $U = \begin{bmatrix} 2 & 5 \\ 0 & -\frac{1}{2} \end{bmatrix}$, $z = \begin{bmatrix} 1 \\ \frac{1}{2} \end{bmatrix}$. Matice U je *horní trojúhelníková*. Příslušnou soustavu rovnic lze snadno řešit algoritmem, kterému se říká *zpětná substituce*: Z (1.5) vypočítáme x_2. Z (1.4) potom dopočítáme x_1.

Řešení x jsme získali ve dvou krocích:

- Gaussovou eliminací jsme původní úlohu $Ax = b$ převedli na ekvivalentní úlohu $Ux = z$, kde U je horní trojúhelníková matice.
- Řešení úlohy $Ux = z$ jsme efektivně získali zpětnou substitucí.

1.2 LU-rozklad matice

Uvažujme problém (1.1). Řešíme tedy soustavu lineárních rovnic

$$\begin{array}{lcllcl} a_{11}x_1 & + & a_{12}x_2 & +\cdots+ & a_{1n}x_n & = & b_1 \\ a_{21}x_1 & + & a_{22}x_2 & +\cdots+ & a_{2n}x_n & = & b_2 \\ \vdots & & \vdots & & \vdots & & \vdots \\ a_{n1}x_1 & + & a_{n2}x_2 & +\cdots+ & a_{nn}x_n & = & b_n \end{array} \qquad (1.6)$$

Eliminujme proměnnou x_1 z i-té rovnice, $i \in \{2, \ldots, n\}$: Nechť $a_{11} \neq 0$. Od i-té rovnice odečteme vhodný násobek první rovnice. Tento násobek je zřejmě $l_{i1} \equiv \frac{a_{i1}}{a_{11}}$:

$$\underbrace{\left(a_{i1} - l_{i1}a_{11}\right)}_{=0} x_1 + \underbrace{\left(a_{i2} - l_{i1}a_{12}\right)}_{=a'_{i2}} x_2 + \cdots + \underbrace{\left(a_{in} - l_{i1}a_{1n}\right)}_{=a'_{in}} x_n = \underbrace{b_i - l_{i1}b_1}_{=b'_i}\,.$$

Dostaneme ekvivalentní soustavu

$$\begin{array}{rcccccl} a_{11}x_1 & + & a_{12}x_2 & +\cdots+ & a_{1n}x_n & = & b_1 \\ & & a'_{22}x_2 & +\cdots+ & a'_{2n}x_n & = & b'_2 \\ & & \vdots & & & & \vdots \\ & & a'_{n2}x_2 & +\cdots+ & a'_{nn}x_n & = & b'_n \end{array} \tag{1.7}$$

Budeme postupovat rekurzí. Zavedeme označení: $A^{(1)} = A$, $b^{(1)} = b$. Nechť $A^{(2)}$ a $b^{(2)}$ je matice soustavy a pravá strana rovnice (1.7). Budeme konstruovat posloupnost matic

$$A = A^{(1)} \to A^{(2)} \to \cdots \to A^{(k)} \to A^{(k+1)} \to \cdots \to A^{(n)} = U \tag{1.8}$$

a pravých stran

$$b = b^{(1)} \to b^{(2)} \to \cdots \to b^{(k)} \to b^{(k+1)} \to \cdots \to b^{(n)} = z\,. \tag{1.9}$$

Přitom

$$A^{(k)} = \begin{bmatrix} a_{11}^{(1)} & a_{12}^{(1)} & \cdots & \cdots & \cdots & a_{1n}^{(1)} \\ & a_{22}^{(2)} & \cdots & \cdots & \cdots & a_{2n}^{(2)} \\ & & \ddots & & & \vdots \\ & & & a_{kk}^{(k)} & \cdots & a_{kn}^{(k)} \\ & & & \vdots & & \vdots \\ & & & a_{nk}^{(k)} & \cdots & a_{nn}^{(k)} \end{bmatrix}, \quad b^{(k)} = \begin{bmatrix} b_1^{(1)} \\ b_2^{(2)} \\ \vdots \\ b_k^{(k)} \\ \vdots \\ b_n^{(k)} \end{bmatrix}. \tag{1.10}$$

Prvku $a_{kk}^{(k)}$ matice $A^{(k)}$ říkáme k-tý *pivot*. Pokud všechny kroky rekurze proběhnou úspěšně, $A^{(n)}$ je horní trojúhelníková. Označíme ji U,

$$U = \begin{bmatrix} a_{11}^{(1)} & a_{12}^{(1)} & \cdots & \cdots & \cdots & a_{1n}^{(1)} \\ & a_{22}^{(2)} & \cdots & \cdots & \cdots & a_{2n}^{(2)} \\ & & \ddots & & & \vdots \\ & & & a_{kk}^{(k)} & \cdots & a_{kn}^{(k)} \\ & & & & \ddots & \vdots \\ & & & & & a_{nn}^{(k)} \end{bmatrix}. \tag{1.11}$$

Popišme rekurzi $A^{(k)} \to A^{(k+1)}$, $b^{(k)} \to b^{(k+1)}$. Nechť $a_{kk}^{(k)} \neq 0$. Cílem je eliminace proměnné x_k z i-té rovnice, $i \in \{k+1, \ldots, n\}$.

Rovnici $A^{(k)}x = b^{(k)}$ lze redukovat na proměnné $x_k, x_{k+1}, \ldots, x_n$, kterých se bude eliminace týkat:

$$\begin{bmatrix} a_{k,k}^{(k)} & a_{k,k+1}^{(k)} & \cdots & a_{1n}^{(k)} \\ a_{k+1,k}^{(k)} & a_{k+1,k+1}^{(k)} & \cdots & a_{2n}^{(k)} \\ \vdots & \vdots & & \vdots \\ a_{n,k}^{(k)} & a_{n,k+1}^{(k)} & \cdots & a_{nn}^{(k)} \end{bmatrix} \begin{bmatrix} x_k \\ x_{k+1} \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} b_k^{(k)} \\ b_{k+1}^{(k)} \\ \vdots \\ b_n^{(k)} \end{bmatrix} \tag{1.12}$$

Po eliminaci proměnné x_k z i-té rovnice, $i \in \{k+1, \ldots, n\}$, soustavy (1.12) dostaneme soustavu

$$\begin{bmatrix} a_{k,k}^{(k)} & a_{k,k+1}^{(k)} & \cdots & a_{1n}^{(k)} \\ & a_{k+1,k+1}^{(k+1)} & \cdots & a_{2n}^{(k+1)} \\ & \vdots & & \vdots \\ & a_{n,k+1}^{(k+1)} & \cdots & a_{nn}^{(k+1)} \end{bmatrix} \begin{bmatrix} x_k \\ x_{k+1} \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} b_k^{(k)} \\ b_{k+1}^{(k+1)} \\ \vdots \\ b_n^{(k+1)} \end{bmatrix}, \tag{1.13}$$

kde

$$\begin{aligned} l_{ik} &= a_{ik}^{(k)} / a_{kk}^{(k)} \quad \text{pro} \quad i = k+1, \ldots, n \\ a_{ij}^{(k+1)} &= a_{ij}^{(k)} - l_{ik} a_{kj}^{(k)} \quad \text{pro} \quad i, j = k+1, \ldots, n \\ b_i^{(k+1)} &= b_i^{(k)} - l_{ik} b_k^{(k)} \quad \text{pro} \quad i = k+1, \ldots, n \end{aligned} \tag{1.14}$$

Matici $A^{(k+1)}$ lze zíslat z $A^{(k)}$ náhradou matice (1.12) maticí soustavy (1.13). Podobně lze najít b^{k+1}.

Lze tedy shrnout, že pokud v každém eliminačním kroku bude $a_{kk}^k \neq 0$, $k \in \{1, \ldots, n\}$, potom lze úlohu (1.1) převést na ekvivalentni soustavu $Ux = z$ postupem (1.8)&(1.9). Soustavu s horní trojúhelníkovou maticí U lze efektivně řešit zpětnou substitucí.

Gaussovu eliminaci, t.j. algoritmus (1.8)&(1.9), lze velmi užitečně interpretovat jako jistý maticový *rozklad*. Přitom tento rozklad nás nic nebude stát, jenom využijeme "odpadků" z Gaussovy eliminace: Jestliže $a_{kk}^{(k)} \neq 0$, potom eliminační krok $A^{(k)} \to A^{(k+1)}$ lze chápat jako lineární transformaci

$$A^{(k+1)} = M_k \, A^{(k)}, \tag{1.15}$$

kde

$$M_k = \begin{bmatrix} 1 & & & & \\ & \ddots & & & \\ & & 1 & & \\ & & -l_{k+1,k} & 1 & \\ & & \vdots & & \ddots \\ & & -l_{n,k} & & 1 \end{bmatrix}, \tag{1.16}$$

přičemž koeficienty $l_{k+1,k}, \ldots, l_{n,k}$ jsou definovány formulí (1.14). Jsou tedy "vedlejším produktem" jednoho eliminačního kroku. Matici M_k si lze představit jako identitu plus poruchu (s hodností 1):

$$M_k = I_n - \mathbf{m}_k \, \mathbf{e}_k^T, \quad \mathbf{m}_k \equiv (0, \ldots, 0, l_{k+1,k}, \ldots, l_{n,k})^T \in \mathbb{R}^n,$$

$$\begin{matrix} \uparrow \\ k-\text{tá} \\ \textit{pozice} \end{matrix}$$

kde $I_n \in \mathbb{R}^{n\times n}$ je identita a $\mathbf{e}_k \in \mathbb{R}^n$ je jednotkový vektor,

$$\mathbf{e}_k \equiv (0,\dots,0,1,0,\dots,0)^T \qquad (1.17)$$

$$\uparrow$$
$$k-\text{tá}$$
$$\textit{pozice}$$

Matici M_k se říká (k-tá) *Gaussova transformační matice* nebo také *Frobeniova matice*.

Pokud eliminační postup neselže, t.j. pokud $a_{kk}^k \neq 0$, $k \in \{1,\dots,n\}$, potom přechod (1.8) od A k U je možno interpretovat jako složenou transformaci

$$U = M_{n-1}...M_2\, M_1\, A\,.$$

Matice M_k jsou zřejmě regulární. Proto

$$M_1^{-1}\, M_2^{-1}\, \dots\, M_{n-1}^{-1}\, U = A\,.$$

Lemma 1.1. *Pro každé* $k \in \{1,\dots,n\}$,

$$M_k^{-1} = \begin{bmatrix} 1 & & & & \\ & \ddots & & & \\ & & 1 & & \\ & & l_{k+1,k} & 1 & \\ & & \vdots & & \ddots \\ & & l_{n,k} & & 1 \end{bmatrix} = I_n + \mathbf{m}_k\, \mathbf{e}_k^T\,.$$

Důkaz je přenechán jako cvičení. Tedy,

$$\begin{aligned} &M_1^{-1}\, M_2^{-1}\, \dots\, M_{n-1}^{-1} = \\ &= \left(I_n + \mathbf{m}_1\, \mathbf{e}_1^T\right)\left(I_n + \mathbf{m}_2\, \mathbf{e}_2^T\right)\, \dots\, \left(I_n + \mathbf{m}_{n-1}\, \mathbf{e}_{n-1}^T\right) = \\ &= I_n + \sum_{i=1}^{n-1} \mathbf{m}_k\, \mathbf{e}_k^T = L\,, \end{aligned}$$

kde

$$L = \begin{bmatrix} 1 & & & & \\ l_{21} & 1 & & & \\ l_{31} & l_{32} & 1 & & \\ \vdots & & \ddots & \ddots & \\ l_{n1} & l_{n2} & \cdots & l_{n,n-1} & 1 \end{bmatrix}. \qquad (1.18)$$

Věta 1.2. *Pokud* $a_{kk}^{(k)} \neq 0$, $k = 1,\dots,n$, *potom*

$$A = L\, U\,,$$

kde L *a* U *jsou horní a dolní trojúhelníkové matice, definované v* (1.18) *a* (1.11).

Algoritmus 1.1. (Gaussova eliminace)

1. $A = L\,U$ ... *najdeme LU-rozklad matice* A
2. *?* z: $L\,z = b$ ... *substituce vpřed*
3. *?* x: $U\,x = z$ ... *zpětná substituce*

Kalkulace ceny výpočtu: Odhadneme počty operací v jednotlivých fázích výpočtu (t.j. LU-rozklad, substituce vpřed, zpětná substituce).

Začněme s LU-rozkladem. Vyjdeme z formulí (1.13): Vyčíslení matice $A^{(2)}$ "stojí" $(n-1)^2$ násobení, $(n-1)^2$ odčítání a $(n-1)$ dělení. Vyčíslení matice $A^{(n)}$ "stojí" $(n-(n-1))^2$ násobení, $(n-(n-1))^2$ odčítání a $(n-(n-1))$ dělení. Nebudeme rozlišovat časovou náročnost jednotlivých typů operací. Za každou operaci si prostě uděláme čárku a počet čárek sečteme. Dospějeme k tomuto závěru:

$$\#\text{flops} = 2\sum_{k=1}^{n-1} k^2 + \sum_{k=1}^{n-1} k\,.$$

Symbol # znamená *počet prvků*. Počítáme *operace v pohyblivé řádové čárce*, t.j. *floating point operations*, zkratka flops. Uvedené řady se dají sečíst,

$$\#\text{flops} = 2(n-1)n(2n-1)/6 + n(n-1)/2 = \frac{2}{3}n^3 - n^2/2 + O(n)\,.$$

Cena substituce vpřed a cena zpětné substituce se odhaduje analogicky: Počet operací při substituci vpřed resp. při zpětné substituci lze odhadnout řádově stejně

$$\#\text{flops} = n^2 + O(n)\,.$$

LU-rozklad matice soustavy je nejnáročnější krok algoritmu. Například, pokud $n = 100$, potom LU-rozklad representuje zhruba $\#\text{flops} = \frac{2}{3}10^6$, zatímco obě substituce "stojí" marginálních $\#\text{flops} = 2 * 10^4$ Jestliže řešíme soustavu s více pravými stranami, potom LU-rozklad se počítá jenom jednou. Substituce vpřed resp. zpětná substituce se modifikují pro různé pravé strany.

Poznámka 1.1. *Pokud* $a_{kk}^{(k)} \neq 0$, $k = 1, \ldots, n$, *potom matice* L *a* U *z* LU*-rozkladu jsou určeny jednoznačně. Jsou jednoznačně definovány formulemi* (1.18) *a* (1.11)

Na LU-rozklad lze pohlížet přímo z definice maticového součinu. Začněme trochu obecněji:

Poznámka 1.2. *Nechť* $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times n}$, $C \in \mathbb{R}^{n\times n}$. *Prvky matic* A, B, C *označujme* $[a_{ij}]$, $[b_{ij}]$, $[c_{ij}]$, *kde první index* $i = 1, \ldots, n$ *je řádkový index a druhý index* $j = 1, \ldots, n$ *je sloupcový index. Jestliže* $A = B\,C$, *potom z definice součinu platí*

$$a_{ij} = \sum_{k=1}^{n} b_{ik}c_{kj} \tag{1.19}$$

pro každé $i = 1, \ldots, n$ *a* $j = 1, \ldots, n$. *Vrátíme se k LU-rozkladu. Nechť* $B = L$, *viz* (1.18) *a* $C = U$, *viz* (1.11). *Určujeme* $(n-1)n/2$ *neznámých prvků matice* L *a* $n(n+1)/2$ *neznámých prvků matice* U. *Neznámé prvky těchto matic musí splňovat* n^2 *podmínek* (1.19). *Tedy počet podmínek je roven počtu neznámých. Vzniklá soustava rovnic pro neznámé prvky matic* L *a* U, *je za jistých předpokladů řešitelná. Naznačená myšlenka LU-rozkladu je základem Croutova algoritmu, viz např. [3], str. 78.*

1.3 Sloupcová pivotace

Začneme jednoduchým příkladem:

$$\begin{bmatrix} 0 & 1 \\ 3 & 7 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 3 \\ 1 \end{bmatrix}$$

Je vidět, že z první rovnice nelze eliminovat proměnnou x_1, první pivot je nulový. Pomůže ale ekvivalentní úprava: přehození rovnic,

$$\begin{bmatrix} 3 & 7 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 1 \\ 3 \end{bmatrix}.$$

Toto přehození lze interpretovat jako násobení *permutační maticí*:

$$\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} 0 & 1 \\ 3 & 7 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} 3 \\ 1 \end{bmatrix}$$

Tento postup lze zobecnit: Uvažujme Gaussovu eliminaci (1.8)&(1.9). Nechť existuje $k \in \{1, \ldots, n-1\}$ tak, že $a_{ii}^{(i)} \neq 0$ pro $i = 1, \ldots, k-1$. Nechť k-tý pivot je nulový, $a_{kk}^{(k)} = 0$. Nabízí se strategie výběru nového pivota: Nechť existuje index $j \in \{k+1, \ldots, n\}$ tak, že $a_{jj}^{(j)} \neq 0$. Potom přehodíme j-tou a k-tou rovnici, a eliminujeme x_j. Popišme eliminační krok $A^{(k)} \to A^{(k+1)}$, $b^{(k)} \to b^{(k+1)}$ operátorově. Definujme *matici transpozice* T_{kj}, $k < j$,

$$
T_{kj} = \begin{array}{c}
\begin{array}{ccccccc} & & k-\text{tý} & & j-\text{tý} & & \\ & & \textit{sloupec} & & \textit{sloupec} & & \\ & & \downarrow & & \downarrow & & \end{array} \\
\begin{bmatrix}
1 & & & & & & \\
& \ddots & & & & & \\
& & 0 & & 1 & & \\
& & & \ddots & & & \\
& & 1 & & 0 & & \\
& & & & & \ddots & \\
& & & & & & 1
\end{bmatrix}
\end{array}
\begin{array}{ll} & \\ & \\ \leftarrow & k-\text{tý řádek} \\ & \\ \leftarrow & j-\text{tý řádek} \\ & \\ & \end{array}
\tag{1.20}
$$

V každém řádku a každém sloupci matice T_{kj} je právě jedna jednička, jinak samé nuly. Potom

$$A^{(k+1)} = M_k\, T_{kj}\, A^{(k)}\,, \quad b^{(k+1)} = M_k\, T_{kj}\, b^{(k)}\,;$$

M_k je příslušná Gaussova transformační matice, viz (1.16).

Jesliže neexistuje kandidát na nenulového pivota, t.j. $a_{jk}^{(j)} = 0$ pro každé $j = k+1, \ldots, n$, potom zřejmě $\det A = 0$.

Užívá se lepší strategie, tzv. *sloupcová pivotace*: V k-tém eliminačním kroku volíme pivot, který je největší v absolutní hodnotě, t.j.

$$j = \arg \max_{i=k,\ldots,n} |a_{ik}^{(i)}|\,. \tag{1.21}$$

Index j je možno definovat jednoznačně, např. z více kandidátů zvolíme nejmenší j. Proto položme $T_k \equiv T_{kj}$, kde j splňuje (1.21) a definujme k-tý krok *Gaussovy eliminace se sloupcovou pivotací*:

$$A^{(k+1)} = M_k\, T_k\, A^{(k)}\,, \quad b^{(k+1)} = M_k\, T_k\, b^{(k)}\,. \tag{1.22}$$

Sloupcová pivotace zaručuje omezenost prvků matic M_k. Platí:

$$l_{ik} \equiv a_{ik}^{(k)} / a_{kk}^{(k)}\,, \quad |l_{ik}| \leq 1\,, \quad 1 \leq k < i \leq n\,. \tag{1.23}$$

Směřujeme k pojmu LU-rozklad s pivotací. Nechť $\det(A) \neq 0$. Původní soustava (1.1) je ekvivalentní soustavě s horní trojúheníkovou maticí U,

$$U = M_{n-1}\, T_{n-1}\, \ldots\, M_{k+1}\, T_{k+1}\, M_k\, T_k\, \ldots\, M_1\, T_1\, A\,.$$

Definujme matici

$$P \equiv T_{n-1}\, \ldots\, T_{k+1}\, T_k\, \ldots\, T_1\,. \tag{1.24}$$

$P \in \mathbb{R}^{n\times n}$ je *permutační matice*, jejíž prvky jsou nuly a jedničky. Přitom v každém řádku a každém sloupci je právě jedna jednička. Permutační matici P lze identifikovat s *permutací* π množiny n prvků,

$$P \longleftrightarrow \pi\,.$$

Speciálním případem permutačních matic jsou matice transpozice (1.20). T_k, viz (1.24), lze identifikovat s *transpozicí* τ_k uvažované množiny n prvků. Proto

$$P \longleftrightarrow \pi = \tau_{n-1} \circ \ \ldots \circ \tau_{k+1} \circ \tau_k \circ \ldots \circ \tau_1 \,.$$

Z "odpadu" Gaussovy eliminace se sloupcovou pivotací sestavme matici

$$\widetilde{L} = \begin{bmatrix} 1 & & & & \\ l_{21} & 1 & & & \\ l_{31} & l_{32} & 1 & & \\ \vdots & & \ddots & \ddots & \\ l_{n1} & l_{n2} & \cdots & l_{n,n-1} & 1 \end{bmatrix}, \tag{1.25}$$

jejíž prvky l_{ik} jsou definovány v (1.23). Na základě $\widetilde{L}$ a rozkladu (1.24) lze zkonstruovat tzv. LU-rozklad s pivotací. Ukážeme si to na příkladě:

Příklad 1.1. *Nechť*

$$A = A^{(1)} = \begin{bmatrix} \frac{1}{2} & -1 & 0 & 0 \\ 1 & -1 & 3 & 0 \\ 0 & -2 & 6 & -3 \\ 0 & 0 & 4 & -7 \end{bmatrix}$$

Eliminace s pivotací:

$$A^{(2)} = M_1\, T_1\, A^{(1)} = M_1 \begin{bmatrix} 1 & -1 & 3 & 0 \\ \frac{1}{2} & -1 & 0 & 0 \\ 0 & -2 & 6 & -3 \\ 0 & 0 & 4 & -7 \end{bmatrix} = \begin{bmatrix} 1 & -1 & 3 & 0 \\ 0 & -\frac{1}{2} & -\frac{3}{2} & 0 \\ 0 & -2 & 6 & -3 \\ 0 & 0 & 4 & -7 \end{bmatrix}$$

$$A^{(3)} = M_2\, T_2\, A^{(2)} = M_2 \begin{bmatrix} 1 & -1 & 3 & 0 \\ 0 & -2 & 6 & -3 \\ 0 & -\frac{1}{2} & -\frac{3}{2} & 0 \\ 0 & 0 & 4 & -7 \end{bmatrix} = \begin{bmatrix} 1 & -1 & 3 & 0 \\ 0 & -2 & 6 & -3 \\ 0 & 0 & -3 & \frac{3}{4} \\ 0 & 0 & 4 & -7 \end{bmatrix}$$

$$A^{(4)} = M_3\, T_3\, A^{(3)} = M_3 \begin{bmatrix} 1 & -1 & 3 & 0 \\ 0 & -2 & 6 & -3 \\ 0 & 0 & 4 & -7 \\ 0 & 0 & -3 & \frac{3}{4} \end{bmatrix} = \begin{bmatrix} 1 & -1 & 3 & 0 \\ 0 & -2 & 6 & -3 \\ 0 & 0 & 4 & -7 \\ 0 & 0 & 0 & -\frac{9}{2} \end{bmatrix}$$

Symbolicky:

$$A = A^{(1)} \to A^{(2)} \to A^{(3)} \to A^{(4)} = U \,,$$

kde

$$U = M_3\, T_3\ M_2\, T_2\ M_1\, T_1\ A \,. \tag{1.26}$$

Přitom

$$T_1 = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad T_2 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad T_3 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}.$$

jsou matice transpozice a

$$M_1 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ -l_{21} & 1 & 0 & 0 \\ -l_{31} & 0 & 1 & 0 \\ -l_{41} & 0 & 0 & 1 \end{bmatrix}, \quad M_2 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & -l_{32} & 1 & 0 \\ 0 & -l_{42} & 0 & 1 \end{bmatrix}, \quad M_3 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & -l_{43} & 1 \end{bmatrix},$$

jsou Gaussovy transformační matice (Frobeniovy matice), přičemž

$$\widetilde{L} \equiv \begin{bmatrix} 1 & 0 & 0 & 0 \\ l_{21} & 1 & 0 & 0 \\ l_{31} & l_{31} & 1 & 0 \\ l_{41} & l_{32} & l_{43} & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ \frac{1}{2} & 1 & 0 & 0 \\ 0 & \frac{1}{4} & 1 & 0 \\ 0 & 0 & -\frac{3}{4} & 1 \end{bmatrix} = M_1^{-1}\, M_2^{-1}\, M_3^{-1}\,.$$

Směřujeme k rozkladu matice A: Z (1.26)

$$U = M_3\; T_3\, M_2\, T_3\;\; T_3\, T_2\;\; M_1\, T_1\;\; A = M_3\;\; \widetilde{M_2}\;\; T_3\, T_2\;\; M_1\, T_1\;\; A\,,$$

kde

$$\widetilde{M_2} \equiv T_3\, M_2\, T_3\,.$$

Z předchozí identity

$$U = M_3\;\; \widetilde{M_2}\;\; T_3\, T_2\, M_1\, T_2\, T_3\;\; T_3\, T_2\, T_1\;\; A = M_3\, \widetilde{M_2}\, \widetilde{M_1}\;\; T_3\, T_2\, T_1\;\; A\,,$$

kde

$$\widetilde{M_1} \equiv T_3\, T_2\, M_1\, T_2\, T_3\,.$$

Proto

$$\widetilde{M_1}^{-1}\, \widetilde{M_2}^{-1}\, M_3^{-1}\; U = T_3\, T_2\, T_1\;\; A\,,$$

kde

$$\widetilde{M_1}^{-1} = T_3\, T_2\, M_1^{-1}\, T_2\, T_3\,, \quad \widetilde{M_2}^{-1} = T_3\, M_2^{-1}\, T_3\,.$$

Položme

$$P \equiv T_3\, T_2\, T_1\,. \tag{1.27}$$

Analyzujme $\widetilde{M_1}^{-1}$:

$$\widetilde{M_1}^{-1} = I + T_3\, T_2 \begin{bmatrix} 0 & 0 & 0 & 0 \\ l_{21} & 0 & 0 & 0 \\ l_{31} & 0 & 0 & 0 \\ l_{41} & 0 & 0 & 0 \end{bmatrix} T_2\, T_3 = I + T_3\, T_2 \begin{bmatrix} 0 & 0 & 0 & 0 \\ l_{21} & 0 & 0 & 0 \\ l_{31} & 0 & 0 & 0 \\ l_{41} & 0 & 0 & 0 \end{bmatrix}.$$

Podobně,

$$\widetilde{M}_2^{-1} = I + T_3 \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & l_{32} & 0 & 0 \\ 0 & l_{42} & 0 & 0 \end{bmatrix} T_3 = I + T_3 \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & l_{32} & 0 & 0 \\ 0 & l_{42} & 0 & 0 \end{bmatrix}.$$

Položme

$$\begin{bmatrix} 1 \\ \widetilde{l}_{21} \\ \widetilde{l}_{31} \\ \widetilde{l}_{41} \end{bmatrix} \equiv T_3\, T_2 \begin{bmatrix} 1 \\ l_{21} \\ l_{31} \\ l_{41} \end{bmatrix}, \quad \begin{bmatrix} 0 \\ 1 \\ \widetilde{l}_{32} \\ \widetilde{l}_{42} \end{bmatrix} \equiv T_3 \begin{bmatrix} 0 \\ 1 \\ l_{32} \\ l_{42} \end{bmatrix}, \quad \begin{bmatrix} 0 \\ 0 \\ 1 \\ \widetilde{l}_{43} \end{bmatrix} \equiv \begin{bmatrix} 0 \\ 0 \\ 1 \\ l_{43} \end{bmatrix}.$$

Potom,

$$\widetilde{M}_1^{-1}\, \widetilde{M}_2^{-1}\, M_3^{-1} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ \widetilde{l}_{21} & 1 & 0 & 0 \\ \widetilde{l}_{31} & \widetilde{l}_{32} & 1 & 0 \\ \widetilde{l}_{41} & \widetilde{l}_{42} & \widetilde{l}_{43} & 1 \end{bmatrix} \equiv L \tag{1.28}$$

Lze tedy shrnout, že

$$P\, A = L\, U\,, \tag{1.29}$$

kde

$$P = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{bmatrix}, \quad L = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ \frac{1}{2} & \frac{1}{4} & -\frac{3}{4} & 1 \end{bmatrix}, \quad U = \begin{bmatrix} 1 & -1 & 3 & 0 \\ 0 & -2 & 6 & -3 \\ 0 & 0 & 4 & -7 \\ 0 & 0 & 0 & -\frac{9}{2} \end{bmatrix}.$$

Platí:

Věta 1.3. *Nechť* $\det(A) \neq 0$. *Potom existuje permutační matice* $P \in \mathbb{R}^{n\times n}$ *tak, že*

$$P\, A = L\, U\,,$$

kde $U \in \mathbb{R}^{n\times n}$ *je horní trojúhelníková a* $L \in \mathbb{R}^{n\times n}$ *je dolní trojúhelníková s prvky* $L = (l_{ij})_{i,j=1,\ldots,n}$, *přičemž*

$$|l_{ij}| \leq 1 \text{ pro } 1 \leq j < i \leq 1\,, \quad l_{ii} = 1 \text{ pro } i = 1, \ldots, n\,.$$

Důkaz vynecháme. Lze jej najít např. v [1], Theorem 1.8, str. 10. Příklad 1.1 naznačuje, že k-tý sloupec matice L je definován na základě k-tého sloupce matice $\widetilde{L}$, viz (1.25):

$$L(:, k) \equiv T_{n-1} \ldots T_{k+1}\, \widetilde{L}(:, k)\,,\ k = 1, \ldots, n-2\,, \quad L(:, n-1) \equiv \widetilde{L}(:, n-1)\,.$$

V praktických implementacích algoritmu se nepoužívají ani matice transpozice T_k ani ekvivalentní transpozice τ_k. Pole, které zaujímá původní matice A, se v $n-1$ krocích přepisuje tak, že v posledním kroku se do pole zapíše horní trojúhelník matice U a striktní dolní trojúhelník matice L. V každém kroku se aktualizuje informace, která je ekvivalentní permutaci $\tau_{n-1} \circ \ldots \circ \tau_{k+1}$. Lze ji zapsat do jednorozměrného pole $\mathbb{N}^{n-1}$, které se v jednotlivých krocích aktualizuje (přepisuje).

Algoritmus 1.2. (Gaussova eliminace se sloupcovou pivotací)

1. $P\,A = L\,U$ *... najdeme LU-rozklad matice A se sloupcovou pivotací. Položme* $c = P\,b$.
2. *? z:* $\quad L\,z = c$ *... substituce vpřed*
3. *? x:* $\quad U\,x = z$ *... zpětná substituce*

Odhad "ceny" algoritmu počítaný v počtu operací #flops je řádově stejný jako u algoritmu 1.1. Transpozice T_k resp. τ_k představují přesuny v operační paměti a těch se počítadlo operací flops netýká. Ostatně, verze `MATLAB` 6 (a výše) počítadlo #flops "nepodporují", jakožto zastaralé měřítko efektivity algoritmu.

1.4 Choleského rozklad

Položíme si tuto otázku: Jak se projeví předpoklad symetrie matice (t.j. předpoklad $A = A^T$) na fungování algoritmu LU-rozkladu?

Nejprve uvažujme obecnou matici $A \in \mathbb{R}^{n\times n}$. Nechť existuje LU-rozklad matice A t.j. existuje L a U, viz (1.11) a (1.18), $a_{kk}^{(k)} \neq 0$ pro $k = 1, \ldots, n$. Potom existuje diagonální matice $D \in \mathbb{R}^{n\times n}$ a horní trojúhelníková matice $R \in \mathbb{R}^{n\times n}$, která má jedničky na diagonále, tak že platí

$$U \;=\; DR\,. \tag{1.30}$$

Toto tvrzení vyplývá z definice součinu matic: Prvky matice U označujme u_{ij}. U je horní trojúhelníková. Definujme

$$D = \begin{bmatrix} u_{11} & & & & & \\ & u_{22} & & & & \\ & & \ddots & & & \\ & & & u_{kk} & & \\ & & & & \ddots & \\ & & & & & u_{nn} \end{bmatrix}, \; R = \begin{bmatrix} R_{11} & R_{12} & \cdots & \cdots & \cdots & R_{1n} \\ & R_{22} & \cdots & \cdots & \cdots & R_{2n} \\ & & \ddots & & & \vdots \\ & & & R_{kk} & \cdots & R_{kn} \\ & & & & \ddots & \vdots \\ & & & & & R_{nn} \end{bmatrix},$$

kde $R_{ij} = \frac{u_{ij}}{u_{ii}}$. Z definice, $R_{ii} = 1$, $i = 1, \ldots, n$. Snadno se ověří, že platí (1.30), Dospěli jsme k následujícímu rozkladu matice A:

$$A \;=\; LDR\,. \tag{1.31}$$

Za předpokladu $a_{kk}^{(k)} \neq 0$ pro $k = 1, \ldots, n$ je tento rozklad jednoznačný.

Nechť A je *symetrická*, t.j. $A = A^T$. Platí-li (1.31), potom

$$A^T \;=\; (LDR)^T \;=\; (DR)^T L^T \;=\; R^T D^T L^T \;=\; R^T D L^T \,.$$

Matice R^T je dolní trojúhelníková, s jedničkami na diagonále, L^T je horní trojúhelníková, s jedničkami na diagonále, D je diagonální přičemž diagonální prvky splňují předpoklad $a_{kk}^{(k)} \neq 0$ pro $k = 1, \ldots, n$. Z jednoznačnosti rozkladu matice A^T, a ze symetrie $A = A^T$ vyplývá, že $R^T = L$. Ukázali jsme, že pokud matice A je symetrická, a pokud $a_{kk}^{(k)} \neq 0$ pro $k = 1, \ldots, n$, potom matici A lze rozložit takto:

$$A \;=\; LDL^T \,, \tag{1.32}$$

kde L je dolní trojúhelníková matice s jedničkami na diagonále, a D je diagonální matice. Diagonální prvky matice D označme d_k, $k = 1, \ldots, n$. Poznamenejme, že $d_k = a_{kk}^{(k)}$ je k-tý pivot eliminace, viz (1.10). Předpokládáme tedy, že $d_k \neq 0$ pro $k = 1, \ldots, n$.

Zaveďme operátor diagonální matice

$$D \;=\; \mathrm{diag}\,(d_1, ..., d_k, ..., d_n) \;. \tag{1.33}$$

Nechť všechny diagonální prvky jsou kladné, t.j. $d_k > 0$ pro $k = 1, \ldots, n$. Potom můžeme korektně definovat odmocninu matice $D^{1/2}$,

$$D^{1/2} \;=\; \mathrm{diag}\left(d_1^{1/2}, ..., d_k^{1/2}, ..., d_n^{1/2}\right) \;.$$

Rozklad (1.32) lze zapsat takto:

$$A \;=\; LDL^T \;=\; GG^T \,, \tag{1.34}$$

kde $G \equiv LD^{1/2}$. Matice G je dolní trojúhelníková a na diagonále má kladné prvky.

Rozkladu symetrické matice A na součin

$$A \;=\; GG^T \,, \tag{1.35}$$

kde G je dolní trojúhelníková s kladnými prvky diagonále, říkáme *Choleského rozklad*. Ukázali jsme, že za jistých předpokladů Choleského rozklad existuje.

Budeme specifikovat důležitou třídu matic, pro kterou Choleského rozklad existuje.

Definice 1.1. *Říkáme, že A je symetrická, positivně definitní (krátce: A je s.p.d.), jestliže*

1. $A = A^T$,
2. *kvadratická forma* $x^T Ax \geq 0$ *pro každý vektor* $x \in \mathbb{R}^n$,

3. jestliže existuje $x \in \mathbb{R}^n$ pro které forma $x^T Ax = 0$, potom $x = 0 \in \mathbb{R}^n$.

Následující postačující podmínku existence Choleského rozkladu vyslovíme bez důkazu:

Věta 1.4. *Jestliže A je s.p.d., potom existuje Choleského rozklad.*

Důkaz viz např. [1], Theorem 1.8, str. 10.

Zformulujeme algoritmus řešení naší základní úlohy (1.1) za předpokladu, že A je s.p.d. Budeme předpokládat, že existuje G, tak že (1.35). Tedy $Ax = GG^T x = b$. Idea je, provést substituci $G^T x = z$:

Algoritmus 1.3. (Řešení soustavy se symetrickou maticí)
Hledáme x: $Ax = b$, kde A je s.p.d.

1. $A = G\,G^T$ *... najdeme Choleského rozklad matice A*
2. *? z:* $G\,z = b$ *... substituce vpřed*
3. *? x:* $G^T\,x = z$ *... zpětná substituce*

Kalkulace ceny výpočtu se dělá analogicky jako při analýze Algoritmu 1.1. Ukazuje se, cena Choleského rozkladu je podstatná. Lze ji odhadnout jako

$$\#\text{flops} = \frac{1}{3}n^3 + O(n^2)\,.$$

Tedy zhruba poloviční, než cena LU-rozkladu. Náklady substituce vpřed a náklady zpětné substituce jsou marginální.

Zatím jsme se nezmínili, jak efektivně zvládnout Choleského rozklad. Víme, že *existuje* (viz Věta 1.4). Tento fakt využijeme při formulaci algoritmu rozkladu. Připomeňme Poznámku 1.2: Předpokládáme, že $A = B\,C$. V kontextu Choleského rozkladu, $B = G$ a $C = G^T$. Prvky matice G označujeme indexy $[g_{ij}]$. G je horní trojúhelníková, t.j. $g_{ij} = 0$ pro $i < j$. Z podmínky (1.19) proto dostáváme

$$a_{ij} = \sum_{k=1}^{n} g_{ik} g_{jk} = \sum_{k=1}^{\min(i,j)} g_{ik} g_{jk} \tag{1.36}$$

pro každé $i = 1, \ldots, n$ a $j = 1, \ldots, n$.

Algoritmus Choleského rozkladu si vysvětlíme na příkladu matice $A \in \mathbb{R}^{3\times 3}$. Předpokládáme, že

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} = \begin{bmatrix} g_{11} & 0 & 0 \\ g_{21} & g_{22} & 0 \\ g_{31} & g_{32} & g_{33} \end{bmatrix} \begin{bmatrix} g_{11} & g_{21} & g_{31} \\ 0 & g_{22} & g_{32} \\ 0 & 0 & g_{33} \end{bmatrix},$$

$a_{21} = a_{12}$, $a_{31} = a_{13}$, $a_{23} = a_{23}$. Hledáme devět neznámých g_{ij}, $3 \geq i \geq j \geq 1$, s tím, že musí být splněno devět požadavků (1.36), $3 \geq i \geq j \geq 1$. Požadavky lze splňit postupně (kanonicky):

1. $a_{11} = g_{11}g_{11} = g_{11}^2$: Vypočítáme $g_{11} = \pm\sqrt{a_{11}}$.
   Volíme kladný kořen, t.j. $g_{11} = \sqrt{a_{11}}$.

2. $a_{21} = g_{21}g_{11}$: Vypočítáme g_{21}.

3. $a_{22} = g_{21}g_{21} + g_{22}g_{22}$: Vypočítáme $g_{22} = \pm\sqrt{a_{22} - g_{21}^2}$.
   Volíme kladný kořen, t.j. $g_{22} = \sqrt{a_{22} - g_{21}^2}$.

4. $a_{31} = g_{31}g_{11}$: Vypočítáme g_{31}.

5. $a_{32} = g_{31}g_{21} + g_{32}g_{22}$: Vypočítáme g_{32}.

6. $a_{33} = g_{31}g_{31} + g_{32}g_{32} + g_{33}g_{33}$: Vypočítáme $g_{33} = \pm\sqrt{a_{33} - g_{31}^2 - g_{32}^2}$.
   Volíme kladný kořen, t.j. $g_{33} = \sqrt{a_{33} - g_{31}^2 - g_{32}^2}$.

Pokud Choleského rozklad existuje (např., pokud A je s.p.d.), potom transcendentní rovnice v krocích 1, 3 a 6 mají reálná, obecně iracionální, řešení g_{11}, g_{22} a g_{33}.

1.5 Analýza zaokrouhlovacích chyb

Kapitola 2

Metoda nejmenších čtverců

2.1 Motivace: Lineární regrese

Uvažujme m, $m \geq 2$, bodů (ξ_i, η_i), $i = 1, \ldots, m$, v rovině $\mathbb{R}^2$. Jsou to data následujícího problému: Hledáme koeficienty a_1, a_2 přímky $\eta = a_1\xi + a_2$ tak, aby přímka $\eta = a_1\xi + a_2$ optimálně prokládala (fitovala) zadaná data, viz Obr. 2.1.

Musíme definovat v jakém smyslu "optimálně": Definujme vektor $r \in \mathbb{R}^m$, $r = (r_i, \ldots, r_i, \ldots, r_m)$,

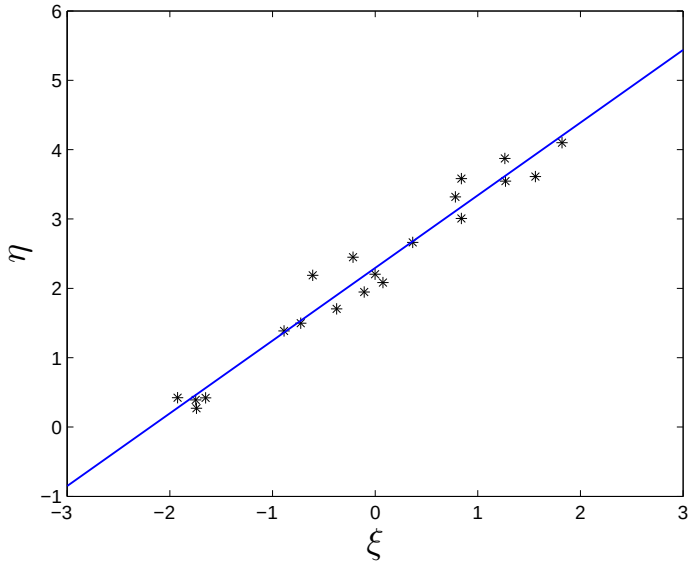
$$\begin{aligned} r_1 &= a_1\xi_1 + a_2 - \eta_1 \\ &\vdots \\ r_i &= a_1\xi_i + a_2 - \eta_i \\ &\vdots \\ r_m &= a_1\xi_m + a_2 - \eta_m \,. \end{aligned} \tag{2.1}$$

Složku vektoru r_i lze chápat jako *residuum* rovnice $a_1\xi_i + a_2 - \eta_i$. Soustavu (2.1) lze zapsat vektorově:

$$\begin{bmatrix} r_1 \\ \vdots \\ r_i \\ \vdots \\ r_m \end{bmatrix} = \begin{bmatrix} \xi_1 & 1 \\ \vdots & \vdots \\ \xi_i & 1 \\ \vdots & \vdots \\ \xi_m & 1 \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \end{bmatrix} - \begin{bmatrix} \eta_1 \\ \vdots \\ \eta_i \\ \vdots \\ \eta_m \end{bmatrix} . \tag{2.2}$$

Označme

$$A = \begin{bmatrix} \xi_1 & 1 \\ \vdots & \vdots \\ \xi_i & 1 \\ \vdots & \vdots \\ \xi_m & 1 \end{bmatrix}, \quad b = \begin{bmatrix} \eta_1 \\ \vdots \\ \eta_i \\ \vdots \\ \eta_m \end{bmatrix} .$$



Obrázek 2.1: Data: Dvacet bodů (ξ_i, η_i) v rovině $\mathbb{R}^2$. Lineární regrese: Aproximace dat přímkou $\eta = a_1\xi + a_2$, $a_1 = 1.0479$, $a_2 = 2.2944$.

Matice $A \in \mathbb{R}^{m\times 2}$ a vektor $b \in \mathbb{R}^m$ reprezentují *data* problému. Uvažujme zobrazení

$$\begin{bmatrix} a_1 \\ a_2 \end{bmatrix} \longmapsto r \equiv A \begin{bmatrix} a_1 \\ a_2 \end{bmatrix} - b\,.$$

Interpretace: Každému "pokusu" o nalezení a_1 a a_2 přiřadíme příslušné residuum $r \in \mathbb{R}^m$.

Cílem je najít takový pokus (takové a_1 a a_2), kdy příslušné residuum bude nejmenší. Reziduum $r \in \mathbb{R}^m$ měříme velikostí *normy*,

$$\|r\| \equiv \left(\sum_{i=1}^{m} r_i^2 \right)^{1/2} . \tag{2.3}$$

Uvedené normě se říká *Eukleidovská norma*. Pokus je vektor $y \in \mathbb{R}^2$ se složkami a_1 a a_2. Formulujeme optimalizační úlohu: Hledáme vektor $x \in \mathbb{R}^2$ tak, aby

$$x = \arg \min_{y\in\mathbb{R}^2} \|Ay - b\| \ . \tag{2.4}$$

Zajímá nás argument $x \in \mathbb{R}^2$ reálné funkce

$$y \longmapsto \|Ay - b\| \ ,$$

pro který se toto minimum nabývá. Tento argument $x \in \mathbb{R}^2$, jehož složky označíme $x \equiv (a_1, a_2)$, lze interpretovat jako koeficienty hledané přímky $\eta = a_1\xi + a_2$. Řešení úlohy (2.4) minimalizuje *součet čtverců* reziduií $\sum_{i=1}^{m} r_i^2$. Je totiž zřejmé, že

$$x = \arg \min_{y \in \mathbb{R}^2} \|Ay - b\| = \arg \min_{y \in \mathbb{R}^2} \|Ay - b\|^2 \ .$$

Poznámka 2.1. *Uvedený příklad ilustruje obecnější úlohu tzv.* regresní analýzy*: Cílem je aproximace zadaných (naměřených) hodnot nějakou funkcí z předepsaného prostoru. Jestliže je předepsaný vztah lineární (representovaný např. výše uvedenou maticí A), potom mluvíme o* lineární regresi*. Matematickou technikou lineární regrese je* metoda nejmenších čtverců*.*

2.2 Normálové rovnice

Motivační příklad zobecníme:

Problém 2.1. *Pro daná data*

$$A = \mathbb{R}^{m \times n}, m \geq n, \quad b \in \mathbb{R}^m \tag{2.5}$$

hledáme takový vektor $x \in \mathbb{R}^n$, aby

$$x \in \arg \min_{y \in \mathbb{R}^n} \|Ay - b\| \ . \tag{2.6}$$

Inkluze $\in$ naznačuje, že řešení nemusí být jednoznačné. (To by ostatně mohlo nastat i v případě naší motivační úlohy.)

Formálně řešíme úlohu

$$Ax = b, \tag{2.7}$$

tedy stejnou úlohu (1.1), jako v Kap. 1. Protože $m \geq n$, soustava rovnic (2.7) může mít víc rovnic než neznámých. Musíme proto formální úloze (2.7) dát rozumný smysl: Definujeme reziduum $r \in \mathbb{R}^m$ rovnice (2.7),

$$r = Ax - b.$$

Hledáme $x \in \mathbb{R}^n$ tak, aby součet čtverců složek reziduií r_i, $i = 1, \ldots, m$, tedy $\|r\|^2 \equiv \sum_{i=1}^{m} r_i^2$, byl minimální tedy, aby platilo (2.6). Říkáme, že x splňuje rovnici (2.7) ve smyslu *nejmenších čtverců*.

Věta 2.1. *Nechť $A = \mathbb{R}^{m \times n}$, $m \geq n$, $b \in \mathbb{R}^m$. Potom $x \in \mathbb{R}^n$ je splňuje* (2.6) *právě když $x \in \mathbb{R}^n$ je řešením*

$$A^{\mathrm{T}}Ax = A^{\mathrm{T}}b. \tag{2.8}$$

Poznamenejme, že $A^{\mathrm{T}}A \in \mathbb{R}^{n\times n}$ a $A^{\mathrm{T}}b \in \mathbb{R}^n$. Řešení problému (2.8) představuje řešení soustavy lineárních rovnic se čtercovou maticí, viz Kap. 1. Ledacos o numerickém řešení tohoto problému víme. Soustavě (2.8) se říká *normálové rovnice*.

K důkazu potřebujeme pomocné tvrzení:

Lemma 2.1. *Nechť N je lineární podprostor prostoru $\mathbb{R}^m$, nechť $b \in \mathbb{R}^m$. Potom dvě následující tvrzení* (2.9) *a* (2.10) *jsou ekvivalentní:*

$$\xi \in N \ : \quad ||\xi - b|| \le ||\eta - b|| \quad \forall \eta \in N\,. \tag{2.9}$$

$$\begin{aligned} \xi \in N \ : \quad & (\xi - b, \eta - \xi) = 0 \quad \forall \eta \in N\,, \\ \textit{t.j.} \qquad & (\xi - b, \eta) = 0 \quad \forall \eta \in N\,. \end{aligned} \tag{2.10}$$

Poznámka 2.2. *V tvrzení vystupuje* skalární součin*. Připomeňme jeho definici: Nechť $u \in \mathbb{R}^m$, $v \in \mathbb{R}^m$. Potom definujeme skalární součin (u, v) prvků u a v jako*

$$(u, v) = \sum_{i=1}^{m} u_i v_i = u^{\mathrm{T}} v\,.$$

Specálně, $(u, u) = \|u\|^2$. Z kontextu je zřejmé, jde o skalární součin v prostoru $\mathbb{R}^m$. Pokud je třeba dimenzi prostoru explicitně uvést, definujeme

$$(u, v) \equiv (u, v)_{\mathbb{R}^m}\,.$$

Include: obr: geometrická interpretace, kolmost

Důkaz (Lemma 2.1) Vyjdeme z identity (kosínová věta)

$$||\eta - b||^2 = ||\xi - b||^2 + ||\xi - \eta||^2 + 2(\xi - \eta, b - \xi)\,, \tag{2.11}$$

která je platná pro každou dvojici $\xi \in \mathbb{R}^m$ a $\eta \in \mathbb{R}^m$.

Nejprve ověříme implikaci (2.10) $\Longrightarrow$ (2.9). Jestliže platí (2.10), potom z identity (2.11) ihned plyne, že

$$||\eta - b||^2 = ||\xi - b||^2 + ||\xi - \eta||^2 \ge ||\xi - b||^2\,.$$

Implikaci (2.9) $\Longrightarrow$ (2.10) dokážeme sporem: Nechť existuje $\eta \in N$ tak, že $(\xi - \eta, b - \xi) \neq 0$. Definujme zobrazení, které každému $\alpha \in \mathbb{R}^1$ přiřadí

$$\Pi(\alpha) \equiv \alpha\eta + (1 - \alpha)\xi\,.$$

Zřejmě $\Pi(\alpha) \in N$. Dvojice ξ, $\Pi(\alpha)$ splňují identitu (2.11), t.j.

$$||\Pi(\alpha) - b||^2 = ||\xi - b||^2 + ||\xi - \Pi(\alpha)||^2 + 2(\Pi(\alpha) - \eta, b - \xi)$$

pro každou hodnotu parametru $\alpha \in \mathbb{R}^1$. Předpokládáme, že je splněn předpoklad (2.9) t.j., v daném kontextu,

$$||\Pi(\alpha) - b||^2 \geq ||\xi - b||^2 \,, \quad \alpha \in \mathbb{R}^1 \,.$$

Proto

$$||\xi - \Pi(\alpha)||^2 + 2(\Pi(\alpha) - \eta, b - \xi) \geq 0 \,, \quad \alpha \in \mathbb{R}^1 \,.$$

Z definice, $\xi - \Pi(\alpha) = \alpha(\xi - \eta)$. Po dosazení,

$$\alpha^2 ||\xi - \eta||^2 + 2\alpha(\xi - \eta, b - \xi) \geq 0 \,, \quad \alpha \in \mathbb{R}^1 \,. \tag{2.12}$$

Jestliže $(\xi - \eta, b - \xi) \neq 0$, potom kvadratická funkce $\alpha \longmapsto f(\alpha) \equiv \alpha^2 ||\xi - \eta||^2 = 2\alpha(\xi - \eta, b - \xi)$ nabývá zřejmě záporného minima. Což je ve sporu s vlastností (2.12). □

Připomeňme pojem *definiční obor* $\mathcal{R}(A)$ matice $A \in \mathbb{R}^{m \times n}$:

$$\mathcal{R}(A) = \{\eta \in \mathbb{R}^m \ : \exists y \in \mathbb{R}^n \,,\, Ay = \eta\} \,. \tag{2.13}$$

Důkaz (Věta 2.1) Aplikujme Lemma 2.1 v situaci, kdy $N = \mathcal{R}(A)$.

Je zřejmé, že $x \in \arg\min_{y \in \mathbb{R}^n} ||Ay - b||$ právě když

$$||Ax - b|| \leq ||Ay - b|| \quad \forall y \in \mathbb{R}^n \,.$$

Z Lemmatu vyplývá, že je to ekvivalentní podmínce

$$(Ax - b, Ax - Ay)_{\mathbb{R}^n} = 0 \quad \forall y \in \mathbb{R}^n \,.$$

Tato podmínka je ekvivalentní následujícím podmínkám:

$$(Ax - b, Ay)_{\mathbb{R}^m} = 0 \qquad \forall y \in \mathbb{R}^n \,,$$

$$(A^{\mathrm{T}} Ax - A^{\mathrm{T}} b, y)_{\mathbb{R}^n} = 0 \qquad \forall y \in \mathbb{R}^n \,,$$

$$A^{\mathrm{T}} Ax = A^{\mathrm{T}} b \,.$$

□

Jestliže matice $A^{\mathrm{T}} A$ soustavy lineárních rovnic (2.8) je regulární, t.j. $\det A^{\mathrm{T}} A \neq 0$, potom Problém 2.1 má jediné řešení

$$x = \arg\min_{y \in \mathbb{R}^n} \|Ay - b\| \,. \tag{2.14}$$

Označme $\operatorname{rank} A$ *hodnost* obdélníkové matice $A \in \mathbb{R}^{m \times n}$. Platí:

Lemma 2.2. *Nechť $A \in \mathbb{R}^{m \times n}$, $m \geq n$. Potom $\det A^{\mathrm{T}} A \neq 0$ právě když matice A má plnou hodnost, t.j. $\operatorname{rank} A = n$.*

Důkaz Nechť $\det A^{\mathrm{T}}A \neq 0$. Předpokládejme, že $\operatorname{rank} A < n$. To znamená, že existuje $x \in \mathbb{R}^n$, $x \neq 0$, $Ax = 0 \in \mathbb{R}^m$. Toto x je současně řešením soustavy $A^{\mathrm{T}}Ax = 0 \in \mathbb{R}^n$ s regulární maticí. To znamená, že $x = 0 \in \mathbb{R}^n$, což je spor s předpokladem $x \neq 0$.

Nechť $\operatorname{rank} A = n$. Předpokládejme, že $\det A^{\mathrm{T}}A = 0$. To znamená, že existuje netriviální řešení $x \in \mathbb{R}^n$, $x \neq 0$, soustavy $A^{\mathrm{T}}Ax = 0 \in \mathbb{R}^n$. Platí:

$$0 = (A^{\mathrm{T}}Ax, x)_{\mathbb{R}^n} = (Ax, Ax)_{\mathbb{R}^m} = ||Ax||^2 .$$

To znamená, že $Ax = 0 \in \mathbb{R}^m$. Protože matice A má plnou hodnost, nutně $x = 0$, což je spor. □

Nechť $\operatorname{rank} A = n$. Víme, že Problém 2.1 má jediné řešení (2.14). Jak toto řešení vypočítat? Formulace (2.8) nabízí použít eliminační techniky z Kap. 1, např. Algoritmus 1.1 resp. Algoritmus 1.2. Ukážeme, matice $A^{\mathrm{T}}A$ soustavy je symetrická, positivně definitní (s.p.d.), viz Definice 1.1. Lze tedy použít efektivnější Algoritmus 1.3, který využívá Choleského rozkladu.

Tvrzení 2.1. *Nechť $A \in \mathbb{R}^{m\times n}$, $m \geq n$. Nechť* $\operatorname{rank} A = n$. *Potom $A^{\mathrm{T}}A$ je s.p.d.*

Důkaz Definujme $B = A^{\mathrm{T}}A$. Postupně ověříme tři požadavky Definice 1.1, aby B byla s.p.d.

1) Symetrii matice B: Protože $B^{\mathrm{T}} = (A^{\mathrm{T}}A)^{\mathrm{T}} = A^{\mathrm{T}}A$, potom $B = B^{\mathrm{T}}$.

2) Semidefinitnost matice B: Pro každé $x \in \mathbb{R}^n$

$$x^{\mathrm{T}}Bx = x^{\mathrm{T}}A^{\mathrm{T}}Ax = (Ax, Ax) = ||Ax||^2 \geq 0$$

3) Positivní definitnost matice B: Nechť $x^{\mathrm{T}}Bx = 0$. Chceme ukázat, že $x = 0 \in \mathbb{R}^n$. Sporem: Nechť $x \neq 0$. Potom stejnou argumentací jako výše,

$$0 = x^{\mathrm{T}}Bx = x^{\mathrm{T}}A^{\mathrm{T}}Ax = (Ax, Ax) = ||Ax||^2 .$$

To znamená, že $Ax = 0 \in \mathbb{R}^m$. Protože A má plnou hodnost, nutně $x = 0 \in \mathbb{R}^n$, což je hledaný spor.

□

Kalkulace nákladů na výpočet řešení (2.14) pomocí Algoritmu 1.3 lze odhadnout jako $\#\text{flops} = \frac{1}{3}n^3 + O(n^2)$. K tomu je třeba přičíst náklady na maticové násobení $A^{\mathrm{T}}A$, které obnášejí $\#\text{flops} = 2n^2m$.

Poznámka 2.3. *Nechť $A \in \mathbb{R}^{m\times n}$, $m \geq n$. Nechť* $\operatorname{rank} A = n$. *Jestliže zadáme $b \in \mathbb{R}^m$, potom řešení $x \in \mathbb{R}^n$ problému* (2.14) *je dáno explicitní formulí*

$$x = (A^{\mathrm{T}}A)^{-1}A^{\mathrm{T}}b . \tag{2.15}$$

Stačí si uvědomit ekvivalentní definici (2.8). *Uvedená formule není vhodná pro výpočet* x. *Má význam z teoretického hlediska: Definujme matici* $A^+ \in \mathbb{R}^{n\times m}$

$$A^+ = (A^{\mathrm{T}}A)^{-1}A^{\mathrm{T}}\,. \tag{2.16}$$

Tedy $x = A^+b$. *Matice* A^+ *represenuje linární operátor*

$$b \in \mathbb{R}^m \longmapsto A^+\,b \;=\; x \in \mathbb{R}^n\,,$$

který zadanému $b \in \mathbb{R}^m$ *přiřazuje řešení* $x \in \mathbb{R}^n$ *problému* (2.14). *Jestliže* $m = n$, *potom* $A^+ = A^{-1}$.

V posledním odstavci této kapitoly (odst. 2.5) budeme definovat A^+ i v případě, kdy matice A nebude mít plnou hodnost.

2.3 Aplikace QR-rozkladu matice

Plán: Zavedeme nový typ rozkladu libovolné matice $A \in \mathbb{R}^{m\times n}$, tzv. QR-rozklad matice. Algoritmus tohoto maticového rozkladu probereme v odst. 2.4. V tomto odstavci uvedeme aplikaci QR-rozkladu pro řešení problému (2.14).

Definice 2.1. *Nechť* $Q \in \mathbb{R}^{m\times m}$. *Říkáme, že matice* Q *je* ortonormální, *jestliže* Q *je regulární (t.j. existuje inverzní matice* Q^{-1}*) a*

$$Q^{-1} = Q^{\mathrm{T}}\,.$$

Jinými slovy, sloupce (resp. řádky) matice Q mají jednotkové normy a jsou na sebe kolmé: Nechť $q_i \in \mathbb{R}^m$ a $q_j \in \mathbb{R}^m$ jsou i-tý a j-tý sloupec matice Q. Potom $(q_i, q_j)_{\mathbb{R}^m} = q_i^{\mathrm{T}}q_j = 0$ pro každé $i \neq j$, $(q_i, q_i)_{\mathbb{R}^m} = q_i^{\mathrm{T}}q_i = 1$ pro každé i.

Definice 2.2. *Nechť* $R \in \mathbb{R}^{m\times n}$. *Říkáme, že matice* $R = [r_{ij}]$ *s prvky* r_{ij}, $i = 1,\ldots,m$, $j = 1,\ldots,n$, *je* horní trojúhelníková, *jestliže* $r_{ij} = 0$ *pro všechna* $i > j$.

Věta 2.2. *Nechť* $A \in \mathbb{R}^{m\times n}$. *Potom existuje ortonormální matice* $Q \in \mathbb{R}^{m\times m}$ *a horní trojúhelníková matice* $R \in \mathbb{R}^{m\times n}$ *tak, že*

$$A = QR\,. \tag{2.17}$$

Důkaz je konstruktivní, viz odst. 2.4. □

Poznamenejme, že QR-rozklad není jednoznačný.

Věta platí pro libovolné rozměry $m \geq 1$, $n \geq 1$ matice $A \in \mathbb{R}^{m\times n}$. V kontextu řešení problému (2.14) budeme předpokládat, že $m \geq n$. Dále předpokládejme, že matice A má plnou hodnost, t.j. $\operatorname{rank} A = n$ a $\det(A^{\mathrm{T}}A) \neq 0$.

Při řešení normálových rovnic (2.8) využijeme rozklad (2.17):

$$A^{\mathrm{T}}Ax = (QR)^{\mathrm{T}}QRx = R^{\mathrm{T}}Q^{\mathrm{T}}QRx = R^{\mathrm{T}}Rx = A^{\mathrm{T}}b\,.$$

Matici $R^{\mathrm{T}}R$ této soustavy lze dále upravit: Definujme čtvercovou matici

$$U \in \mathbb{R}^{n\times n}\,, \quad U = [u_{ij}] \tag{2.18}$$

jako *restrikci* matice $R \in \mathbb{R}^{m\times n}$, $R = [R_{ij}]$ tak, že

$$u_{ij} = r_{ij}\,, \quad i = 1,\ldots,n\,, \quad j = 1,\ldots,n\,.$$

Lze ověřit, že

$$A^{\mathrm{T}}A = R^{\mathrm{T}}R = U^{\mathrm{T}}U\,.$$

Přitom

$$\det(A^{\mathrm{T}}A) = \det(R^{\mathrm{T}}R) = \det(U^{\mathrm{T}}U) \neq 0\,,$$
$$\operatorname{rank} A = \operatorname{rank} R = \operatorname{rank} U = n\,.$$

Normálové rovnice (2.8) lze pomocí rozkladu (2.17) ekvivalentně formuloval jako

$$U^{\mathrm{T}}Ux = A^{\mathrm{T}}b\,. \tag{2.19}$$

Věta 2.3. *Nechť $A = \mathbb{R}^{m\times n}$, $m \geq n$, $b \in \mathbb{R}^m$. Nechť* $\operatorname{rank} A = n$. *Uvažujme QR rozklad matice A:*

1. *$A = QR$, kde $Q \in \mathbb{R}^{m\times m}$, $Q^{-1} = Q^{\mathrm{T}}$ a $R^{m\times n}$ je horní trojúhelníková.*
2. *$U \in \mathbb{R}^{n\times n}$ je matice* (2.18)*, která je restrikcí matice $R^{m\times n}$.*

Potom $x = \arg\min_{y\in\mathbb{R}^n} ||Ay - b||$ *právě když*

$$x = U^{-1}d_1\,, \quad Q^{\mathrm{T}}b = \begin{bmatrix} d_1 \\ d_2 \end{bmatrix} \begin{matrix} \} & n \\ \} & m-n \end{matrix}\,.$$

Poznámka 2.4. (Věta 2.3) *Položme $d = Q^{\mathrm{T}}b \in \mathbb{R}^m$. Potom $d_1 \in \mathbb{R}^n$ označuje prvních n složek vektoru d. Podobně $d_2 \in \mathbb{R}^{m-n}$ označuje posledních $m-n$ složek vektoru d. Čtvercová matice U má plnou hodnost. Inverze U^{-1} tedy existuje. Nicméně v praktickém výpočtu nevyčíslujeme inverzi $x = U^{-1}d_1$, ale řešíme soustavu $Ux = d_1$ s horní trojúhelníkovou maticí U. Můžeme použít algoritmu zpětné substituce, viz Algoritmus 1.1.*

Důkaz (Věta 2.3) Nechť $y \in \mathbb{R}^n$. Počítejme $||Ay - b||$:

$$||Ay - b||^2 = ||QRy - b||^2 = ||Q(Ry - Q^{\mathrm{T}}b)||^2 = \underbrace{(Q(Ry - Q^{\mathrm{T}}b))^{\mathrm{T}}}(Q(Ry - Q^{\mathrm{T}}b)) =$$

$$(Ry - Q^{\mathrm{T}}b)^{\mathrm{T}}Q^{\mathrm{T}}$$

$= (Ry - Q^{\mathrm{T}}b)^{\mathrm{T}}(Ry - Q^{\mathrm{T}}b) = ||Ry - Q^{\mathrm{T}}b||^2$. Položme

$$z = Ry - Q^{\mathrm{T}}b = \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} \begin{matrix} \} n \\ \} m-n \end{matrix}$$

Když si uvědomíme blokovou strukturu vektorů $Ry \in \mathbb{R}^m$ a $Q^{\mathrm{T}}b \in \mathbb{R}^m$,

$$Ry = \begin{bmatrix} Uy \\ 0 \end{bmatrix} \begin{matrix} \} n \\ \} m-n \end{matrix} , \qquad Q^{\mathrm{T}}b = \begin{bmatrix} d_1 \\ d_2 \end{bmatrix} \begin{matrix} \} n \\ \} m-n \end{matrix}$$

je jasné, že

$$||Ay-b||^2 = ||Ry-Q^{\mathrm{T}}b||^2 = ||z||^2 = ||z_1||^2_{\mathbb{R}^n} + ||z_2||^2_{\mathbb{R}^{m-n}} = ||Uy-d_1||^2_{\mathbb{R}^n} + ||d_2||^2_{\mathbb{R}^{m-n}} .$$

Hledáme $x \in \mathbb{R}^n$,

$$x \in \mathbb{R}^n \ : \quad ||Ax - b||^2 \leq ||Ay - b||^2 \quad \forall y \in \mathbb{R}^n$$

t.j.

$$x \in \mathbb{R}^n \ : \quad ||Ux - d_1||^2 + ||d_2||^2 \leq ||Uy - d_1||^2 + ||d_2||^2 \quad \forall y \in \mathbb{R}^n .$$

Je zřejmé, že

$$x \in \mathbb{R}^n \ : \qquad Ux = d_1 .$$

□

2.4 Konstrukce QR-rozkladu

Motivace: Matici $G \in \mathbb{R}^{2\times 2}$ nazvěme *maticí rotace*, jestliže existují $c \in \mathbb{R}^1$ a $s \in \mathbb{R}^1$ tak, že

$$G = \begin{bmatrix} c & s \\ -s & c \end{bmatrix} , \quad c^2 + s^2 = 1 . \tag{2.20}$$

G je zřejmě ortonormální matice, viz Definice 2.1. Konstanty c a s lze interpretovat jako $c = \cos\theta$ a $s = \sin\theta$ nějakého úhlu $\theta \in \mathbb{R}^1$. Matice G definuje lineární transformaci

$$x \in \mathbb{R}^2 \longmapsto Gx = y \in \mathbb{R}^2 ,$$

která odpovídá rotaci zadaného vektoru x kolem počátku o úhel θ ve smyslu hodinových ručiček.

Řešme následující úlohu: K zadanému vektoru $\begin{bmatrix} a \\ b \end{bmatrix} \neq 0$ hledáme matici rotace G, viz (2.20) tak, aby

$$G \begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} * \\ 0 \end{bmatrix} . \tag{2.21}$$

Jinými slovy, nezajímá nás první složka obrazu zadaného vektoru, ale trváme na tom, aby druhá složka obrazu byla nulová. Dojdeme k následujícím požadavkům na c a s:

1. $c^2 + s^2 = 1$, viz (2.20)
2. $-sa + cb = 0$, viz druhá složka obrazu.

Úloha má dvojznačné řešení:

$$c = \frac{a}{r}\,, \quad s = \frac{b}{r} \tag{2.22}$$

a

$$c = -\frac{a}{r}\,, \quad s = -\frac{b}{r}$$

kde

$$\sqrt{a^2 + b^2}\,. \tag{2.23}$$

Vybereme variantu (2.22)&(2.23). Potom

$$G \begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} r \\ 0 \end{bmatrix}.$$

Jestliže $\begin{bmatrix} a \\ b \end{bmatrix} = 0$, potom každá dvojice c, s, která splňuje $c^2 + s^2 = 1$, je řešením úlohy (2.21). Vybereme variantu

$$c = 1\,, \quad s = 0\,. \tag{2.24}$$

Jak vypadá QR-rozklad zadané matice $A \in \mathbb{R}^{2\times 2}$? Nechť

$$A = \begin{bmatrix} a & c \\ b & d \end{bmatrix}.$$

K zadanému prvnímu sloupci matice A, t.j. vektoru $\begin{bmatrix} a \\ b \end{bmatrix}$ definujeme matici rotace $G \in \mathbb{R}^{2\times 2}$, viz (2.20), kde c a s jsou definovány formulemi (2.22)&(2.23), nebo v triviálním případě (2.24). Matice $R \equiv GA$ je horní trojúhelníková. Definujme $Q = G^{\mathrm{T}}$. Potom $A = QR$ je hledaný rozklad.

Matice G z definice (2.20) je planární příklad jisté lineární transformace v $\mathbb{R}^m$:

Definice 2.3. *Pro zadanou dvojici indexů i a j, $1 \le i < j \le m$, definujme matici $G_{ij} \in \mathbb{R}^{m\times m}$* Givensovy rotace,

$$G_{ij} = \begin{array}{l} \\ \\ i-tý\ řádek \quad \to \\ \\ j-tý\ řádek \quad \to \\ \\ \end{array} \begin{array}{c} \begin{array}{ccccccc} & & i-tý & & j-tý & & \\ & & sloupec & & sloupec & & \\ & & \downarrow & & \downarrow & & \end{array} \\ \begin{bmatrix} 1 & & & & & & \\ & \ddots & & & & & \\ & & c & & s & & \\ & & & \ddots & & & \\ & & -s & & c & & \\ & & & & & \ddots & \\ & & & & & & 1 \end{bmatrix} \end{array},$$

která závisí na volbě parametrů c a s, kde $c^2 + s^2 = 1$.

Matice G_{ij} jsou ortonormální t.j. $G_{ij}^{-1} = G_{ij}^{\mathrm{T}}$, $G_{ij}G_{ij}^{\mathrm{T}} = G_{ij}^{\mathrm{T}}G_{ij} = I$, kde $I \in \mathbb{R}^{m\times m}$ je identita. Lineární transformace

$$x \in \mathbb{R}^m \longmapsto G_{ij}\,x = y \in \mathbb{R}^m$$

každému $x \in \mathbb{R}^m$ přiřazuje $y \in \mathbb{R}^m$ s tím, že

$$y_k = x_k\,, \quad k \neq i, j\,.$$

Jinými slovy, obraz y je kopií vzoru x s výjimkou i-té a j-té složky.

Řešíme problém analogický úloze (2.21): Je dán vektor $x \in \mathbb{R}^m$. Jsou dány indexy i a j, $1 \le i < j \le m$. Hledáme Givensovu transformaci G_{ij} tak, aby obraz $y = G_{ij}x \in \mathbb{R}^m$ měl j-tou složku nulovou, t.j. $y_j = 0$. Analogie (2.22)&(2.23) a (2.24) nabízí následující řešení:

Tvrzení 2.2. *Nechť $x \in \mathbb{R}^n$. Položme $r = \sqrt{x_i^2 + x_j^2}$. Jestliže $r \neq 0$, definujme*

$$c = \frac{x_i}{r}, \quad s = \frac{x_j}{r}\,. \tag{2.25}$$

Jestliže $r = 0$, definujme

$$c = 1, \quad s = 0\,. \tag{2.26}$$

Potom

$$G_{ij}\,x \;=\; \begin{bmatrix} x_1 \\ \vdots \\ x_{i-1} \\ r \\ x_{i+1} \\ \vdots \\ x_{j-1} \\ 0 \\ x_{j+1} \\ \vdots \\ x_m \end{bmatrix}$$

Nechť $A \in \mathbb{R}^{m\times n}$. Matice $Q \in \mathbb{R}^{m\times m}$ příslušného QR-rozkladu je *konstruována* jako součin matic Givensovových transformací. Ukážeme si to na příkladě:

Příklad 2.1. *Nechť $A \in \mathbb{R}^{4\times 3}$. V následujícím schématu hvězdičky $*$ označují obecně nenulové prvky matic:*

$$A = \begin{bmatrix} * & * & * \\ * & * & * \\ * & * & * \\ * & * & * \end{bmatrix} \xrightarrow{G_{34}^{(1)}} \begin{bmatrix} * & * & * \\ * & * & * \\ * & * & * \\ 0 & * & * \end{bmatrix} \xrightarrow{G_{23}^{(1)}} \begin{bmatrix} * & * & * \\ * & * & * \\ 0 & * & * \\ 0 & * & * \end{bmatrix} \xrightarrow{G_{12}^{(1)}} \begin{bmatrix} * & * & * \\ 0 & * & * \\ 0 & * & * \\ 0 & * & * \end{bmatrix} \longrightarrow$$

$$\xrightarrow{G_{34}^{(2)}} \begin{bmatrix} * & * & * \\ 0 & * & * \\ 0 & * & * \\ 0 & 0 & * \end{bmatrix} \xrightarrow{G_{23}^{(2)}} \begin{bmatrix} * & * & * \\ 0 & * & * \\ 0 & 0 & * \\ 0 & 0 & * \end{bmatrix} \xrightarrow{G_{34}^{(3)}} \begin{bmatrix} * & * & * \\ 0 & * & * \\ 0 & 0 & * \\ 0 & 0 & 0 \end{bmatrix} = R$$

Konstruujeme šest transformací

$$G_{34}^{(1)} \in \mathbb{R}^{4\times 4}\,,\; G_{23}^{(1)} \in \mathbb{R}^{4\times 4}\,,\; G_{12}^{(1)} \in \mathbb{R}^{4\times 4}\,,\; G_{34}^{(2)} \in \mathbb{R}^{4\times 4}\,,\; G_{23}^{(2)} \in \mathbb{R}^{4\times 4}\,,\; G_{34}^{(3)} \in \mathbb{R}^{4\times 4}$$

tak, aby jejich složeným vznikla horní trojúhelníková matice $R \in \mathbb{R}^{4\times 3}$,

$$R = G_{34}^{(3)}\, G_{23}^{(2)}\, G_{34}^{(2)}\, G_{12}^{(1)}\, G_{23}^{(1)}\, G_{34}^{(1)}\, A\,.$$

Schéma představuje mezivýsledky postupných změn matice A. K nastavení parametrů c a s se využívá Tvrzení 2.2.

Matici A lze vyjádřit jako

$$A = (G_{34}^{(1)})^{-1}\,(G_{23}^{(1)})^{-1}\,(G_{12}^{(1)})^{-1}\,(G_{34}^{(2)})^{-1}\,(G_{23}^{(2)})^{-1}\,(G_{34}^{(3)})^{-1}\,R\,.$$

Matice G_{ij} jsou ortonormální. Proto

$$A = (G_{34}^{(1)})^{\mathrm{T}}\,(G_{23}^{(1)})^{\mathrm{T}}\,(G_{12}^{(1)})^{\mathrm{T}}\,(G_{34}^{(2)})^{\mathrm{T}}\,(G_{23}^{(2)})^{\mathrm{T}}\,(G_{34}^{(3)})^{\mathrm{T}}\,R\,.$$

Definujme

$$Q = (G_{34}^{(1)})^{\mathrm{T}}\,(G_{23}^{(1)})^{\mathrm{T}}\,(G_{12}^{(1)})^{\mathrm{T}}\,(G_{34}^{(2)})^{\mathrm{T}}\,(G_{23}^{(2)})^{\mathrm{T}}\,(G_{34}^{(3)})^{\mathrm{T}} \in \mathbb{R}^{4\times 4}\,.$$

Potom

$$A = Q\,R\,.$$

Při agoritmizaci si nemusíme pamatovat všech šest Givensových transformací: Položme

$$R := A\,, \quad M := I\,.$$

1. Volme $G_{34}^{(1)}$ tak, aby

$$G_{34}^{(1)}\,R(:,1) = \begin{bmatrix} * \\ * \\ * \\ 0 \end{bmatrix}\,,$$

   kde $R(:,1)$ je první sloupec matice R. Aktualisujme

$$R\ := G_{34}^{(1)}\,R\,, \quad M\ := G_{34}^{(1)}\,M\,.$$

2. Volme $G_{23}^{(1)}$ tak, aby

$$G_{23}^{(1)}\,R(:,1) = \begin{bmatrix} * \\ * \\ 0 \\ 0 \end{bmatrix}\,.$$

   Transformace $G_{23}^{(1)}$ ovlivňuje druhou a třetí složku obrazu. Vzor měl čtvrtou složku nulovou. Má ji tedy i obraz. Aktualisujme

$$R\ := G_{23}^{(1)}\,R\,, \quad M\ := G_{23}^{(1)}\,M\,.$$

3. Volme $G_{12}^{(1)}$ tak, aby

$$G_{12}^{(1)}\,R(:,1) = \begin{bmatrix} * \\ 0 \\ 0 \\ 0 \end{bmatrix}\,.$$

   Transformace $G_{12}^{(1)}$ ovlivňuje první a druhou složku obrazu. Vzor měl čtvrtou a třetí složku nulovou. Má ji tedy i obraz. Aktualisujme

$$R\ := G_{12}^{(1)}\,R\,, \quad M\ := G_{12}^{(1)}\,M\,.$$

4. Volme $G_{34}^{(2)}$ tak, aby

$$G_{34}^{(2)}\, R(:,2) = \begin{bmatrix} * \\ * \\ * \\ 0 \end{bmatrix},$$

kde $R(:,2)$ je druhý sloupec matice R. Poznamenejme, že první sloupec $R(:,1)$ má druhou, třetí a čtvrtou složku nulovou. Potom obraz $G_{34}^{(2)}\, R(:,1)$ má druhou, třetí a čtvrtou složku nulovou. Aktualisujme

$$R := G_{34}^{(2)}\, R\,, \quad M := G_{34}^{(2)}\, M\,.$$

5. Volme $G_{23}^{(2)}$ tak, aby

$$G_{23}^{(2)}\, R(:,2) = \begin{bmatrix} * \\ * \\ 0 \\ 0 \end{bmatrix},$$

kde $R(:,2)$ je druhý sloupec matice R. Poznamenejme, že

   (a) druhý sloupec $R(:,2)$ má čtvrtou složku nulovou. Potom obraz $G_{23}^{(2)}\, R(:,2)$ má čtvrtou složku nulovou.

   (b) první sloupec $R(:,1)$ má druhou, třetí a čtvrtou složku nulovou. Potom obraz $G_{23}^{(2)}\, R(:,1)$ má druhou, třetí a čtvrtou složku nulovou.

   Aktualisujme

$$R := G_{34}^{(2)}\, R\,, \quad M := G_{34}^{(2)}\, M\,.$$

6. Volme $G_{34}^{(3)}$ tak, aby

$$G_{34}^{(2)}\, R(:,3) = \begin{bmatrix} * \\ * \\ * \\ 0 \end{bmatrix},$$

kde $R(:,3)$ je druhý sloupec matice R. Poznamenejme, že

   (a) druhý sloupec $R(:,2)$ má třetí a čtvrtou složku nulovou. Potom obraz $G_{23}^{(3)}\, R(:,2)$ má třetí a čtvrtou složku nulovou.

   (b) první sloupec $R(:,1)$ má druhou, třetí a čtvrtou složku nulovou. Potom obraz $G_{23}^{(3)}\, R(:,1)$ má druhou, třetí a čtvrtou složku nulovou.

   Aktualisujme

$$R := G_{34}^{(3)}\, R\,, \quad M := G_{34}^{(3)}\, M\,.$$

   Lze tedy shrnout, že

- $R \in \mathbb{R}^{4\times 3}$ je horní trojúhelníková,
- $M \in \mathbb{R}^{4\times 4}$ lze interpetovat jako $M = G_{34}^{(3)}\, G_{23}^{(2)}\, G_{34}^{(2)}\, G_{12}^{(1)}\, G_{23}^{(1)}\, G_{34}^{(1)}$,
- $Q = M^{\mathrm{T}}$ je ortonormální,
- $Q\,R = A$.

Kalkulace nákladů QR-rozkladu: #flops $= 2n^2(m - \frac{n}{3})$.

Poznámka 2.5. *Uvedli jsme algoritmus QR-rozkladu, který je založen na Givensových transformacích. Alternativní technikou je QR-rozklad, který využívá Householderových transformací, např. [4], str. 224.*

2.5 Pseudoinverze matice

Nechť $A \in \mathbb{R}^{m\times n}$, $m \geq n$, $b \in \mathbb{R}^m$. Jestliže $\operatorname{rank} A = n$, potom řešení $x \in \mathbb{R}^n$ problému (2.6) je jednoznačně určeno. V Poznámce 2.3 jsme odvodili explicitní formuli $x = A^+ b$, kde $A^+ \in \mathbb{R}^{n\times m}$. Cílem odstavce je popsat strukturu řešení problému (2.6) v obecném případě, kdy

$$\operatorname{rank} A = r \leq n\,.$$

Připomeňme pojem *jádro* t.j. *nulový prostor* $\mathcal{N}(A)$ matice $A \in \mathbb{R}^{m\times n}$:

$$\mathcal{N}(A) = \{\, x \in \mathbb{R}^n \;:\; Ax = 0 \in \mathbb{R}^m \}\,. \tag{2.27}$$

Dimenze nulového prostoru $\dim \mathcal{N}(A) = n - r$. Zajímá nás případ $r < n$. Využijeme Větu 2.1: Nechť $x \in \arg\min_{y\in\mathbb{R}^n} \|Ay - b\|$ a $z \in \arg\min_{y\in\mathbb{R}^n} \|Ay - b\|$ t.j.,

$$A^{\mathrm{T}}Ax = A^{\mathrm{T}}b\,, \quad A^{\mathrm{T}}Az = A^{\mathrm{T}}b\,, \quad A^{\mathrm{T}}A(x - z) = 0 \in \mathbb{R}^n\,.$$

Položme $v = x - z$. Potom $A^{\mathrm{T}}Av = 0 \in \mathbb{R}^n$ právě když $v \in \arg\min_{y\in\mathbb{R}^n} \|Ay\|$. Tedy $v \in \mathcal{N}(A)$.

Z této analýzy vyplývá, že množina

$$\mathcal{X} = \left\{ z \in \mathbb{R}^n \;:\; z \in \arg\min_{y\in\mathbb{R}^n} \|Ay - b\| \right\} \tag{2.28}$$

všech řešení Problému 2.1 je *afinní prostor* dimense $\dim \mathcal{N}(A) = n - r$: Jestliže $x \in \mathcal{X}$, potom pro každé $z \in \mathcal{X}$ existuje $v \in \mathcal{N}(A)$ tak, že $z = x + v$.

Definice 2.4. *Označme $x_{LS} \in \mathcal{X}$ takové řešení Problému 2.1, které má nejmenší normu t.j.,*

$$x_{LS}^{\mathrm{T}} x_{LS} \leq z^{\mathrm{T}} z$$

pro každé $z \in \mathcal{X}$. Řekneme, že x_{LS} je řešení ve smyslu nejmenších čtverců *(least squares).*

Pro zadaná data problému $A \in \mathbb{R}^{m\times n}$, $m \geq n$, $b \in \mathbb{R}^m$ existuje právě jedno $x_{LS} \in \mathcal{X}$, kde x_{LS} je řešení ve smyslu nejmenších čtverců. Uvažujme operátor

$$A \in \mathbb{R}^{m\times n}\,,\, b \in \mathbb{R}^m \quad \longmapsto \quad x_{LS} \in \mathbb{R}^n\,,$$

který datům úlohy A a b přiřazuje řešení x_{LS}. Lze ukázat, že k dané $A \in \mathbb{R}^{m\times n}$ existuje matice $A^+ \in \mathbb{R}^{n\times m}$ tak, že

$$x_{LS} = A^+ b\,.$$

Lineárnímu operátoru A^+ se říká *pseudoinverze* matice A.

Věta 2.4. (Moore-Penrose) *Nechť $A \in \mathbb{R}^{m\times n}$. Potom pseudoinverze matice $A^+ \in \mathbb{R}^{n\times m}$ je jednoznačně definována čtyřmi vlastnostmi:*

(i) $(A^+A)^{\mathrm{T}} = A^+A\,,$

(ii) $(AA^+)^{\mathrm{T}} = AA^+\,,$

(iii) $A^+AA^+ = A^+\,,$

(iv) $AA^+A \;= A\,.$

Důkaz viz např. [1], Theorem 3.16, str. 75.

Pseudoinverzi A^+ matice A lze definovat pomocí *singulárního rozkladu* matice A, viz [4], Theorem 5.5.1, str. 257.

Kapitola 3

Nelineární rovnice

3.1 Motivace

Uvažujme zobrazení

$$f : D \longrightarrow \mathbb{R}^n ,$$

kde $D \subset \mathbb{R}^n$ je otevřená podmnožina. V této kapitole budeme hledat takové $x^* \in D$, pro které $f(x^*) = 0$. Říkáme, že x^* je kořen zobrazení f. Jinými slovy, $x^* \in D$ je řešením soustavy nelineárních rovnic

$$f(x^*) \equiv \begin{bmatrix} f_1(x^*) \\ \vdots \\ f_n(x^*) \end{bmatrix} = 0 \in \mathbb{R}^n , \qquad x^* = \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} .$$

Poznámka 3.1. *Úlohu 1.1 v kap. 1 lze formulovat jako hledání kořene $x^* \in \mathbb{R}^n$ afinního zobrazení $f(x) \equiv Ax - b$, kde $A \in \mathbb{R}^{n\times n}$, $b \in \mathbb{R}^n$ jsou daná data.*

Pro jednoduchost, nechť $n = 1$. Zobrazení $f : \mathbb{R}^1 \to \mathbb{R}^1$ můžeme interpretovat jako reálnou funkci $y = f(x)$, viz např. Obr. 3.1. Potom kořeny funkce f jsou průsečíky zadané funkce $y = f(x)$ s osou x, t.j. funkcí $y = 0$.

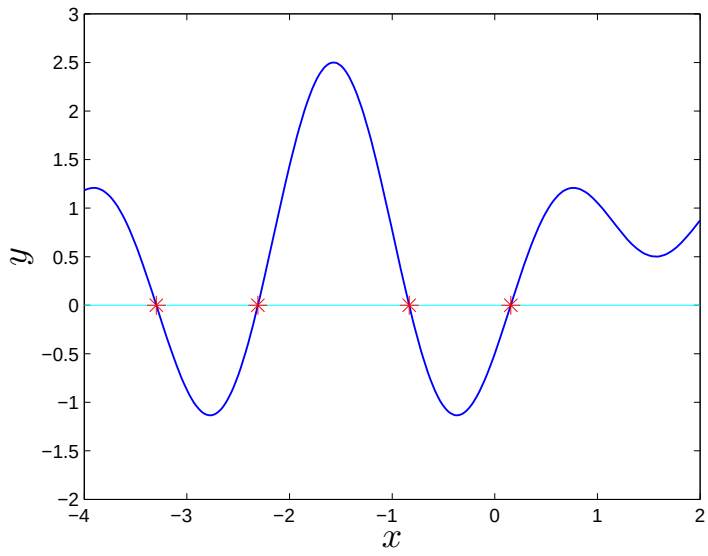
Místo kořenů funkce f budeme hledat pevné body jisté reálné funkce $g : \mathbb{R}^1 \to \mathbb{R}^1$. Říkáme, že x^* je pevným bodem zobrazení g, jestliže

$$g(x^*) = x^* .$$

Funkce g musí být volena tak, aby x^* byl kořenem funkce f právě když x^* je pevným bodem zobrazení g. Ukážeme si tři příklady volby g. Nechť $y = f(x)$ je reálná funkce:

1) Definujme $g(x) \equiv x + f(x)$. Potom

$$x^* = \underbrace{x^* + f(x^*)}_{\equiv\ g(x^*)} \quad \Longleftrightarrow \quad f(x^*) = 0 .$$



Obrázek 3.1: Zobrazení $f : \mathbb{R}^1 \to \mathbb{R}^1$, t.j. funkce $f(x) = y = \sin(3x) - \cos(2x) + .5$. Čtyři kořeny na intervalu $[-4, 2]$.

2) Definujme $g(x) \equiv x + \alpha f(x)$ kde $\alpha \neq 0$ je zvolený parametr. Potom

$$\begin{aligned} x^* = \underbrace{x^* + \alpha f(x^*)}_{\equiv\ g(x^*)} \quad &\Longleftrightarrow \quad \alpha f(x^*) = 0 \\ &\qquad\quad f(x^*) = 0. \end{aligned}$$

Ilustrace této volby je na Obr. 3.2.

3) Definujme $g(x) \equiv x + (f'(x))^{-1} f(x), \quad f'(x) \neq 0$. Potom

$$\begin{aligned} x^* = \underbrace{x^* + (f'(x^*))^{-1} f(x^*)}_{\equiv\ g(x^*)} \quad &\Longleftrightarrow \quad (f'(x^*))^{-1} f(x^*) = 0, \quad f'(x^*) \neq 0 \\ &\qquad\quad f(x^*) = 0\,. \end{aligned}$$

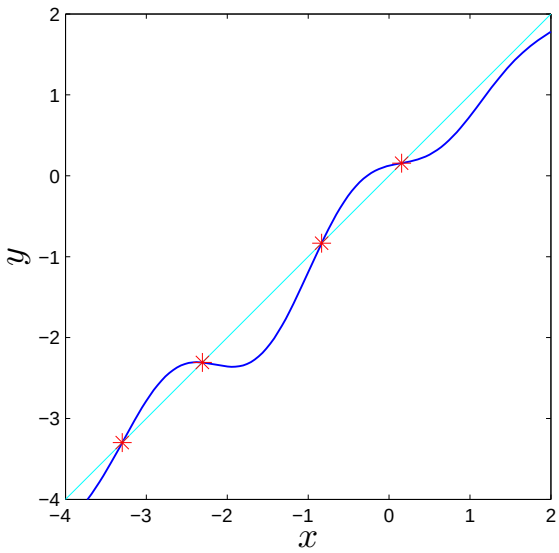
Této volbě g se říká Newtonova metoda. Podrobně si ji rozebereme v odst. 3.3.

Proč je výhodné formulovat úlohu hledání pevného bodu zobrazení $g : \mathbb{R}^1 \to \mathbb{R}^1$? Existuje totiž efektivní numerická metoda řešení: Pro zadané x^0 definujme rekurentně posloupnost

$$x^{k+1} = g(x^k), \quad k = 0, 1, \ldots\,. \tag{3.1}$$

Za jistých předpokladů

$$x^k \longrightarrow x^*$$



Obrázek 3.2: Čtyři pevné body funkce $g : \mathbb{R}^1 \to \mathbb{R}^1$, $g(x) \equiv x + \alpha(\sin(3x) - \cos(2x) + .5)$, $\alpha = -1/4$, na intervalu $[-4, 2]$.

viz. odst. 3.2.

Zadané x^0 chápeme jako počáteční aproximaci, nebo první přiblížení, pevného bodu x^*. Generováním posloupnosti (3.1) tuto aproximaci postupně zpřesňujeme. Budeme analyzovat vztah x^k, t.j. k-té aproximace pevného bodu, a pevného bodu x^*. Navržené technice (3.1) říkáme iterace zobrazení $g : \mathbb{R}^1 \to \mathbb{R}^1$. Na Obr. 3.3 je geometrická interpretace několika iterací funkce $g : \mathbb{R}^1 \to \mathbb{R}^1$, $g(x) \equiv x + \alpha(\sin(3x) - \cos(2x) + .5)$, $\alpha = -1/4$, viz Obr. 3.2, aplikovaná v okolí pevného bodu $x^* = -2.3076$. Vizualizace iterací připomíná schody. Anglicky se jí proto říká staircase diagram.

V odst. 3.2 tuto techniku zobecníme: Budeme uvažovat iterace zobrazení $g : \mathbb{R}^n \to \mathbb{R}^n$.

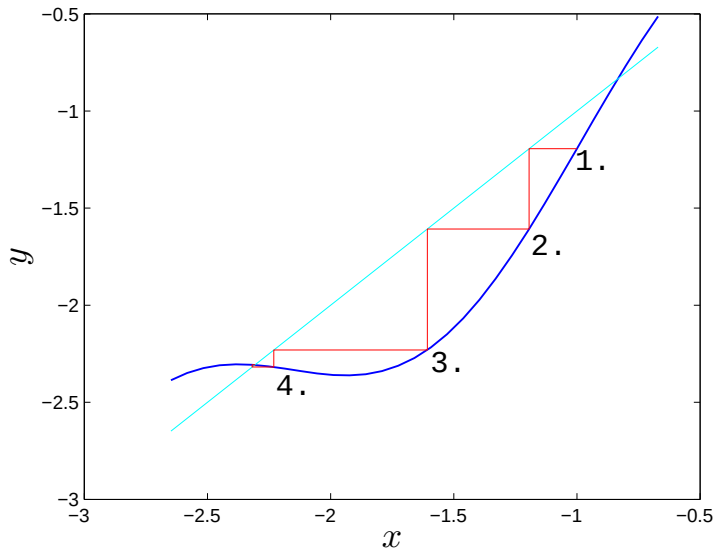
3.2 Věta o pevném bodě

Problém 3.1. *Uvažujme spojité zobrazení $g : G \longrightarrow \mathbb{R}^n$, kde $G \subset \mathbb{R}^n$ je podmnožina. Hledáme $x^* \in G$ tak, aby*

$$g(x^*) = x^* \,. \tag{3.2}$$

Říkáme, že $x^* \in G$ je pevným bodem zobrazení g.

Budeme potřebovat silnější předpoklady, než spojitost zobrazení g:



Obrázek 3.3: Iterace funkce $g : \mathbb{R}^1 \to \mathbb{R}^1$, $g(x) \equiv x + \alpha(\sin(3x) - \cos(2x) + .5)$, $\alpha = -1/4$, v okolí pevného bodu $x^* = -2.3076$. Počáteční aproximace: $x^0 = -1$. Postupné aproximace pevného bodu: $x^1 = -1.1938$, $x^2 = -1.6074$, $x^3 = -2.2302$, $x^4 = -2.3185$, $x^5 = -2.3066$, $x^6 = -2.3077$, $x^7 = -2.3076$; požadovaná přesnost: 5 platných cifer.

Definice 3.1. *Nechť* $G \subset \mathbb{R}^n$, $g : G \longrightarrow \mathbb{R}^n$. *Říkáme, že* g *je Lipschitzovsky spojitá na* G, *pokud* $\exists\ \theta \geq 0$:

$$||g(x) - g(y)|| \leq \theta ||x - y|| \quad \forall\ x, y \in G.$$

Poznámka 3.2. $||\cdot||$ *může být libovolná norma, např.* $||x|| = \sqrt{\sum x_i^2}$.

Definice 3.2. *Nechť* $G \subset \mathbb{R}^n$, $g : G \longrightarrow \mathbb{R}^n$. *Říkáme, že* g *je kontrahující na* G, *pokud* $\exists\ \theta,\ 0 \leq \theta < 1$:

$$||g(x) - g(y)|| \leq \theta ||x - y|| \quad \forall\ x, y \in G.$$

Konstantě θ *říkáme faktor kontrakce.*

Věta 3.1. *Nechť* $G \subset \mathbb{R}^n$, $g : G \longrightarrow \mathbb{R}^n$; *nechť* g *je kontrahující na* G *s faktorem kontrakce* $\theta < 1$. *Předpokládejme, že*

a) $g : G \longrightarrow G$, *t.j.* g *zobrazuje množinu* G *na* G,

b) $G = \overline{G}$, *t.j. uzavřenost množiny* G.

Potom

(i) Existuje právě jeden pevný bod $x^* \in G$.

(ii) Pro každou počáteční aproximaci $x^0 \in G$, příslušná posloupnost iterací

$$\{x^k\}_{k=0}^{+\infty} \ : \ x^{k+1} = g(x^k)$$

konverguje k x^* , t.j.

$$x^k \longrightarrow x^* \, .$$

(iii) Platí následující konvergenční odhady:

$$\begin{aligned} ||x^{k+1} - x^k|| &\leq \theta ||x^k - x^{k-1}|| \, , \\ ||x^{k+1} - x^*|| &\leq \theta ||x^k - x^*|| \, , \\ ||x^k - x^*|| &\leq \frac{\theta^k}{1-\theta} ||x^1 - x^0|| \, . \end{aligned}$$

Důkaz Jednoznačnost dokážeme sporem: Nechť existuje jiný pevný bod $x^{new} \in G$, $g(x^{new}) = x^{new}$, $x^* \neq x^{new}$. Z definice pevného bodu, a kontraktivnosti,

$$||x^* - x^{new}|| = ||g(x^*) - g(x^{new})|| \leq \theta ||x^* - x^{new}|| \, ,$$

t.j. $1 \leq \theta$, což je ve sporu s předpokladem $\theta < 1$.

Existenci x^* dokážeme konstruktivně: Nechť $x^0 \in G$. Definujme posloupnost $\{x^k\}_{k=0}^{+\infty}$ iterací $x^{k+1} = g(x^k)$. Z předpokladu a) $g : G \longrightarrow G$ plyne, že $\{x^k\}_{k=0}^{+\infty} \subset G$. V několika krocích ukážeme, že $x^k \longrightarrow x^*$.

1. Z definice iterace, a z kontraktivnosti,

$$||g(x^k) - g(x^{k-1})|| = ||x^{k+1} - x^k|| \leq \theta ||x^k - x^{k-1}|| \, .$$

Odhadneme rozdíl dvou sousedních iterací. Indukcí,

$$\begin{aligned} ||x^{k+1} - x^k|| &\leq \theta ||x^k - x^{k-1}|| \, , \\ &\leq \theta^2 ||x^{k-1} - x^{k-2}|| \, , \\ &\vdots \\ &\leq \theta^k ||x^1 - x^0|| \, . \end{aligned}$$

2. Odhadneme rozdíl k-té a $(k+m)$-té iterace. Z trojúhelníkové nerovnosti, a z odhadu sousedních iterací,

$$\begin{aligned} ||x^{k+m} - x^k|| &\leq ||x^{k+m} - x^{k+m-1}|| + ||x^{k+m-1} - x^{k+m-2}|| + \cdots + ||x^{k+1} - x^k|| \, , \\ &\leq \underbrace{(\theta^{k+m-1} + \cdots + \theta^k)}_{\theta^k(\theta^{m-1} + \cdots + 1)} ||x^1 - x^0|| \, . \end{aligned}$$

Geometrickou posloupnost sečteme a odhadneme:

$$\theta^{m-1} + \cdots + 1 = \frac{1-\theta^m}{1-\theta} < \frac{1}{1-\theta}\,.$$

Proto

$$||x^{k+m} - x^k|| \leq \frac{\theta^k}{1-\theta}||x^1 - x^0|| \tag{3.3}$$

pro libovolné přirozené m.

3. Z odhadu (3.3) plyne, že posloupnost $\{x^k\}_{k=0}^{+\infty}$ je Cauchyovská t.j.
$\forall \varepsilon > 0\ \exists N\ :$

$$k \geq N,\ k+m \geq N \quad \Longrightarrow \quad ||x^k - x^{k+m}|| \leq \varepsilon\,.$$

Z uzavřenosti množiny G plyne, že posloupnost iterací konverguje k nějakému limitnímu prvku $x^\infty \in G$.

4. Ukážeme, že x^∞ je pevný bod zobrazení g t.j. $g(x^\infty) = x^\infty$. Platí:

$$||x^\infty - g(x^\infty)|| = ||x^\infty - x^{k+1} + g(x^k) - g(x^\infty)|| \leq$$

$$\leq ||x^\infty - x^{k+1}|| + ||g(x^k) - g(x^\infty)|| \leq ||x^\infty - x^{k+1}|| + \theta||x^k - x^\infty||\,.$$

Protože $x^k \longrightarrow x^\infty$, potom $||x^\infty - g(x^\infty)|| = 0$. Z jednoznačnosti pevného bodu plyne, že $x^\infty = x^*$.

Poslední z konvergenčních odhadů získáme z (3.3) limitním přechodem. První dva jsou zřejmé. □

Věta 3.2 zaručuje existenci a jednoznačnost pevného bodu x^*. V příkladu na Obr. 3.2 existuje více pevných bodů. Pokud existuje nějaký pevný bod x^*, potom můžeme zaručit konvergenci iterací $\{x^k\}_{k=0}^{+\infty}$, pokud počáteční aproximace x^0 je dostatečně blízko uvažovaného pevného bodu x^*. (Samozřejmě nadále předpokládáme, že g je kontrahující.) V tom případě mluvíme o lokální konvergenci iterací. Podaří se nám vypustit velmi omezující předpoklad a) $g : G \longrightarrow G$ předchozí Věty 3.1, t.j. že g zobrazuje množinu G na G:

Věta 3.2. (Lokální konvergence)
Nechť $D \subset \mathbb{R}^n$, D je otevřená.

Nechť $g : D \longrightarrow \mathbb{R}^n$, g je kontrahující na D, s faktorem kontrakce $\theta < 1$.

Nechť $x^* \in D$: $g(x^*) = x^*$.

Potom existuje dostatečně malé $\varepsilon > 0$ tak, že je-li $x^0 \in D$, $||x^0 - x^*|| < \varepsilon$,

pak posloupnost $\{x^k\}_{k=0}^{+\infty}$, $x^{k+1} = g(x^k)$, konverguje $x^k \longrightarrow x^*$.

Důkaz Nechť $x^0 \in D$. Definujme iterace $\{x^k\}_{k=0}^{+\infty}$, $x^{k+1} = g(x^k)$. Předpokládejme, že $x^k \in D$ pro každé k, a že $x^k \longrightarrow x^*$. Pro analýzu posloupnosti $\{x^k\}_{k=0}^{+\infty}$ lze použít kroky 1.-4. předchozí Věty 3.1: Posloupnost je Cauchyovská, a předpokládáme že má limitu $x^\infty = x^*$. Platí (3.3). Limitním přechodem $x^{k+m} \longrightarrow x^*$ pro $m \longrightarrow \infty$ dostaneme odhad

$$||x^k - x^*|| \ \leq \ \frac{\theta^k}{1-\theta}||x^1 - x^0|| \leq \frac{\theta}{1-\theta}||x^1 - x^0||$$

nezávisle na k. Chybu první iterace $||x^1 - x^0||$ odhadneme pomocí $x^0 - x^*$:

$$||x^1 - x^0|| = ||x^1 - x^* + x^* - x^0|| \leq \underbrace{||x^1 - x^*||}_{\leq \theta||x^0 - x^*||} + ||x^* - x^0|| \leq (1+\theta)\,||x^0 - x^*||\,.$$

Lze shrnout, že

$$||x^k - x^*|| \leq \theta\frac{1+\theta}{1-\theta}||x^0 - x^*|| \quad \forall k\,.$$

Jestliže tedy $x^0 \in D$, $||x^0 - x^*|| < \varepsilon$, potom $||x^k - x^*|| \leq \varepsilon\,\theta\,\frac{1+\theta}{1-\theta}$ pro každý index k. Protože $x^* \in D$, D je otevřená, potom existuje dostatečně malé $\varepsilon > 0$ tak, aby $\varepsilon\,\theta\,\frac{1+\theta}{1-\theta}$-okolí bodu x^* patřilo do množiny D.

Ukázali jsme, že existují dvě okolí bodu x^*: ε-okolí a $\varepsilon\,\theta\,\frac{1+\theta}{1-\theta}$ -okolí. Jestliže $x^0 \in D$, $||x^0 - x^*|| < \varepsilon$, potom každý prvek x^k posloupnosti $\{x^k\}_{k=0}^{+\infty}$, $x^{k+1} = g(x^k)$, patří do $\varepsilon\,\theta\,\frac{1+\theta}{1-\theta}$ -okolí bodu x^*, a tedy do množiny D. Pro analýzu posloupnosti $\{x^k\}_{k=0}^{+\infty}$ lze použít kroky 1.-4. předchozí Věty 3.1 a tím ukázat, že $x^k \longrightarrow x^*$.

Platí analogické konvergenční odhady jako ve Větě 3.1. □

Uvedeme několik poznámek, které se týkají konvergence iteračního procesu $\{x^k\}_{k=0}^{+\infty}$, $x^{k+1} = g(x^k)$. Jde o komentář ke konvergenčním odhadům Věty 3.1 a Věty 3.2.

Poznámka 3.3. *Rekurentním použitím odhadu* $||x^{k+1} - x^*|| \leq \theta||x^k - x^*||$ *odvodíme, že*

$$||x^k - x^*|| \leq \theta||x^{k-1} - x^*|| \leq \ \ldots \ \leq \theta^k||x^0 - x^*||\,.$$

Definujme $\varepsilon_k \equiv \ ||x^k - x^*||$. *Nezáporné číslo* ε_k *interpretujeme jako chybu* $k-$*té aproximace (*$k-$*té iterace). Platí:*

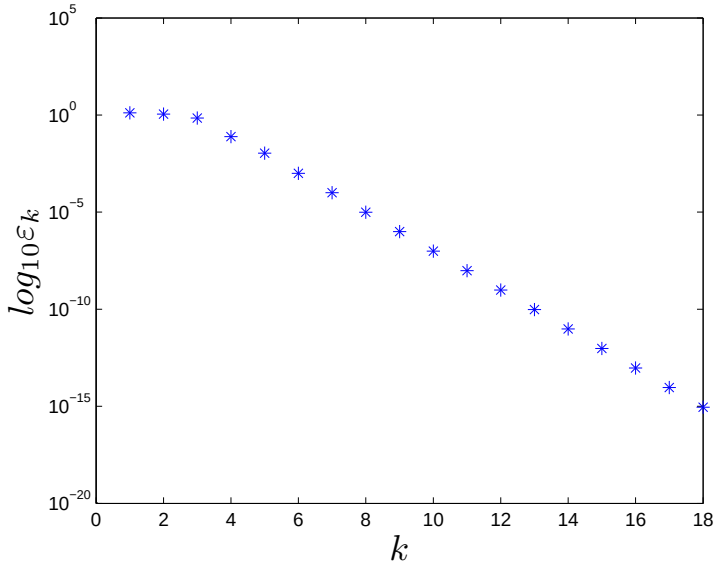
$$\varepsilon_k \leq \theta^k\varepsilon_0\,. \tag{3.4}$$

Čím menší je faktor kontrakce $\theta < 1$, *tím větší je pokles chyby iteračního procesu.*

Poznámka 3.4. *Je třeba si uvědomit, že* (3.4) *představuje horní odhad chyby. Jinými slovy, iterační proces může konvergovat rychleji. Jsou ale iteračni procesy, kde asymptoticky platí*

$$\varepsilon_k \approx \theta^k\varepsilon_0 \tag{3.5}$$

pro dostatečně velké indexy k. *V tom případě*



Obrázek 3.4: Iterace funkce $g : \mathbb{R}^1 \to \mathbb{R}^1$, $g(x) \equiv x + \alpha(\sin(3x) - \cos(2x) + .5)$, $\alpha = -1/4$, v okolí pevného bodu $x^* = -2.30763404376158$, s počáteční aproximací $x^0 = -1$. Analýza chyby $\varepsilon_k \equiv ||x^k - x^*||$ v závislosti na pořadovém čísle k dané iterace.

- *chyba ε_k klesá úměrně počtu iterací k,*
- *počet platných cifer výsledku roste lineárně s k .*

Říkáme, že iterační proces (3.5) *konverguje lineárně, jestliže*

$$\lim_{k \to +\infty} \frac{\varepsilon_{k+1}}{\varepsilon_k} = \theta \,. \tag{3.6}$$

Mluvíme o lineární konvergenci.

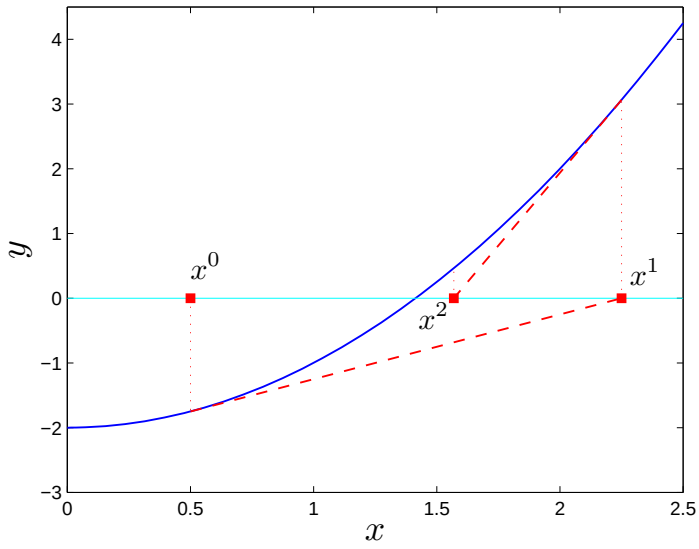
Na Obr. 3.4 jsou analyzovány chyby $\varepsilon_k \equiv ||x^k - x^*||$ iterací funkce $g : \mathbb{R}^1 \to \mathbb{R}^1$, $g(x) \equiv x + \alpha(\sin(3x) - \cos(2x) + .5)$, $\alpha = -1/4$, v okolí pevného bodu $x^* = -2.30763404376158$, s počáteční aproximací $x^0 = -1$. Vychází, že

$$\theta = 0.10007561154043 \,.$$

Jedné iteraci odpovídá zisk jedné platné cifry pevného bodu.

3.3 Newtonova metoda a její modifikace

Probereme postupně tři iterační metody. Každou z nich uvedeme nejprve v jednodimenziální verzi jako metodu hledání kořene $x^* \in \mathbb{R}^1$ funkce $f \in \mathbb{R}^1 \to \mathbb{R}^1$, $y = f(x)$.



Obrázek 3.5: Newtonova metoda aplikovaná na hledání kořenů funkce $y = f(x) \equiv x^2 - 2$. První tři aproximace kořene $x^* = \sqrt{2}$.

Ideu zobecníme pro řešení soustavy nelineárních rovnic:

Problém 3.2. *Je dáno*

$$f : D \longrightarrow \mathbb{R}^n \,, \tag{3.7}$$

kde $D \subset \mathbb{R}^n$ je otevřená podmnožina. Hledáme $x^* \in D$ tak, aby

$$f(x^*) = 0 \,. \tag{3.8}$$

3.3.1 Newtonova metoda

Newtonova metoda: $n = 1$.

Definujme funkci $g : \mathbb{R}^1 \to \mathbb{R}^1$, $g(x) \equiv x - (f'(x))^{-1} f(x)$. Pro zadané x^0 konstruujeme posloupnost iterací $\{x^k\}_{k=0}^{+\infty}$

$$x^{k+1} = g(x^k) \equiv x^k - \left(f'(x^k)\right)^{-1} f(x^k) = x^k - \frac{f(x^k)}{f'(x^k)} \,. \tag{3.9}$$

Zadané x^0 je počáteční aproximace kořene x^*. Předpokládáme, že iterace existují, t.j. $f'(x^k) \neq 0$ pro každé k.

Geometrickou interpretaci objasňuje Obr. 3.5: V bodě x^k definujme tečnu

$$y = f(x^k) + f'(x^k)(x - x^k)$$

funkce $y = f(x)$. Průsečík této tečny s osou x, t.j. s přímkou $y = 0$, definuje x^{k+1}.

Nechť $x^* \in \mathbb{R}^1$ je kořen funkce f. Budeme ověřovat lokální konvergenci iterací (3.9) podle Věty 3.2. Klíčové je ověřit, že funkce $x \longmapsto g(x) \equiv x + (f'(x))^{-1} f(x)$ je v okolí kořene x^* kontrahující.

Poznámka 3.5. *Formální výpočet dává*

$$g'(x) = 1 - \frac{(f'(x))^2 - f(x)f''(x)}{(f'(x))^2} = \frac{f''(x)}{(f'(x))^2} f(x)\,.$$

Nechť $f'(x^*) \neq 0$. Jestliže $f(x^*) = 0$, potom

$$g(x^*) = x^*\,, \quad g'(x^*) = 0\,, \quad g''(x^*) = \frac{f''(x^*)}{f'(x^*)}\,. \tag{3.10}$$

Poznámka 3.6. *Nechť $f'(x^*) \neq 0$. Nechť g' je spojitá v x^*. Podle definice spojitosti, pro každé $\varepsilon > 0$ existuje $\delta > 0$ tak, že platí:*

$$\textit{jestliže} \quad ||\xi - x^*|| < \delta, \quad \textit{potom} \quad ||g'(\xi) - \underbrace{g'(x^*)}_{=0}|| < \varepsilon\,.$$

Tvrzení 3.1. *Nechť $f'(x^*) \neq 0$. Nechť g' je spojitá v x^*. Potom pro každé $\varepsilon > 0$ existuje $\delta > 0$ tak, že platí: Jestliže $||x - x^*|| < \delta$ a $||y - x^*|| < \delta$, potom*

$$||g(x) - g(y)|| \leq \varepsilon\, ||x - y||\,.$$

Důkaz Zvolme $\varepsilon > 0$. Potom existuje $\delta > 0$ z Poznámky 3.6. Uvažujme libovolnou dvojici x, y splňující $||x - x^*|| < \delta$, $||y - x^*|| < \delta$.

Nechť $x < y$. Potom podle Věty o střední hodnotě existuje ξ tak, že

$$x < \xi < y\,, \quad g(x) - g(y) = g'(\xi)(x - y)\,.$$

Zřejmě $||\xi - x^*|| < \delta$. Proto podle Poznámky 3.6 je $||g'(\xi)|| < \varepsilon$. Tedy

$$||g(x) - g(y)|| \leq ||g'(\xi)|| \cdot ||x - y|| < \varepsilon ||x - y||\,.$$

Případ $x > y$ se analyzuje analogicky. □

Tvrzení 3.1 použijeme k verifikaci předpokladů Věty 3.2 o lokální konvergenci: Je třeba ověřit, že funkce $g(x) \equiv x + (f'(x))^{-1} f(x)$ je kontrahující v nějakém δ-okolí bodu x^*, s faktorem kontrakce $\theta < 1$. Tvrzení 3.1 ale ukazuje, že tímto faktorem kontrakce je právě ε. Pro libovolné $\varepsilon < 1$ existuje δ-okolí bodu x^* tak, funkce g je na tomto okolí kontrahující, a tedy iterační proces je lokálně konvergentní.

Poznámka 3.7. *Newtonova metoda konverguje rychleji než pouze lineárně: Iterace Newtonovy metody jsou charakterizovány tím, že faktor kontrakce ε si lze předepsat jako libovolně malý na odpovídajícím δ-okolí bodu x^*. Říkáme, že příslušný iterační proces konverguje superlineárně, jestliže*

$$\lim_{k\to+\infty} \frac{\varepsilon_{k+1}}{\varepsilon_k} = 0\,. \tag{3.11}$$

kde $\varepsilon_k = ||x^k - x^*||$ je chyba $k-$té iterace.

Nicméně Newtonova metoda konverguje ještě rychleji než superlineárně.

Poznámka 3.8. *Nechť $f'(x^*) \neq 0$. Nechť g'' je spojitá v x^*. Podle definice spojitosti, pro každé $\varepsilon > 0$ existuje $\delta > 0$ tak, že platí:*

$$\textit{jestliže} \quad ||\xi - x^*|| < \delta\,, \quad \textit{potom} \quad ||g''(\xi) - g''(x^*)|| < \varepsilon\,.$$

Tvrzení 3.2. *Nechť $f'(x^*) \neq 0$. Nechť g'' je spojitá v bodě x^*. Potom pro každé $\varepsilon > 0$ existuje $\delta > 0$ tak, že platí: Jestliže $||x - x^*|| < \delta$, pak*

$$||g(x) - g(x^*)|| \leq C\ ||x - x^*||^2\,,$$

kde

$$C \equiv \frac{1}{2}(\varepsilon + ||g''(x^*)||)\,, \quad g''(x^*) = \frac{f''(x^*)}{f'(x^*)}\,.$$

Důkaz Nechť $x < x^*$.

Uvažujme Taylorův rozvoj funkce $g = g(x)$ se středem v bodě x^*: Existuje ξ : $x < \xi < x^*$ tak, že

$$g(x) = g(x^*) + g'(x^*)(x - x^*) + \frac{1}{2}g''(\xi)(x - x^*)\,.$$

Připomeňme (3.10). Potom

$$||g(x) - x^*|| \leq \frac{1}{2}||g''(\xi)|| \cdot ||x - x^*||^2 \leq \frac{1}{2}\left(\underbrace{||g''(\xi) - g''(x^*)||}_{} + ||g''(x^*)||\right) \cdot ||x - x^*||^2\,.$$

Z Poznámky 3.8 plyne, že pro každé $\varepsilon > 0$ existuje $\delta > 0$ tak, že

$$||g(x) - x^*|| \leq \frac{1}{2}\left(\varepsilon + ||g''(x^*)||\right)\ ||x - x^*||^2$$

jestliže $||x - x^*|| < \delta$. Případ $x > x^*$ lze analyzovat analogicky.

□

Uvažujeme iterační proces $\{x^k\}_{k=0}^{+\infty}$ definovaný Newtonovou metodou (3.9). Pokud počáteční aproximace x^0 je dostatečně blízko kořene x^* rovnice $f(x^*) = 0$,

potom $x^k \to x^*$. Proto říkáme, že Newtonova metoda konverguje lokálně. Označme $\varepsilon_k \equiv ||x^k - x^*||$ chybu $k-$té aproximace řešení x^*. Na základě Tvrzení 3.2 lze odhadnout, že

$$\varepsilon_{k+1} \leq C\,\varepsilon_k^2\,, \tag{3.12}$$

kde $C \sim ||g''(x^*)||$.

Jestliže iterační proces je charakterizován odhadem (3.12), potom říkáme, že proces konverguje kvadraticky. Uveďme schematický příklad:

Příklad 3.1. *Nechť $C = 1$. Předpokládejme, že $\varepsilon_0 \leq 0.1$. Potom*

$$\varepsilon_1 \leq (\varepsilon_0)^2 \leq 10^{-2}\,, \quad \varepsilon_2 \leq (\varepsilon_1)^2 \leq 10^{-4}\,, \quad \varepsilon_3 \leq (\varepsilon_2)^2 \leq 10^{-8}\,, \quad \varepsilon_4 \leq (\varepsilon_3)^2 \leq 10^{-16}\,.$$

To znamená, že

$$\begin{array}{rcl} \varepsilon_0 & \leq & 0.**\ldots \\ \varepsilon_1 & \leq & 0.0**\ldots \\ \varepsilon_2 & \leq & 0.000**\ldots \\ \varepsilon_3 & \leq & 0.0000000**\ldots \end{array}$$

(∗ ... platné cifry výsledku).

Metodu (3.9) zobecníme. Řešíme Problém 3.2.

Algoritmus 3.1. (Newtonova metoda) *Nechť $f \in C^1(D, \mathbb{R}^n)$. Pro zadané $x^0 \in D$ konstruujeme posloupnost iterací $\{x^k\}_{k=0}^{+\infty}$*

$$x^{k+1} = g(x^k) \equiv x^k - \left(f'(x^k)\right)^{-1} f(x^k)\,. \tag{3.13}$$

Zadané x^0 je počáteční aproximace nějakého kořene $x^* \in D$ soustavy (3.8). Předpokládáme, že iterace existují, t.j. $f'(x^k) \neq 0$ a že $x^k \in D$ pro každé k.

Připomeňme, že $f = f(x)$ je vektorová funkce (3.7),

$$f(x) = \begin{bmatrix} f_1(x) \\ \vdots \\ f_n(x) \end{bmatrix} \in \mathbb{R}^n\,, \qquad x = \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} \in \mathbb{R}^n\,,$$

a

$$f'(x) = \begin{bmatrix} \dfrac{\partial f_1(x)}{\partial x_1}\,, & \cdots, & \dfrac{\partial f_1(x)}{\partial x_n} \\ \vdots & & \\ \dfrac{\partial f_n(x)}{\partial x_1}\,, & \cdots\,, & \dfrac{\partial f_n(x)}{\partial x_n} \end{bmatrix} \in \mathbb{R}^{n\times n}\,.$$

Matici $f'(x) \in \mathbb{R}^{n\times n}$ se říká Jacobiho matice.

Předpokládáme, že kořen x^* je regulární, t.j.

$$x^* \ : \ f(x^*) = 0\,, \quad \det\, f'(x^*) \neq 0\,. \tag{3.14}$$

Tento kořen počítáme jako pevný bod x^* iterací (3.13), t.j.

$$x^* \ : \ g(x^*) = x^*\,. \tag{3.15}$$

Algoritmus 3.1 je schopen generovat pouze konečný počet iteračních kroků. Proto je třeba definovat nějaké konvergenční kriterium. Např., můžeme požadovat, aby

$$||f(x^k)|| < \mathrm{tol}_1 \ \ \& \ \ ||x^k - x^{k-1}|| < \mathrm{tol}_2\,, \tag{3.16}$$

kde tol_1 a tol_2 jsou předepsané tolerance. Za numerické řešení x^k považujeme první člen posloupnosti $\{x^j\}_{j=0}^{+\infty}$, který splňuje kriterium (3.16).

Při implementaci algoritmu se nepočítá inverze matice $\left(f'(x^k)\right)^{-1}$ v příslušném kroku (3.13) t.j. v kroku $x^k \longrightarrow x^{k+1}$. Místo toho se x^{k+1} počítá z řešení jisté soustavy lineárních rovnic:

(i) Hledáme $\delta x \in \mathbb{R}^n$ jako řešení soustavy lineárních rovnic

$$A \cdot \delta x = b\,,$$

kde

$$A \equiv f'(x^k) \in \mathbb{R}^{n\times n}\,, \quad b \equiv -f(x^k) \in \mathbb{R}^n\,.$$

(ii) $x^{k+1} \ := \ x^k + \delta x \ .$

Vektor δx označuje mnemotechnicky přírůstek. Nová aproximace řešení x^{k+1} je definována jako stará aproximace x^k plus přírůstek δx. Pro řešení soustavy lineárních rovnic lze použít např. Gaussovu eliminaci se sloupcovou pivotací, t.j. Algoritmus 1.2

Zformulujeme konvergenční větu:

Věta 3.3. *Nechť $f \in C^1(D, \mathbb{R}^n)$. Předpokládejme, že existuje kořen $x^* \in D$. Nechť existuje inverze $(f'(x^*))^{-1}$. Položme $C = ||\,(f'(x^*))^{-1}\,||$. Nechť existují kladné konstanty R a L takové, že*

$$||f'(x) - f'(y)|| \leq L||x - y|| \qquad \forall x, y \in \mathrm{B}(x^*, R)\,,$$

kde $\mathrm{B}(x^*, R) = \{x \in \mathbb{R}^n \ : \ ||x - x^*|| < R\}$ je okolí bodu x^*. Potom existuje $r > 0$ tak, že pro každou počáteční aproximaci $x^0 \in \mathrm{B}(x^*, r)$ posloupnost iterací $\{x^k\}_{k=0}^{+\infty}$ (3.13) je dobře definovaná a konverguje k x^*. Platí:

$$||x^{k+1} - x^*|| \leq CL||x^k - x^*||^2\,.$$

Důkaz viz např. [1], Theorem 7.1, str. 283.

Lze tedy shrnout, že Newtonova metoda konverguje lokálně a kvadraticky.

3.3.2 Modifikace Newtonovy metody

Uvedeme dvě metody.

a) Newtonova sečná metoda, The Newton-chord (angl. sečna) method : $n = 1$.

Pro zadané x^0 definujme funkci $g : \mathbb{R}^1 \to \mathbb{R}^1$, $g(x) \equiv x - (f'(x^0))^{-1} f(x)$. Předpokládáme, že $f'(x^0) \neq 0$. Konstruujeme posloupnost iterací $\{x^k\}_{k=0}^{+\infty}$

$$x^{k+1} = g(x^k) \equiv x^k - \left(f'(x^0)\right)^{-1} f(x^k) = x^k - \frac{f(x^k)}{f'(x^0)}\,. \tag{3.17}$$

Zadané x^0 je počáteční aproximace kořene x^*.

Metodu (3.17) zobecníme. Řešíme Problém 3.2.

Algoritmus 3.2. (Newtonova sečná metoda) *Nechť $f \in C^1(D, \mathbb{R}^n)$. Pro zadané $x^0 \in D$, $\det(f'(x^0)) \neq 0$, konstruujeme posloupnost iterací $\{x^k\}_{k=0}^{+\infty}$*

$$x^{k+1} = g(x^k) \equiv x^k - \left(f'(x^0)\right)^{-1} f(x^k)\,. \tag{3.18}$$

Poznamenejme, že x^{k+1} se počítá z řešení soustavy lineárních rovnic:

(i) Hledáme $\delta x \in \mathbb{R}^n$ jako řešení soustavy lineárních rovnic

$$A \cdot \delta x = b\,,$$

kde

$$A \equiv f'(x^0) \in \mathbb{R}^{n\times n}\,, \quad b \equiv -f(x^k) \in \mathbb{R}^n\,.$$

(ii) $x^{k+1} := x^k + \delta x$.

Výhodou metody je, že nemusíme v každém počítat Jakobiho matici $f'(x^k)$. Lze ukázat, že Newtonova sečná metoda lokálně konveguje. Nicméně, konverguje pouze lineárně, viz (3.6). Pokud faktor kontrakce $\theta < 1$ je blízko jedničce, potom zisk jedné platné cifry výsledku může představovat desítky iterací.

b) Metoda sečen, The secant method : $n = 1$.

Metoda představuje dvojčlennou rekurenci: Pro zadané x^0 a x^{-1} generujeme posloupnost iterací $\{x^k\}_{k=-1}^{+\infty}$,

$$x^{k+1} = x^k - A_k^{-1} f(x^k)\,, \tag{3.19}$$

kde

$$A_k = \frac{f(x^k) - f(x^{k-1})}{x^k - x^{k-1}}\,. \tag{3.20}$$

Geometrická interpretace: Nechť $y = f(x)$ je graf funkce. K dané dvojici x^k, x^{k-1} definujme sečnu

$$y = f(x^k) + \frac{f(x^k) - f(x^{k-1})}{x^k - x^{k-1}} (x - x^k) ,$$

která prochází body $[x^k, f(x^k)]$ a $[x^{k-1},\ f(x^{k-1})]$. Průsečík této sečny s osou $y = 0$ definuje x^{k+1}:

$$0 = f(x^k) + \frac{f(x^k) - f(x^{k-1})}{x^k - x^{k-1}} (x^{k+1} - x^k) .$$

Tato definice x^{k+1} je ekvivalentní (3.19)&(3.20).

Include: geometrická interpretace

Cílem je zobecnění algoritmu (3.19)&(3.20) pro dimenzi $n \geq 1$. V zobecněném algoritmu musí vystupovat nové stavové proměnné $x^k \in \mathbb{R}^n$, $A_k \in \mathbb{R}^{n\times n}$. Vektorová analogie (3.19) je zřejmá, viz (3.23). Analogii skalární podmínky (3.20) lze zformulovat takto: Hledáme $A_k \in \mathbb{R}^{n\times n}$ tak, aby

$$A_k s_k = y_k , \quad s_k = x^k - x^{k-1} , \quad y_k = f(x^k) - f(x^{k-1}) . \tag{3.21}$$

Matice A_k není, na rozdíl od skalárního případu, jednoznačně určena daty $x^k - x^{k-1}$ a $f(x^k) - f(x^{k-1})$. Počet neznámých prvků matice je n^2, ale rovnice (3.21) představují jen n podmínek.

Matici A_k hledejme ve tvaru

$$A_k = A_{k-1} + \delta A ,$$

kde matice $A_{k-1} \in \mathbb{R}^{n\times n}$ je určena v předchozím iteračním kroku, a $\delta A \in \mathbb{R}^{n\times n}$ je nějaká aktualizace, angl. update. Každá aktualizace δA musí být konsistentní s (3.21) t.j.,

$$A_k s_k = A_{k-1} s_k + \delta A s_k = y_k .$$

Tzv. Broydenova aktualizace je definována takto:

$$\delta A = \frac{y_k - A_{k-1} s_k}{s_k^{\mathrm{T}} s_k} s_k^{\mathrm{T}} . \tag{3.22}$$

Je konsistentní. Přitom $\operatorname{rank} \delta A = 1$. V jistém smyslu je to nejmenší možná aktualizace matice A_{k-1}, viz Poznámka 3.11. Krok (3.24) je definován jako Broydenova aktualizace.

Algoritmus 3.3. (Broydenova metoda) *Nechť $f \in C^1(D, \mathbb{R}^n)$. Pro zadané $x^{-1} \in D$, $x^0 \in D$ a $A_{-1} \in \mathbb{R}^{n\times n}$ definujeme posloupnost iterací $\{x^k\}_{k=-1}^{+\infty}$*

$$x^{k+1} = x^k - A_k^{-1} f(x^k) , \tag{3.23}$$

kde

$$A_k = A_{k-1} + \frac{y_k - A_{k-1}s_k}{s_k^{\mathrm{T}} s_k}\, s_k^{\mathrm{T}}\,, \tag{3.24}$$

přičemž

$$s_k = x^k - x^{k-1}\,, \quad y_k = f(x^k) - f(x^{k-1})\,. \tag{3.25}$$

Poznámka 3.9. *K inicializaci algoritmu lze např. použít jeden krok Newtonovy metody:*

$$x^{-1} \in D\,, \quad A_{-1} = f'(x^{-1})\,, \quad x^0 = x^{-1} - (A_{-1})^{-1} f(x^{-1})$$

Poznámka 3.10. Realizace iteračního kroku $x^{k-1}, x^k, A_{k-1} \longrightarrow x^k, x^{k+1}, A_k$.

1. *Položíme* $s_k = x^k - x^{k-1}$, $y_k = f(x^k) - f(x^{k-1})$.
2. *Definujeme* $A_k = A_{k-1} + \frac{y_k - A_{k-1}s_k}{s_k^{\mathrm{T}} s_k}\, s_k^{\mathrm{T}}$.
3. *Hledáme* $\delta x \in \mathbb{R}^n$ *tak, aby* $A_k \cdot \delta x = -f(x^k)$.
4. *Definujeme* $x^{k+1} = x^k + \delta x$.

Poznámka 3.11. *Platí:*

$$A_k = \arg \min_{A \in \mathbb{R}^{n\times n}} \{||A - A_{k-1}||_F \;:\; As_k = y_k\}\,,$$

$$A - A_{k-1} = B\,, \;\; ||B||_F \equiv \left(\sum_{i,j=1}^{n} (B_{ij})^2 \right)^{1/2}.$$

Normě $||B||_F$ se říká Frobeniova norma matice B.

Lze dokázat lokální, superlineární konvergenci t.j. pro chybu platí (3.11). Více informací lze najít např. v [3], odst. 7.1.4, str. 288.

Kapitola 4

Minimalizace funkce více proměnných

Nechť $f : \mathbb{R}^n \to \mathbb{R}^1$ je reálná funkce proměnné $x \in \mathbb{R}^n$, se složkami $x = (x_1, \ldots, x_n)$. Předpokládejme, že f je spojitá s definičním oborem $D \subset \mathbb{R}^n$, kde D je otevřená množina. Označení: $f \in C(D)$.

Problém 4.1. *Hledáme $x^* \in D$, ve kterém funkce f nabývá lokální minimum t.j., existuje otevřená podmnožina $\Omega \subset D$ tak, že*

$$x^* = \arg \min_{x \in \Omega} f(x)$$

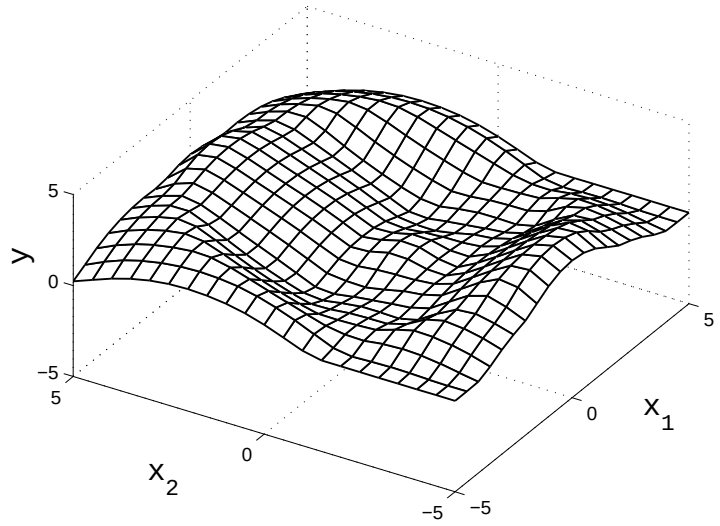
Definice 4.1. *Nechť $c \in \mathbb{R}^1$. Potom množině*

$$\mathcal{L}_c = \{x \in D \ : \ f(x) = c\}$$

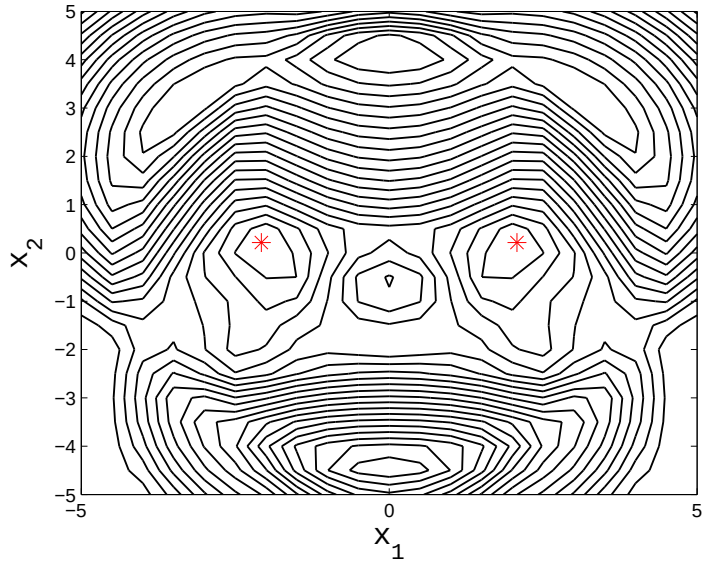
říkáme vrstevnice funkce f.

Jestliže $y \in D$, potom $y \in \mathcal{L}_{f(y)}$.

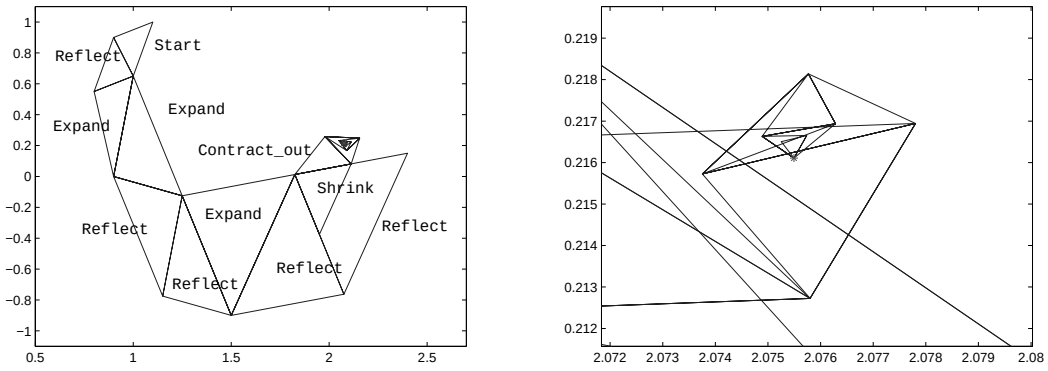
Problém je ilustrován na Obrázku 4.1. Budeme mluvit o numerických metodách lokalizace bodů lokálního minima. Na Obrázku 4.2 jsou některé z vrstevnic, které odpovídají grafu funkce z Obrázku 4.1. Numerická metoda detekovala dvě pozice lokální minima.



Obrázek 4.1: Příklad funkce reálné funkce dvou proměnných $f : \mathbb{R}^2 \to \mathbb{R}^1$. Graf funkce $y = f(x_1, x_2)$.



Obrázek 4.2: Vrstevnice funkce $f : \mathbb{R}^2 \to \mathbb{R}^1$ z Obrázku 4.1. Hvězdička označuje pozici lokálního minima.



Obrázek 4.3: Nelder-Meadův algoritmus v $\mathbb{R}^2$. Historie konvergence (50 kroků algoritmu). Hvězdička $*$ na detailu vpravo označuje pozici lokálního minima.

4.1 Nelder-Meadův algoritmus

Představíme metodu, která se jmenuje *Nelder-Meadův algoritmus* nebo také *emoeba = měňavka*. Metoda generuje posloupnost simplexů v $\mathbb{R}^n$. Příkladem simplexu v $\mathbb{R}^2$ je (nedegenerovaný) trojúhelník, příkladem simplexu v $\mathbb{R}^3$ je (nedegenerovaný) čtyřstěn, viz Obrázek 4.4.

Metodu budeme ilustrovat na příkladu. Budeme aproximovat jedno ze dvou lokálních minim funkce $f = f(x)$, viz Obrázek 4.2. Na Obrázku 4.3 je poslupnost simplexů v $\mathbb{R}^2$. Posloupnost je definována rekurzivně. Začíná simplexem označeným **Start**. V těžisti simplexu **Start** je první aproximace lokálního minima. Nechť S_k je k-tý simplex posloupnosti. Potom simplex S_{k+1} je definován jako právě jedna z transformací

$$\textbf{Reflect}\,,\quad \textbf{Expand}\,,\quad \textbf{Contract\_out}\,,\quad \textbf{Contract\_in}\,,\quad \textbf{Shrink} \tag{4.1}$$

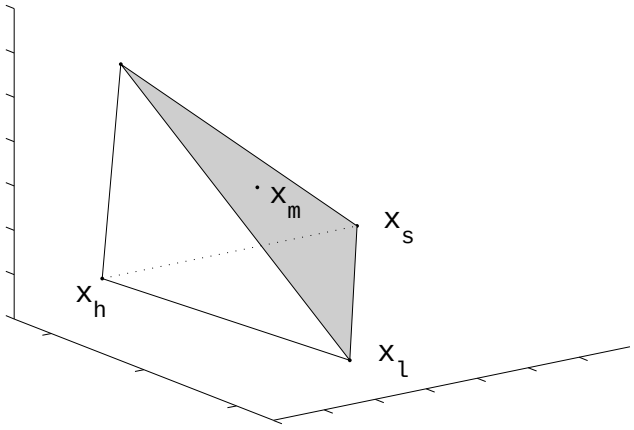
simplexu S_k. Cílem každé z pěti transformací je dosáhnout lokálního poklesu funkce $f = f(x)$. Vybíráme právě takovou, která má největší šanci na úspěch.

V dalším výkladu probereme

- definici simplexu v $\mathbb{R}^n$,
- definici transformací (4.1),
- Algoritmus 4.1 přechodu $S_k \longrightarrow S_{k+1}$.

Definice 4.2. *Uvažujme $n+1$ bodů v prostoru $\mathbb{R}^n$, t.j.*

$$x_1 \in \mathbb{R}^n\,,\ x_2 \in \mathbb{R}^n\,,\ \ldots\,,\ x_n \in \mathbb{R}^n\,,\ x_{n+1} \in \mathbb{R}^n\,.$$



Obrázek 4.4: Příklad simplexu S v $\mathbb{R}^3$. Vrcholy $x_1 \in \mathbb{R}^3$, $x_2 \in \mathbb{R}^3$, $x_3 \in \mathbb{R}^3$, $x_4 \in \mathbb{R}^3$. Předpoklad: $f(x_1) \geq f(x_2) \geq f(x_3) \geq f(x_4)$. Mnemotechnické označení vrcholů: $x_1 \equiv x_h$ ... "high", $x_2 \equiv x_s$ ... "next-to-high", $x_4 \equiv x_l$ ... "low". $x_m = \frac{1}{3}\sum_{j\neq 1} x_j$ ... geometrický střed (centroid, "mean") stěny, která je protilehlá vrcholu x_h.

Nechť jsou afinně nezávislé, t.j. lineární obal vektorů

$$\mathcal{L}\left\{x_2 - x_1\,,\ \ldots\ ,\ x_n - x_1\,,\ x_{n+1} - x_1\right\} = \mathbb{R}^n$$

má dimenzi n. Definujme množinu

$$S = \mathrm{co}\left\{x_1, x_2, \ldots, x_n, x_{n+1}\right\} \tag{4.2}$$

jako konvexní obal vektorů $x_1, x_2, \ldots, x_n, x_{n+1}$. Říkáme, že

1. *S je <u>simplex</u> v $\mathbb{R}^n$,*
2. *$x_i \in S$, $i = 1, \ldots, n+1$, je <u>vrchol</u> simplexu S.*

Připomeňme, že

$$\mathrm{co}\left\{x_1, x_2, \ldots, x_n, x_{n+1}\right\} \equiv \left\{\sum_{i=1}^{n+1} \alpha_i x_i \ :\ \alpha_i \in \mathbb{R}^1\,,\ \alpha_i \geq 0\,,\ \sum_{i=1}^{n+1} \alpha_i = 1\right\}.$$

Každá podmnožina vrcholů $\{y_1, ..., y_{r+1}\} \subset \{x_1, ..., x_{n+1}\}$ definuje <u>stěnu</u> simplexu jakožto $\mathrm{co}\left\{y_1, \ldots, x_{r+1}\right\}$; r definuje dimenzi stěny. Ke každému vrcholu lze přiřadit

"protilehlou" stěnu dimenze $n-1$. Např. protilehlá stěna k vrcholu x_1 je stěna co $\{x_2, ..., x_{n+1}\}$.

Vrcholy simplexu S lze libovolně očíslovat. Bez újmy na obecnosti lze požadovat, aby vrcholy $x_1 \in \mathbb{R}^n$, $x_2 \in \mathbb{R}^n$, ... , $x_n \in \mathbb{R}^n$, $x_{n+1} \in \mathbb{R}^n$ splňovaly

$$f(x_1) \geq f(x_2) \geq \cdots \geq f(x_n) \geq f(x_{n+1})\,.$$

Pro popis algoritmu budou podstatné vrcholy x_1, x_2 a x_{n+1}. Zavedeme jejich mnemotechnické označení:

$$x_1 \equiv x_h\,, \quad x_2 \equiv x_s\,, \quad x_{n+1} \equiv x_l\,,$$

t.j. "high", "next-to-high", "low". Uvažujme stěnu co $\{x_2, \ldots, x_{n+1}\}$, která je protilehlá k vrcholu x_h. Definujme

$$x_m = \frac{1}{n}\sum_{j\neq 1} x_j$$

geometrický střed (centroid, "mean") této stěny, viz Obrázek 4.4.

Nechť S je simplex v $\mathbb{R}^n$. Budeme definovat pět transformací simplexu S. Jednotlivé transformace jsou chápany jako příkazy:

Reflect ... vytvoř zrcadlový obraz

Expand ... expanduj, roztáhni se

Contract_out, **Contract_in** ... stáhni se ven/dovnitř

Shrink ... zcvrkni se

Každá z uvedených transformací bude definovat nový simplex S^{new} v $\mathbb{R}^n$.

- **Reflect** , **Expand** , **Contract_out** , **Contract_in**
  Simplexy S a S^{new} budou mít právě jedinou společnou stěnu, která je protilehlá vrcholu x_h simplexu S. Simplex S^{new} bude tak určen pozicí jediného vrcholu $x^{\text{new}} \in \mathbb{R}^n$. Definujme přímku

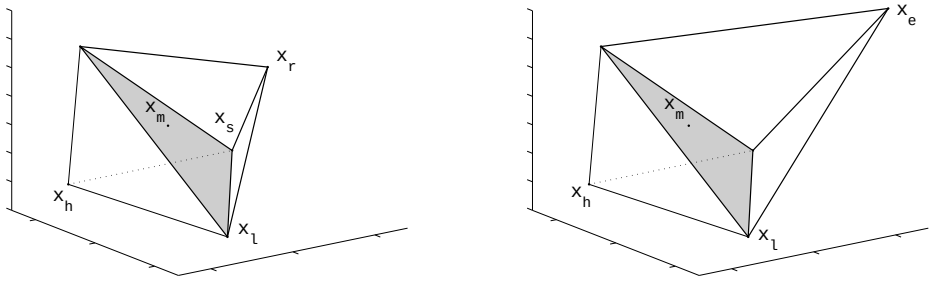
  $$\alpha \in \mathbb{R}^1 \longmapsto x_m + \alpha(x_m - x_h) \in \mathbb{R}^n\,.$$

  Vrcholy $x^{\text{new}} \in \mathbb{R}^n$ budou ležet na této přímce. Konkrétní transformace budou definovány výběrem parametru α:

Reflect :	$\alpha = 1\,,$	$x^{\text{new}} = x_r \equiv x_m + (x_m - x_h)\,,$	viz Obr. 4.5
Expand :	$\alpha = 2\,,$	$x^{\text{new}} = x_e \equiv x_m + 2(x_m - x_h)\,,$	viz Obr. 4.5
Contract_out :	$\alpha = \frac{1}{2}\,,$	$x^{\text{new}} = x_{co} \equiv x_m + \frac{1}{2}(x_m - x_h)\,,$	viz Obr. 4.6
Contract_in :	$\alpha = -\frac{1}{2}\,,$	$x^{\text{new}} = x_{ci} \equiv x_m - \frac{1}{2}(x_m - x_h)\,,$	viz Obr. 4.6

- **Shrink**
  S a S^{new} jsou stejnolehlé s vrcholem x_l a koeficientem $\frac{1}{2}$ t.j. $x \in S$ právě když $x^{\text{new}} = x_l + \frac{1}{2}(x - x_l) \in S^{\text{new}}$. Doplňme tedy tabulku transformací o položku

  Shrink : homotetie $S^{\text{new}} = x_l + \frac{1}{2}(S - x_l)\,,$ viz Obr. 4.7



Obrázek 4.5: a) **Reflect**. Simplexu S je přiřazen nový simplex S_r, který má stejnou stěnu a nový vrchol x_r, který je reflexí x_h. b) **Expand**. Simplexu S je přiřazen nový simplex S_e, který má stejnou stěnu a nový vrchol x_e. Parametr konstrukce: $||x_e - x_m||/||x_m - x_h|| = 2$.

Nyní můžeme formulovat

Algoritmus 4.1. (Nelder-Mead, amoeba = měňavka)
Dáno: Simplex S s vrcholy uspořádanými vrcholy. Nechť x_h, x_s, x_l. Položme $f_h := f(x_h)$, $f_s := f(x_s)$, $f_l := f(x_l)$. Potom

compute x_r, $f_r := f(x_r)$

if $f_l \le f_r < f_s$ *then* **Reflect**

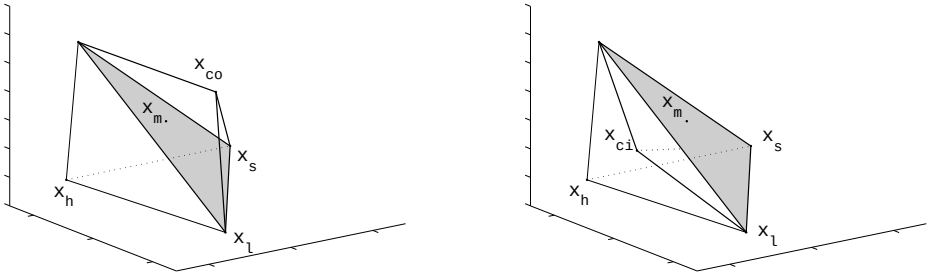
else if $f_r < f_l$, *compute* x_e, $f_e := f(x_e)$

if $f_e < f_r$ *then* **Expand**
else **Reflect**

else if $f_r \ge f_s$

if $f_r < f_h$, *compute* x_{co}, $f_c := f(x_{co})$
if $f_c \le f_r$ *then* **Contract_out**

if $f_r \ge f_h$, *compute* x_{ci}, $f_c := f(x_{ci})$



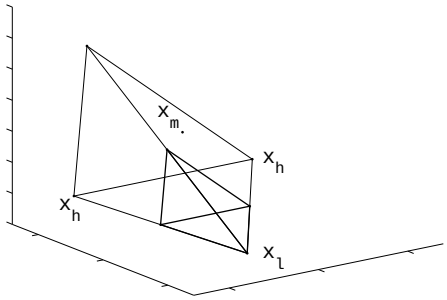
Obrázek 4.6: c) **Contract_out**. Simplexu S je přiřazen nový simplex S_{co}, který má stejnou stěnu a nový vrchol x_{co}. Parametr konstrukce: $||x_{co} - x_m||/||x_m - x_h|| = 1/2$. d) **Contract_in**. Simplexu S je přiřazen nový simplex S_{ci}, který má stejnou stěnu a nový vrchol x_{ci}. Parametr konstrukce: $||x_{ci} - x_m||/||x_m - x_h|| = 1/2$.

if $f_c < f_h$ *then* **Contract_in**

else **Shrink**

Jeden krok této iterační metody se opírá o vyhodnocení nejvýše šesti funkčních hodnot v šesti konkrétních bodech. Na základě této informace, algoritmus hledá nějakou tendenci směrového poklesu funkce $f = f(x)$. Formuluje hypotézu o pozici lokálního minima. Na základě této hypotézy definuje nový simplex. Algoritmus je heuristický. Nepotřebuje počítat gradient. Teoreticky předpokládá pouhou spojitost funkce $f = f(x)$. Za jistých předpokladů lze dokázat lokální konvergenci.

Poznámka 4.1. (terminologická) *Autoři algoritmu jej původně prezentovali jako* simplexovou metodu *pro hledání minima reálné funkce více proměnných ("bez omezení"), viz*
Nelder, J.A. and Mead, R. (1965), "A simplex method for function minimization", Comput. J., 7, pp. 308–313.
Nicméně, pod heslem Simplexová metoda *se v literatuře uvádí metoda* lineárního programování *t.j. metoda maximalizace/minimalizace lineární účelové funkce vzhledem k lineárním omezením.*



Obrázek 4.7: e) **Shrink**. Simplexu S je přiřazen nový simplex S_{sh}, který je stejnolehlý se středem stejnolehlosti v bodě x_l a koeficientem stejnolehlosti $1/2$.

4.2 Metody sestupu

Nechť $f : \mathbb{R}^n \to \mathbb{R}^1$ je reálná funkce s definičním oborem $D \subset \mathbb{R}^n$, D je otevřená množina. Předpokládejme, že f je jednou spojitě diferencovatelná na D. Označení: $f \in C^1(D)$. Nechť $x^* \in D$ je lokální minimum funkce f. Řešíme Problém 4.1.

Nechť $x \in D$ je aproximací x^*. K zadanému vektoru $d \in \mathbb{R}^n$ definujme reálnou funkci $g : \mathbb{R}^1 \longrightarrow \mathbb{R}^1$,

$$g = g(\alpha)\,, \quad g(\alpha) \equiv f(x + \alpha d)\,, \quad \alpha \geq 0\,. \tag{4.3}$$

Definiční obor funkce g je polopřímka

$$x + \alpha d \in \mathbb{R}^n\,, \quad \alpha \geq 0\,.$$

Vektoru $d \in \mathbb{R}^n$ říkáme směr sestupu. Směr sestupu není libovolný. Požadujeme, aby

$$\lim_{\alpha \to 0_+} g'(\alpha) < 0 \tag{4.4}$$

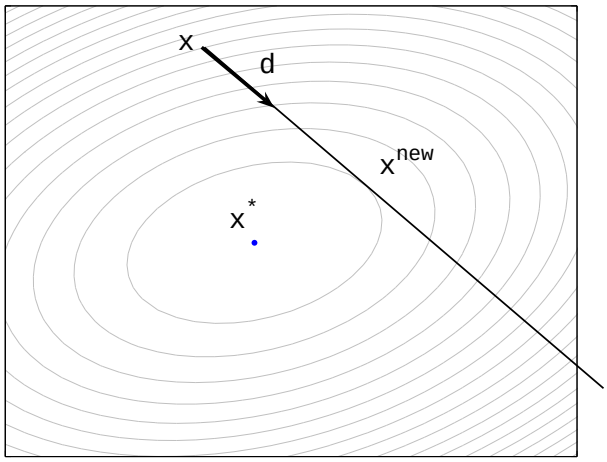
t.j. funkce $g = g(\alpha)$ musí lokálně klesat v okolí počátku $\alpha = 0$. Každá volba parametru $\alpha > 0$, pro kterou $g(\alpha) < g(0)$, bude potom znamenat, že $x^{\text{new}} = x + \alpha d$ je lepší aproximací minima než původní x. Největšího poklesu dosáhneme v bodě lokálního minima funkce g: Hledáme $\alpha_{\min} > 0$ tak, aby $g(\alpha_{\min}) < g(0)$ a aby funkce $g = g(\alpha)$ nabývala v bodě $\alpha_{\min}$ lokálního minima t.j.

$$\alpha_{\min} > 0\,, \quad g(\alpha_{\min}) < g(0)\,, \quad \alpha_{\min} = \arg\min_{\alpha \in \Delta} g(\alpha)\,, \tag{4.5}$$

kde Δ je otevřený interval, $\alpha_{\min} \in \Delta$. Potom definujeme

$$x^{\text{new}} = x + \alpha_{\min} d\,, \tag{4.6}$$

viz Obrázek 4.8&4.9.



Obrázek 4.8: Nechť $f : \mathbb{R}^2 \to \mathbb{R}^1$. Dáno $x \in \mathbb{R}^2$, t.j. aproximace lokálního minima x^*, $d \in \mathbb{R}^2$, t.j. směr sestupu. Definujme restrikci funkce f na polopřímku $\alpha \geq 0 \longmapsto x + \alpha d \in \mathbb{R}^n$. Potom $x^{\text{new}} = x + \alpha_{\min} d$, viz Obrázek 4.9, je nová aproximace x^*.

Poznámka 4.2. *Počítejme $g'(\alpha)$ funkce* (4.3)*. Aplikací pravidla o derivování složené funkce dostaneme, že*

$$g'(\alpha) = \frac{d}{d\alpha}(f(x + \alpha d)) = \sum_{i=1}^{n} \frac{\partial f}{\partial x_i}(x + \alpha d) d_i \,.$$

Potom podmínku (4.4) *na směr sestupu lze ekvivalentně formulovat jako*

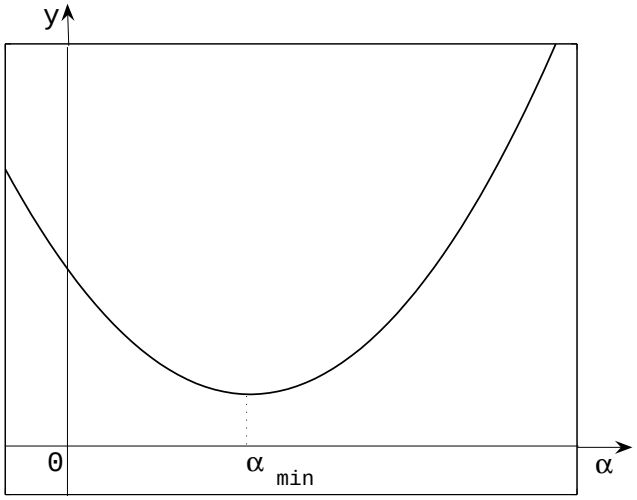
$$g'(0) = \sum_{i=1}^{n} \frac{\partial f}{\partial x_i}(x) d_i < 0 \,. \tag{4.7}$$

Definice 4.3. (Přípustný směr sestupu) *Uvažujme $f \in C^1(D)$, kde $D \subset \mathbb{R}^n$ je otevřená. Nechť $x \in D$. Nechť $d \in \mathbb{R}^n$ je směr sestupu. Říkáme, že d je přípustný směr sestupu v bodě x, jestliže je splněna podmínka* (4.7).

Idea (4.6) vede k formulaci iteračního algoritmu:

Algoritmus 4.2. (Metoda sestupu) *Dáno: $x^{(0)} \in \mathbb{R}^n$, $d^{(0)} \in \mathbb{R}^n$. Nechť $d^{(0)}$ je přípustný směr sestupu v bodě $x^{(0)}$. Definujeme posloupnosti $\left\{x^{(k)}\right\}_{k=0}^{\infty}$ a $\left\{d^{(k)}\right\}_{k=0}^{\infty}$*

- *aproximací $x^{(k)} \in \mathbb{R}^n$ lokálního minima a*
- *přípustných směrů sestupu $d^{(k)} \in \mathbb{R}^n$*



Obrázek 4.9: Graf funkce $y = g(\alpha)$, kde $g(\alpha) \equiv f(x + \alpha d)$. Hledáme pozici lokálního minima $\alpha_{\min} \in \mathbb{R}^1$ funkce $y = g(\alpha)$, $\alpha > 0$.

rekurencí

$$x^{(k+1)} = x^{(k)} + \alpha_k d^{(k)}\,, \tag{4.8}$$

kde α_k *je pozice lokálního minima,*

$$\alpha_k > 0\,, \quad f(x^{(k)} + \alpha_k d^{(k)}) < f(x^{(k)})\,, \quad \alpha_k = \arg\min_{\alpha \in \Delta} f(x^{(k)} + \alpha d^{(k)})\,, \tag{4.9}$$

Δ *je otevřený interval,* $\alpha_k \in \Delta$*. Směr* $d^{(k+1)}$ *splňuje podmínku přípustného sestupu*

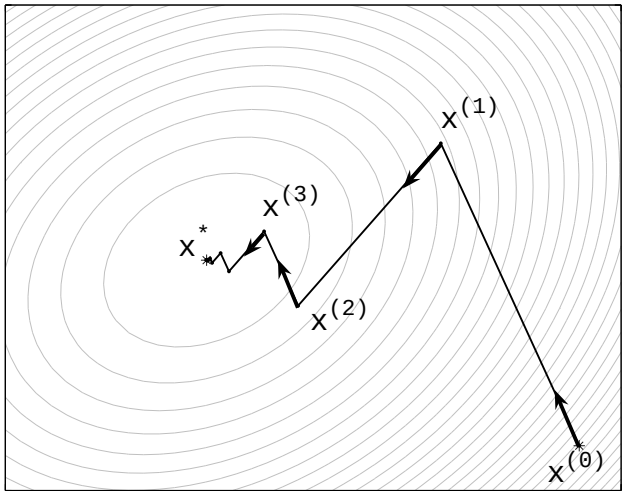
$$\sum_{i=1}^{n} \frac{\partial f}{\partial x_i}(x^{(k+1)}) d_i^{(k+1)} < 0\,. \tag{4.10}$$

Ilustrativní příklad fungování algoritmu je na Obrázku 4.10. Ve formulaci algoritmu zůstávají nezodpovězené dvě otázky:

1. Jak realizovat krok (4.9) t.j., jak najít lokální minimum reálné funkce jedné proměnné.

2. Jak formulovat strategii pro výběr směru sestupu, viz (4.10).

Na tyto otázky odpovíme v odstavcích 4.2.1 a 4.2.2.



Obrázek 4.10: Nechť $f : \mathbb{R}^2 \to \mathbb{R}^1$. Ilustrace fungování Algoritmu 4.2.

4.2.1 "Line-search": Minimalizace funkce jedné proměnné

Uvedeme několik algoritmů, které lze použít pro numerickou realizaci kroku (4.3)&(4.4). Tomuto kroku se říká "Line-search": Hledáme lokální minimum reálné funkce $g(\alpha) \equiv f(x+\alpha d)$ proměnné α na intervalu $\alpha > 0$. Z praktických důvodů hledáme minimum na konečném intervalu. Soustředíme se na algoritmy, které nevyčíslují derivace.

Označení: Objektem našeho zájmu bude reálná funkce f proměnné $x \in \mathbb{R}^1$ na intervalu $[a, b]$.

Problém 4.2. *Uvažujme reálnou funkci $f : \mathbb{R}^1 \longrightarrow \mathbb{R}^1$, která je spojitá na intervalu $[a, b]$, $a < b$. Hledáme pozici x^* lokálního minima na intervalu (a, b) t.j., existuje interval $(c, d) \subset (a, b)$ tak, že*

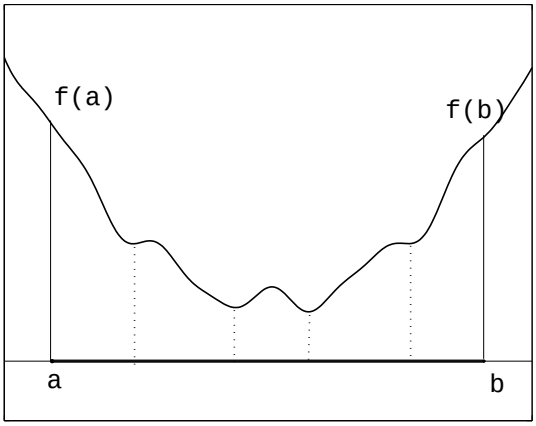
$$x^* = \arg \min_{(c,d) \in x} f(x)\,.$$

Lokálních minim může být víc, viz ilustrační Obrázek 4.11. Problém najít <u>všechna</u> lokální minima je nepoměrně složitější. V souvislosti a krokem "Line-search" se spokojíme zpravidla s nějakým řešením.

Definice 4.4. *Uvažujme funkci $f \in C\,[a, b]$. Nechť existuje $x \in \mathbb{R}^1$ tak, že*

$$a < x < b\,, \quad f(a) > f(x)\,, \quad f(b) > f(x)\,. \tag{4.11}$$

Podmínce (4.11) říkáme <u>uzávorkování minima</u> na intervalu $[a, b]$. Říkáme, že minimum na intervalu $[a, b]$ je uzávorkováno v bodě x.



Obrázek 4.11: Uvažujme reálnou funkci $f : \mathbb{R}^1 \longrightarrow \mathbb{R}^1$ na intervalu $[a, b]$. Cílem je najít jedno z lokálních minim funkce f.

Z předpokladu uzávorkování minima na intervalu $[a, b]$ plyne existence lokálního minima $a < x^* < b$. Odvodíme algoritmus, který lokalizuje jednu z pozic x^*. Tento algoritmus generuje posloupnost zmenšujících se intervalů. Na každém z těchto intervalů je uzávorkováno minimum.

Ukážeme ideu algoritmu symetrické lokalizace:

Algoritmus 4.3. (Symetrická lokalizace) *Nechť minimum na intervalu* $[a, b]$ *je uzávorkováno v bodě* x *t.j. předpokládejme* (4.11). *Položme*

$$\vartheta = \frac{x - a}{b - a}\,. \tag{4.12}$$

Předpokládejme, že $\vartheta \neq 1/2$. *Definujme* $y \in \mathbb{R}^1$ *tak, že*

$$y = a + b - x\,. \tag{4.13}$$

Na základě bodů a, b, x *a* y *definujeme nové uzávorkování minima na intervalu* $[a^{\rm new}, b^{\rm new}]$ *v bodě* $x^{\rm new} \in \mathbb{R}^1$, $a^{\rm new} < x^{\rm new} < b^{\rm new}$, *takto:*

1. *Jestliže* $\vartheta < 1/2$ *a* $f(y) < f(x)$, *potom*

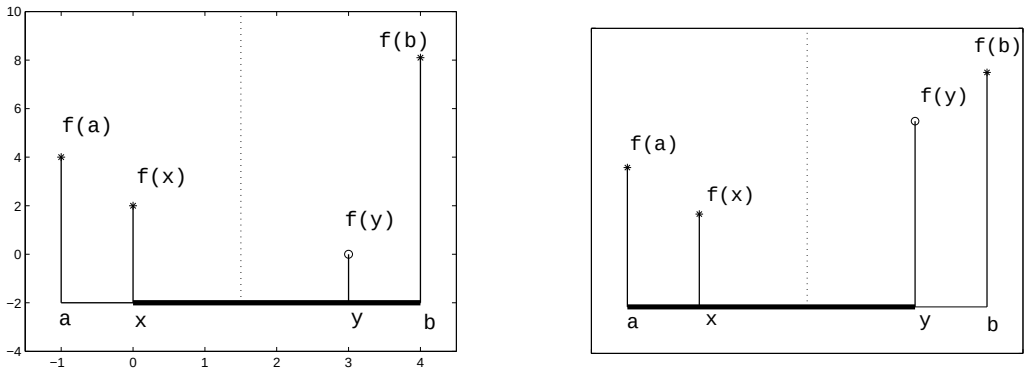
$$a^{\rm new} = x\,, \quad x^{\rm new} = y\,, \quad b^{\rm new} = b\,,$$

   viz Obrázek 4.12, vlevo

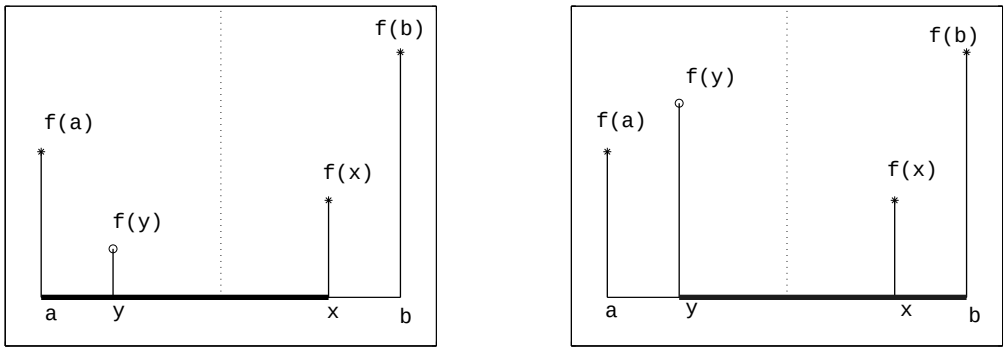
2. *Jestliže* $\vartheta < 1/2$ *a* $f(y) > f(x)$, *potom*

$$a^{\rm new} = a\,, \quad x^{\rm new} = x\,, \quad b^{\rm new} = y\,,$$

   viz Obrázek 4.12, vpravo



Obrázek 4.12: Symetrické prohledávání: $y = a + b - x$, $\vartheta = \frac{x-a}{b-a}$, $\vartheta < 1/2$.



Obrázek 4.13: Symetrické prohledávání: $y = a + b - x$, $\vartheta = \frac{x-a}{b-a}$, $\vartheta > 1/2$.

3. *Jestliže* $\vartheta > 1/2$ *a* $f(y) < f(x)$, *potom*

$$a^{\text{new}} = a\,, \quad x^{\text{new}} = y\,, \quad b^{\text{new}} = x\,,$$

viz Obrázek 4.13, vlevo

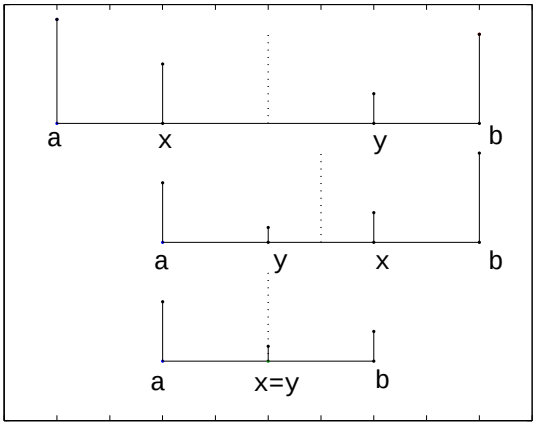
4. *Jestliže* $\vartheta > 1/2$ *a* $f(y) > f(x)$, *potom*

$$a^{\text{new}} = y\,, \quad x^{\text{new}} = x\,, \quad b^{\text{new}} = b\,,$$

viz Obrázek 4.13, vpravo.

Je zřejmé, že

$$(b^{\text{new}} - a^{\text{new}}) = \min(\vartheta, 1-\vartheta)\ (b-a)\,, \tag{4.14}$$



Obrázek 4.14: Nechť $\vartheta = 1/4$, t.j. $(x-a)/(b-a) = 1/4$. Potom algoritmus zkolabuje ve dvou krocích.

Algoritmus definuje nové uzávorkování minima na menším intervalu. Nabízí se iterační použití algoritmu tím, že definujeme aktualizaci intervalu

$$a := a^{\text{new}}\,, \quad x := x^{\text{new}}\,, \quad b := b^{\text{new}}\,.$$

Všimněme si, že algoritmus není definován v případě, že $f(x) = f(y)$. Především, situace kdy v konečné aritmetice zaznamenáme, že $f(x) = f(y)$, je krajně nepravděpodobná. S výjimkou situace, která je popsána v Poznámce 4.3. Nicméně, do algoritmu můžeme případ $f(x) = f(y)$ zakomponovat, a "ošetřit" jej vhodnou heuristikou. Konečně, nepříjemné situaci se lze vyhnout vhodným matematickým předpokladem. Například tím, že budeme požadovat, aby uvažovaná funkce f byla unimodální:

Definice 4.5. *Nechť $f \in C\,[a,b]$. Nechť existuje $a < x^* < b$ tak, že f je klesající na intervalu $[a, x^*)$ a f je rostoucí na intervalu $(x^*, b]$, potom říkáme, že funkce f je unimodální na intervalu $[a,b]$.*

Bod x^* je samozřejmě jediným bodem lokálního minima.

Poznámka 4.3. *Uvedeme příklad, kdy Algoritmus 4.3 nebude fungovat. Aplikujme algoritmus v situaci kdy $\vartheta = 1/4$ t.j. $(x-a)/(b-a) = 1/4$. Na Obrázku 4.14 jsou schématicky znázorněny dvě možné iterace algoritmu. V druhé iteraci dojde k porušení předpokladu $\vartheta \neq 1/2$.*

Proto jako parametr ϑ volíme nějaké iracionální číslo z intervalu $0 < \vartheta < 1$. Během iterací se samozřejně hodnota ϑ mění. Zůstává ale iracionální. Jaká je optimální volba ϑ?

Nechť ϑ^{new} je hodnota parametru v druhém kroku algoritmu. Např., jestliže $\vartheta < 1/2$ a $f(y) < f(x)$, viz Obrázek 4.12 vlevo, potom

$$\vartheta^{\text{new}} = \frac{y-x}{b-x} = \frac{1-2\vartheta}{1-\vartheta}\,. \tag{4.15}$$

Za optimální považujeme, aby $\vartheta = \vartheta^{\text{new}}$. Z (4.15) odvodíme příslušnou podmínku $\vartheta = (1-2\vartheta)/(1-\vartheta)$. Optimální hodnota parametru ϑ je tedy dána řešením kvadratické rovnice $\vartheta^2 - 3\vartheta + 1$ s kořeny $\vartheta = (3 \pm \sqrt{5})/2$. Na intervalu $0 < \vartheta < 1$ existuje jediný kořen

$$\vartheta = \frac{3-\sqrt{5}}{2}\,. \tag{4.16}$$

Poznámka 4.4. (Metoda zlatého řezu) *Algoritmu 4.3 symetrické lokalizace se speciální volbou* $\vartheta = \frac{3-\sqrt{5}}{2}$*, říkáme metoda zlatého řezu. Alternativně lze volit* $\vartheta = 1 - \frac{3-\sqrt{5}}{2}$*. V konečné aritmetice je iracionální parametr* $\frac{3-\sqrt{5}}{2}$ *aproximován číslem* 0.38196601125011.

Proč zlatý řez? Zlatým řezem je označován poměr

$$\frac{a+b}{a} = \frac{a}{b} \equiv \kappa$$

kde $\kappa > 0$. Z příslušné kvadratické rovnice odvodíme, že $\kappa = (1+\sqrt{5})/2$. Od starověku je zlatý řez považován za ideální proporce obdélníka. Uplatňuje se v kompozici renezančních obrazů. V přírodě vystupuje ve formě Fibonacciho posloupnosti. V metodě zlatého řezu

$$\vartheta = \frac{1-\kappa}{\kappa}\,.$$

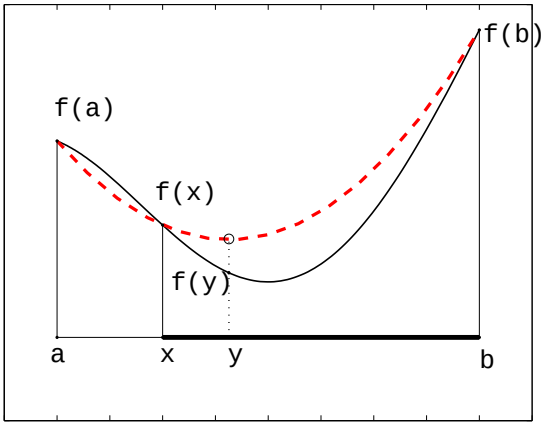
Include: Konvergenční kriterium

Alternativní metodou řešení Problému 4.2 je metoda parabolické interpolace. Idea metody:
Nechť minimum funkce f na intervalu $[a, b]$ je uzávorkováno v bodě x, viz Obrázek 4.15. Uvažujme kvadratickou funkci

$$t \longmapsto h(t) \equiv \alpha + \beta t + \gamma t^2\,,$$

kde α, β a γ jsou koeficienty. Požadujeme, aby

$$\begin{aligned} \alpha + \beta a + \gamma a^2 &= h(a) = f(a)\,, \\ \alpha + \beta x + \gamma x^2 &= h(x) = f(x)\,, \\ \alpha + \beta b + \gamma b^2 &= h(b) = f(b)\,. \end{aligned}$$



Obrázek 4.15:

Jinými slovy, hledáme koeficienty α, β, γ tak, aby

$$\begin{bmatrix} 1 & a & a^2 \\ 1 & x & x^2 \\ 1 & b & b^2 \end{bmatrix} \begin{bmatrix} \alpha \\ \beta \\ \gamma \end{bmatrix} = \begin{bmatrix} f(a) \\ f(x) \\ f(b) \end{bmatrix} .$$

Jestliže $a < x < b$, potom koeficienty jsou určeny jedoznačně jako řešení soustavy lineárních rovnic. Říkáme, že kvadratická funkce h je Lagrangeovou interpolací funkce f v bodech a, x a b, viz Obrázek 4.15. S Lagrangeovou interpolací se setkáme v Kap. 5.

Vzhledem k předpokladu o uzávorkování minima na intervalu $[a, b]$, kvadratická funkce $h(t) = \alpha + \beta t + \gamma t^2$ je konvexní. Proto existuje bod y, $a < y < b$, ve kterém kvadratická funkce h tohoto minima nabývá. Bod y můžeme explicitně určit z podmínky na extrém: $h'(y) \equiv \beta + 2\gamma y = 0$. Tedy

$$y = -\frac{\beta}{2\gamma} .$$

Bod y chápeme jako aproximaci lokálního minima funkce f. V případě, který analyzován na Obrázku 4.15, je minimum funkce f uzávorkováno na menším intervalu $[x, b]$. Lze si jistě představit jinou funkci f na intervalu $[a, b]$, kdy metoda parabolické interpolace identifikuje interval $[a, x]$ a bod y, $a < y < x$, jako pozici minima funkce h.

Algoritmus 4.4. (Metoda parabolické interpolace) *Nechť minimum funkce f na intervalu $[a, b]$ je uzávorkováno v bodě x. Nechť $h(t) = \alpha + \beta t + \gamma t^2$ je Lagrangeova interpolace funkce f v bodech a, x a b. Položme $y = -\beta/2\gamma$. Potom definujeme nové uzávorkování funkce f na intervalu $[a^{\text{new}}, b^{\text{new}}]$, v bodě $x^{\text{new}} = y$:*

Jestliže $y < x$, *potom* $a^{\text{new}} = a$, $b^{\text{new}} = x$.

Jestliže $y > x$, *potom* $a^{\text{new}} = x$, $b^{\text{new}} = b$.

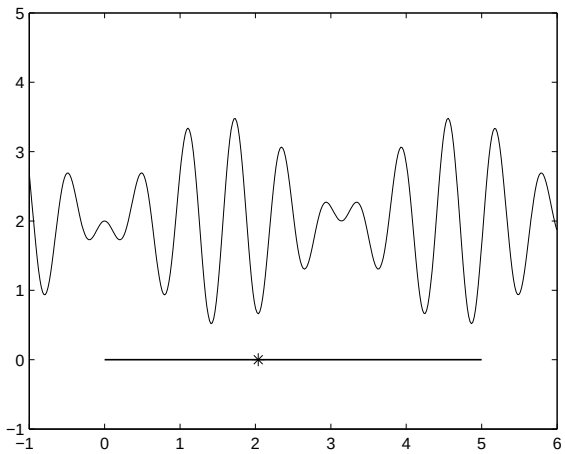
Iterace algoritmu: Aktualizujme $a := a^{\text{new}}$, $x := x^{\text{new}}$, $b := b^{\text{new}}$.

Jestliže funkce f je unimodální, viz Definice 4.5, lze dokázat, že algoritmus konverguje superlineárně. Metoda zlatého řezu konverguje pouze lineárně. Nicméně konverguje i v situaci, kdy existuje více lokálních minim. Je tedy spolehlivá (říkáme, robustní), najde vždy nějaké řešení.

Nabízí se kombinovat metodu zlatého řezu (angl. golden section) s metodou parabolické interpolace (angl. parabolic search). V systému `Matlab` je tato myšlenka implementována prostřednictvím funkce `fminbnd`, která řeší Problém 4.2, viz

`http://www.mathworks.com/help/techdoc/ref/fminbnd.html`

Na Obrázku 4.16 je periodická funkce. Hledáme nějaké lokální minimum na intervalu $[0, 5]$ pomocí funkce `fminbnd`, se zadanou tolerancí 1e-12. Záznam o konvergenci je uveden níže, viz (4.17).



Obrázek 4.16: Uvažujme funkci $f = 1.5\cos(x + \pi/2)\sin(10\,x) + 2$. Hledáme nějaké lokální minimum na intervalu $[0, 5]$ pomocí funkce `fminbnd`, se zadanou tolerancí 1e-12. Hvězdička $*$ označuje pozici lokálního minima.

	x	$f(x)$	Procedure
1	1.90983	1.65174	initial
2	3.09017	2.03792	golden
3	1.18034	2.95859	golden
4	2.35253	3.06382	parabolic
5	1.63119	2.85027	golden
6	2.07893	0.777691	golden
7	2.18344	1.80837	golden
8	2.04142	0.663098	parabolic
9	2.03573	0.661887	parabolic
10	2.03696	0.661777	parabolic
11	2.03701	0.661776	parabolic
12	2.03701	0.661776	parabolic
13	2.03701	0.661776	parabolic
14	2.03701	0.661776	parabolic

(4.17)

4.2.2 Volba směru sestupu

Začneme klíčovým pojmem:

Definice 4.6. *Nechť $f \in C^1(D)$, $x \in D$. Potom*

$$\nabla f(x) \equiv (\frac{\partial f}{\partial x_1}(x), \ldots, \frac{\partial f}{\partial x_n}(x))^{\mathrm{T}} \in \mathbb{R}^n \tag{4.18}$$

je gradient funkce f v bodě x.

Připomeňme, že v celém Odstavci 4.2 předpokládáme, že $f \in C^1(D)$. Jestliže $x^* \in D$ je lokální minimum funkce f t.j., x^* řeší Problém 4.1, potom

$$\nabla f(x^*) = 0 \ \in \mathbb{R}^n \,. \tag{4.19}$$

Každý iterační proces, tedy i Algoritmus 4.2, je třeba ukončit po provedení konečného počtu kroků. Jako konvergenční kriterium můžeme např. požadovat, aby

$$||\nabla f(x^k)|| < \mathrm{tol} \,, \tag{4.20}$$

kde tol je předepsaná tolerance. Za numerické řešení x^k považujeme první člen posloupnosti $\{x^j\}_{j=0}^{\infty}$, který splňuje kriterium (4.20). Jestliže položíme $x^* = x^{(k)}$, potom si musíme uvědomit, že x^* udává pouze aproximativní pozici řešení. Poznamenejme, že stanovení spolehlivých konvergenčních kriterií iteračních procesů je vážný problém.

Nicméně (4.19) je pouze podmínka nutná, aby bod x^* byl řešením Problému 4.1.

Definice 4.7. *Nechť $f \in C^2(D)$, $x \in D$. Potom $\nabla^2 f(x) \in \mathbb{R}^{n\times n}$,*

$$\nabla^2 f(x) \equiv \left[\frac{\partial^2 f(x)}{\partial x_i \partial x_j}\right], \quad i,j = 1,\ldots,n \tag{4.21}$$

je Hessova matice funkce f v bodě x.

Uvažujme Taylorův rozvoj funkce $f = f(x)$ v bodě x^*:

$$f(x) = f(x^*) + \nabla f(x^*)^{\mathrm{T}} (x - x^*) + \frac{1}{2} (x - x^*)^{\mathrm{T}} \nabla^2 f(x^*) (x - x^*) + O(||x - x^*||^3) . \tag{4.22}$$

Druhý člen rozvoje je nulový vzhledem k podmínce (4.19). Třetí člen rozvoje je kvadratická forma

$$x \in \mathbb{R}^n \longmapsto \frac{1}{2} (x - x^*)^{\mathrm{T}} \nabla^2 f(x^*) (x - x^*) \in \mathbb{R}^1 .$$

Formulujme podmínky optimality, viz např. [3], Property 7.4., str. 295:

Poznámka 4.5. *Nechť $f \in C^2(D)$. Jestliže $x^* \in D$ splňuje (4.19) a Hessova matice $\nabla^2 f(x^*)$ funkce f v bodě x^* je positivně definitní, potom x^* je řešením Problému 4.1.*

Algoritmus 4.5. (Metoda největšího spádu) *Dáno: $x^{(0)} \in \mathbb{R}^n$, $d^{(0)} = -\nabla f(x^{(0)}) \in \mathbb{R}^n$. Definujeme posloupnosti $\left\{x^{(k)}\right\}_{k=0}^{\infty}$ a $\left\{d^{(k)}\right\}_{k=0}^{\infty}$ rekurencí*

$$x^{(k+1)} = x^{(k)} + \alpha_k d^{(k)} ,$$

kde $\alpha_k = \arg\min_{\alpha\in\Delta} f(x^{(k)} + \alpha d^{(k)})$ je pozice lokálního minima, viz (4.9), a

$$d^{(k+1)} = -\nabla f(x^{(k+1)}) \in \mathbb{R}^n . \tag{4.23}$$

Je splněna podmínka sestupu? Podmínku sestupu (4.10) lze formulovat pomocí operátoru gradient ∇,

$$\sum_{i=1}^{n} \frac{\partial f}{\partial x_i}(x^{(k+1)}) d_i^{(k+1)} \equiv \left(\nabla x^{(k+1)} , d^{(k+1)} \right) .$$

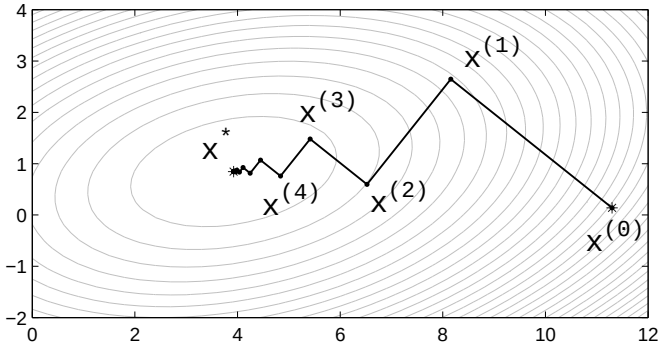
Volbou směru největšího spádu (4.23) požadujeme, aby

$$\left(\nabla x^{(k+1)} , d^{(k+1)} \right) = - \left(\nabla x^{(k+1)} , \nabla x^{(k+1)} \right) = ||\nabla x^{(k+1)}||^2 < 0 .$$

To je zřejmě v každém kroku splněno. V limitě

$$||\nabla x^{(k+1)}||^2 \to ||\nabla x^*||^2 = 0 .$$

Nicméně pokles $||\nabla x^{(k+1)}||^2$ může být velmi pomalý. Obrázek 4.17 ilustruje fungování algoritmu v aplikaci na konkrétní funkci:



Obrázek 4.17: Příklad 4.1, ilustrace fungování Algoritmu 4.5 největšího spádu. Směr sestupu $d^{(k)} \in \mathbb{R}^2$ v bodě $x^{(k)} \in \mathbb{R}^2$ je kolmý k vrstevnici. K dosažení aproximativní pozice minima $x^* = [3.9231;\ 0.8462]$, tol=1.e-5, viz (4.20), bylo třeba dvacetipěti iterací.

Příklad 4.1. *Definujme*

$$x \in \mathbb{R}^2 \longmapsto f(x) \equiv b^{\mathrm{T}} x + \frac{1}{2} x^{\mathrm{T}} A x \ \in \mathbb{R}^1 \,,$$

kde

$$b = \begin{bmatrix} 7 \\ 2 \end{bmatrix} \,, \quad A = \begin{bmatrix} 2 & -1 \\ -1 & 7 \end{bmatrix} \,.$$

Matice A je s.p.d. V Odstavci 8.2 ukážeme, že existuje právě jedno $x^* \in \mathbb{R}^2$, které řeší Problém 4.1. Ukážeme, že x^* je řešením soustavy lineárních rovnic

$$A x^* = b \,.$$

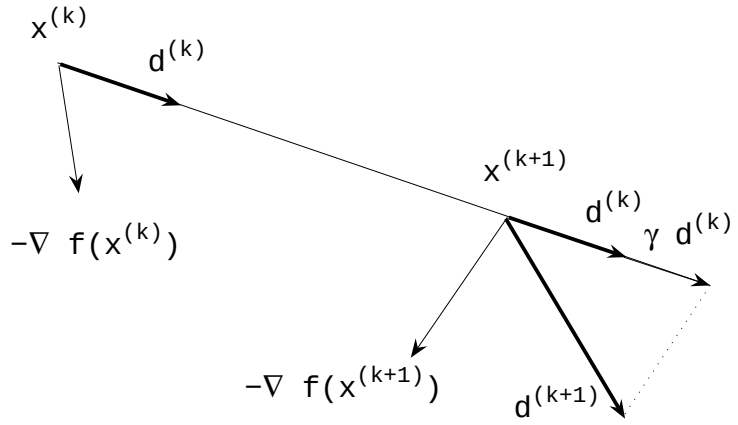
Ve vztahu k rozvoji (4.22)*,*

$$b = f(x^*) \,, \quad \nabla f(x^*) = 0 \in \mathbb{R}^2 \,, \quad A = \nabla^2 f(x^*) \in \mathbb{R}^{2\times 2} \,,$$

a členy vyššího řádu $O(||x - x^*||^3)$ jsou nulové.

Algoritmus 4.6. (Metoda sdružených gradienů) *Dáno: $x^{(0)} \in \mathbb{R}^n$, $d^{(0)} = -\nabla f(x^{(0)}) \in \mathbb{R}^n$. Definujeme posloupnosti $\left\{x^{(k)}\right\}_{k=0}^{\infty}$ a $\left\{d^{(k)}\right\}_{k=0}^{\infty}$ rekurencí*

$$x^{(k+1)} = x^{(k)} + \alpha_k d^{(k)} \,,$$



Obrázek 4.18: Jeden krok metody sdružených gradientů: a) $x^{(k+1)} = x^{(k)} + \alpha_k d^{(k)}$, kde α_k je pozice lokálního minima, viz "line-search". b) $d^{(k+1)} = -\nabla f(x^{(k+1)}) + \gamma d^{(k)}$. Hledáme lineární kombinaci směru největšího spádu s původním směrem $d^{(k)}$.

kde $\alpha_k = \arg\min_{\alpha \in \Delta} f(x^{(k)} + \alpha d^{(k)})$ *je pozice lokálního minima, viz* (4.9)*, a*

$$d^{(k+1)} = -\nabla f(x^{(k+1)}) + \gamma d^{(k)} \in \mathbb{R}^n \,, \tag{4.24}$$

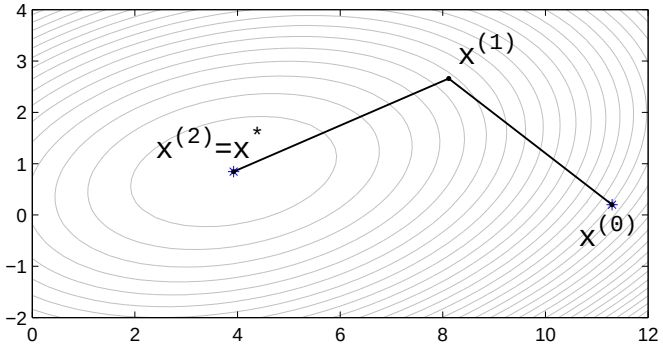
přičemž

$$\gamma = \frac{\left(\nabla f(x^{(k+1)}), \nabla f(x^{(k+1)})\right)}{(\nabla f(x^{(k)}), \nabla f(x^{(k)}))} \,. \tag{4.25}$$

Poznámka 4.6. *Na principu sdružených gradienů je založena řada metod řešení Problému 4.1, např.* Fletcher-Reeves-Polak-Ribierův algoritmus*, [6], str. 321.*

Jeden krok algoritmu lze popsat schematem na Obrázku 4.18. Nový směr sestupu $d^{(k+1)}$ hledáme jako lineární kombinaci směru největšího spádu a původního směru $d^{(k)}$, tedy $d^{(k+1)} = -\nabla f(x^{(k+1)}) + \gamma d^{(k)}$. K volbě parametru γ:

Předpokládejme, že f je kvadratický funkcionál. To znamená, že Taylorův rozvoj (4.22) obsahuje nejvýše kvadratické členy. Potom volba (4.25) parametru γ zaručuje, že metoda sdružených gradientů bude konvergovat v nejvýše n krocích, kde n je dimense problému. V rámci Příkladu 4.1 stačí dva kroky, viz Obrázek 4.19. Podrobné zdůvodnění bude v Kapitole 8. Nicméně finitnost algoritmu sdružených gradientů platí za předpokladu, že výpočet není ovlivněn zaokrouhlovacími chybami. Vliv konečné aritmetiky na fungování algoritmu se prakticky projeví až pro velké dimenze n.



Obrázek 4.19: Příklad 4.1, ilustrace fungování Algoritmu 4.6 sdružených gradientů. Aproximativní pozice minima (tol=1.e-5, viz (4.20)) bylo dosaženo ve dvou krocích.

Za předpokladů, které jsou formulovány v Poznámce 4.5, lze dokázat lokální konvergenci Algoritmu 4.5 i Algoritmu 4.6.

Kapitola 5

Interpolace funkce

Uvažujme spojitou reálnou funkci $y = f(x)$ reálné proměnné x na intervalu $[a, b]$. Předpokládejme, že o této funkci nemáme úplné informace např. známe jenom tabulku některých funkčních hodnot

$$x_i \in [a, b] \longmapsto f(x_i)\,, \quad i = 0, 1, \ldots, n\,.$$

Naším cílem je (přibližná) rekonstrukce funkce v libovolném bodě x z intervalu $[a, b]$.

Budeme specifikovat

1. konečně-dimenzionální podprostory $\mathcal{C}$ prostoru spojitých funkcí, s dimenzí $\dim \mathcal{C} = n + 1$ a bází
$$\mathcal{C} = \operatorname{span}\{v_0, v_1, \ldots, v_n\}\,, \tag{5.1}$$
2. množiny
$$\{x_i\}_{i=0}^{n}\,, \quad x_i \in [a, b]\,, \quad i = 0, 1, \ldots, n \tag{5.2}$$
interpolačních uzlů.

Řešíme následující

Problém 5.1. *Pro zadané interpolační uzly $\{x_i\}_{i=0}^{n}$ hledáme $g \in \mathcal{C}$ tak, aby*

$$g(x_i) = f(x_i)\,, \quad i = 0, 1, \ldots, n\,. \tag{5.3}$$

Podmínce (5.3) *se říká interpolační podmínka.*

5.1 Lagrangeův interpolační polynom

Uvažujme lineární prostor

$$\mathcal{C} = \operatorname{span}\{1, x, \ldots, x^n\} \tag{5.4}$$

všech polynomů stupně nejvýše n. Tento prostor $\mathcal{C}$ budeme nadále označovat symbolem $\mathbb{P}_n$. Uvažujme $n+1$ navzájem různých interpolačních uzlů (5.2). Bez újmy na obecnosti předpokládejme, že

$$a \leq x_0 < x_1 < \cdots < x_n \leq b\,. \tag{5.5}$$

Ukážeme, že existuje právě jeden polynom $g \in \mathbb{P}_n$, který splňuji interpolační podmínku (5.3): Nechť

$$g(x) = a_0 + a_1 x + \cdots + a_n x^n \;=\; \sum_{i=0}^{n} a_i x^i$$

je hledaný polynom. Určíme jeho koeficienty $a_0, a_1, \ldots, a_n$.

Z interpolačních podmínek

$$g(x_i) = a_0 + a_1 x_i + a_2 x_i^2 + \cdots + a_n x_i^n \;=\; f(x_i)\,, \quad i = 0, 1, \ldots, n$$

sestavíme soustavu lineárních rovnic pro koeficienty $a_0, a_1, \ldots, a_n$:

$$\begin{bmatrix} 1 & x_0 & x_0^2 & \ldots & x_0^n \\ \vdots & & & & \\ 1 & x_i & x_i^2 & \ldots & x_i^n \\ \vdots & & & & \\ 1 & x_n & x_n^2 & \ldots & x_n^n \end{bmatrix} \begin{bmatrix} a_0 \\ \vdots \\ a_i \\ \vdots \\ a_n \end{bmatrix} = \begin{bmatrix} f(x_0) \\ \vdots \\ f(x_i) \\ \vdots \\ f(x_n) \end{bmatrix} \tag{5.6}$$

Matici soustavy lineárních rovnic (5.6) se říká Vandermondova matice. Tuto matici označme $V \in \mathbb{R}^{(n+1)\times(n+1)}$. Indukcí vzhledem k n se dá ukázat, že determinant matice V

$$\det V \;=\; \prod_{0 \leq j < i \leq n} (x_i - x_j)\,.$$

Matice soustavy je tedy regulární a tedy koeficienty $a_0, a_1, \ldots, a_n$ jsou jednoznačně určeny. Soustavu lze řešit eliminací.

Polynomu $g \in \mathbb{P}_n$, který splňuje interpolační podmínku (5.3), se říká Lagrangeův interpolační polynom. Zavedeme operátor P, který funkci f, definované v zadaných uzlech (5.5) přiřazuje Lagrangeův interpolační polynom g:

$$g \;=\; P(f; x_0, x_1, \ldots, x_n)\,. \tag{5.7}$$

Existenci a jednoznačnost Lagrangeova polynomu $g = P(f; x_0, x_1, \ldots, x_n)$ lze dokázat i jiným způsobem. V prostoru $\mathbb{P}_n$ zavedeme novou bázi. Definujme

$$L_i(x) = \frac{(x - x_0)\cdots(x - x_{i-1})(x - x_{i+1})\cdots(x - x_n)}{(x_i - x_0)\cdots(x_i - x_{i-1})(x_i - x_{i+1})\cdots(x_i - x_n)}$$

tedy

$$L_i(x) = \prod_{\substack{j=0 \\ j \neq i}}^{n} \frac{x - x_j}{x_i - x_j}\,, \quad i = 0, 1, \ldots, n\,. \tag{5.8}$$

Funkce $L_i = L_i(x)$ jsou polynomy n-tého stupně, t.j. $L_i \in \mathbb{P}_n$ pro $i = 0, 1, \ldots, n$. Z definice

$$L_i(x_j) \;=\; \delta_{ij}\,, \quad i, j = 0, 1, \ldots, n\,. \tag{5.9}$$

Hledáme polynom $g \in \mathbb{P}_n$, který splňuje interpolační podmínku (5.3). Definujme

$$g(x) \;=\; \sum_{i=0}^{n} f(x_i)\, L_i(x)\,. \tag{5.10}$$

Vzhledem k vlastnosti (5.9), polynom g splňuje interpolační podmínku. Je určen jednoznačně. Sporem: Nechť jiný polynom $\tilde{g} \in \mathbb{P}_n$ splňuje interpolační podmínku. Definujme polynom $h = g - \tilde{g}$. Tento polynom stupně n má $n+1$ kořenů $h(x_i) = 0$, $i = 0, 1, \ldots, n$. V důsledku Základní věty algebry, polynom h musí být identicky nulový. Tedy $g = \tilde{g}$, což je spor. Je nyní zřejmé, že polynomy $L_i \in \mathbb{P}_n$, $i = 0, 1, \ldots, n$ tvoří bázi prostoru $\mathbb{P}_n$:

$$\mathbb{P}_n \;=\; \operatorname{span}\{L_0, L_1, \ldots, L_n\}\;. \tag{5.11}$$

Této bázi se říká Lagrangeova báze.

Budeme směřovat k odhadu interpolační chyby. Opřeme se o následující tvrzení:

Věta 5.1. *Nechť $f \in C^{n+1}[a,b]$. Uvažujme Lagrangeův interpolační polynom $g = P(f; x_0, \ldots, x_n)$. Potom $\forall\, x \in [a,b] \quad \exists\, \xi \in (a,b)$ tak, že*

$$f(x) - g(x) = \frac{f^{(n+1)}(\xi)}{(n+1)!}\; \omega_{n+1}(x)\,, \tag{5.12}$$

kde

$$\omega_{n+1}(x) = (x - x_0)(x - x_1)\cdots(x - x_n) \in \mathbb{P}_{n+1}\,. \tag{5.13}$$

Důkaz Jestliže $x = x_i$, $i = 0, \ldots n$, potom tvrzení platí vzhledem (5.13). Zvolme libovolné $x \in [a,b]$, $x \neq x_i\,,\ i = 0, \ldots n$. Ukážeme, že platí (5.12).

Definujme pomocnou funkci $t \longmapsto F(t)$,

$$F(t) = f(t) - g(t) - \frac{f(x) - g(x)}{\omega_{n+1}(x)}\, \omega_{n+1}(t)\,.$$

Protože $f \in C^{n+1}[a,b]$, potom $F \in C^{n+1}[a,b]$. Funkce F má určitě tyto kořeny:

$$F(x) = 0\,, \quad F(x_i) = 0\,,\ i = 0, \ldots, n\;.$$

Indukcí dojdeme k závěru, že

funkce $F : [a,b] \longrightarrow \mathbb{R}^1$	má na intervalu (a,b) alespoň n+2 kořenů
funkce $F' : [a,b] \longrightarrow \mathbb{R}^1$	má na intervalu (a,b) alespoň n+1 kořenů
$\vdots$	
funkce $F^{(n)} : [a,b] \longrightarrow \mathbb{R}^1$	má na intervalu (a,b) alespoň n+2-n = 2 kořeny
funkce $F^{(n+1)} : [a,b] \longrightarrow \mathbb{R}^1$	má na intervalu (a,b) alespoň n+2-n-1 = 1 kořen

Tedy existuje $a < \xi < b$ tak, že $F^{(n+1)}(\xi) = 0$. Z definice funkce F,

$$0 = F^{(n+1)}(\xi) = f^{(n+1)}(\xi) - g^{(n+1)}(\xi) - \frac{f(x) - g(x)}{\omega_{n+1}(x)}\,\omega_{n+1}^{(n+1)}(\xi)\,.$$

Protože $g \in \mathbb{P}_n$, potom derivace $g^{(n+1)}(t) = 0$ v každém bodě t. Konečně ukážeme, že $\omega_{n+1}^{(n+1)}(t) = (n+1)!$ v každém bodě t: Z definice (5.13),

$$\omega_{n+1}(t) = (t - x_0)(t - x_1)\cdots(t - x_n) = t^{n+1} + a_n t^n + \cdots + a_0\,.$$

Derivaci $\omega_{n+1}^{(n+1)}(t)$ ovlivní pouze monomiál nejvyššího stupně t.j. t^{n+1}. Proto

$$\omega_{n+1}^{(n+1)}(t) = (n+1)n(n-1)\cdots 1 = (n+1)!\,.$$

Lze tedy shrnout, že

$$0 = f^{(n+1)}(\xi) - \frac{f(x) - g(x)}{\omega_{n+1}(x)}\,(n+1)!\,.$$

Po úpravě dostaneme formuli (5.12). □

Naším cílem je odhad interpolační chyby. Ve Větě 5.1 předpokládáme, že $f \in C^{n+1}[a,b]$. Definujme

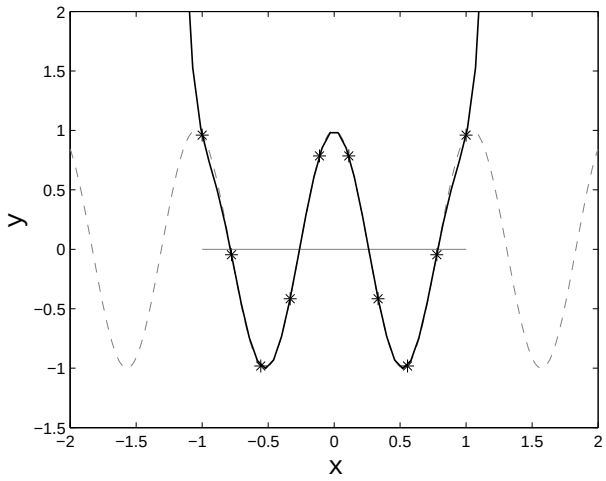
$$K = \max_{a \le x \le b} |f^{(n+1)}(x)|\,. \tag{5.14}$$

Definice 5.1. *K zadaným $n+1$ interpolačním uzlům $a \le x_0 < x_1 < \cdots < x_n \le b$, viz* (5.5)*, definujme* uzlový polynom

$$\omega_{n+1}(x) \;=\; \prod_{i=0}^{n}(x - x_i) \;\in \mathbb{P}_{n+1}\,. \tag{5.15}$$

Je to samozřejmě stejný polynom jako polynom (5.5) ve formulaci Věty 5.1. Polynom ω_{n+1} je tzv. monický polynom, který je charakterizován tím, že koeficient u nejvyšší mocniny je jednička.

Konečně, budeme potřebovat definovat jistou normu reálné funkce:



Obrázek 5.1: Příklad Lagrangeovy interpolace funkce $y = \cos(6x)$ na intervalu $[-1, 1]$, $n + 1 = 10$ ekvidistantních interpolačních uzlů.

Definice 5.2. *Nechť $h \in C[a, b]$. Potom*

$$||h||_\infty = \max_{a \le x \le b} |h(x)|$$

Z Věty 5.1 plyne, že

$$|f(x) - g(x)| \le \frac{K}{(n+1)!} |\omega_{n+1}(x)| \qquad \forall\, a \le x \le b\,.$$

Proto

$$|f(x) - g(x)| \le \frac{K}{(n+1)!} ||\omega_{n+1}||_\infty \qquad \forall\, a \le x \le b\,,$$

a tedy

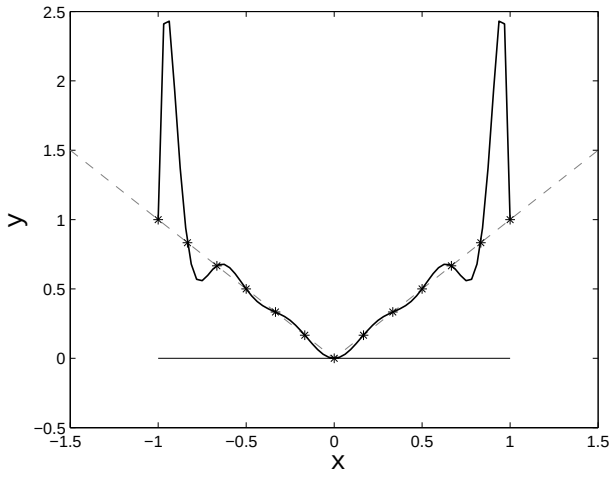
$$||f - g||_\infty \le \frac{K}{(n+1)!} ||\omega_{n+1}||_\infty \tag{5.16}$$

Odhad (5.16) naznačuje, že může záležet na rozložení interpolačních uzlů. Nejprve definujme, co jsou to rovnoměrně rozložené interpolační uzly:

Definice 5.3. *Interpolačním uzlům $\{x_i\}_{i=0}^n$ na intervalu $[a, b]$, které splňují podmínku*

$$x_i = a + i\,\frac{b-a}{n}\,, \quad i = 0, 1, \ldots, n\,, \tag{5.17}$$

říkáme ekvidistantní interpolační uzly.



Obrázek 5.2: Příklad Lagrangeovy interpolace funkce $y = \mathrm{abs}(x)$ na intervalu $[-1, 1]$, $n + 1 = 13$ ekvidistantních interpolačních uzlů.

Lagrangeova interpolace je definována pro každou spojitou funkci na konečném intervalu. Na Obrázcích 5.1 a 5.2 uvádíme dva kvalitativně odlišné příklady interpolace na ekvidistantním dělení. V prvním případě interpolujeme hladkou funkci. S rostoucím počtem uzlů očekáváme menší interpolační chybu. V druhém případě interpolovaná funkce má nespojitou derivaci. Interpolační chyba dramaticky osciluje paradoxně daleko od bodu nespojitosti derivace. S rostoucím počtem uzlů se situace zhoršuje.

Vraťme se k otázce rozložení interpolačních uzlů, tedy k otázce rozložení kořenů uzlového polynomu $\omega_{n+1} \in \mathbb{P}_{n+1}$, viz (5.15). Hledáme optimální rozložení interpolačních uzlů

$$a \leq x_0^{\mathrm{opt}} < x_1^{\mathrm{opt}} < \cdots < x_n^{\mathrm{opt}} \leq b\,, \tag{5.18}$$

které odpovídá kořenům příslušného optimálního uzlového polynomu

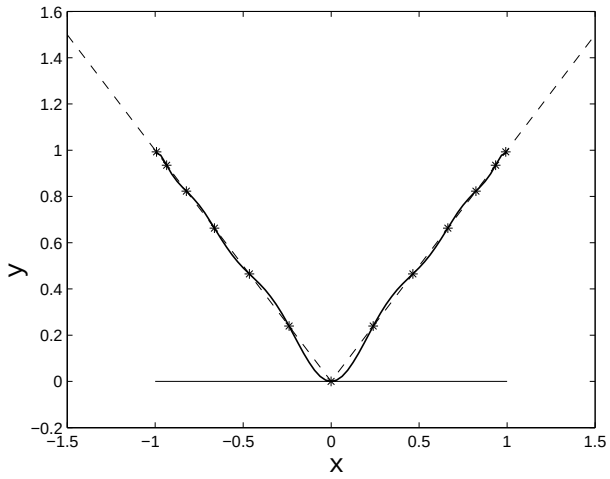
$$\omega_{n+1}^{\mathrm{opt}}(x) \;=\; \prod_{i=0}^{n}(x - x_i^{\mathrm{opt}}) \;\in \mathbb{P}_{n+1}\,.$$

Požadujeme, aby

$$||\omega_{n+1}^{\mathrm{opt}}||_\infty \leq ||\omega_{n+1}||_\infty \tag{5.19}$$

pro každý uzlový polynom ω_{n+1}, (5.15).

V následujícím odstavci problém (5.19) vyřešíme. Předesíláme, že optimální jsou tzv. Čebyševovy interpolační uzly. Na Obrázku 5.3 je příklad jejich použití. Nabízí se srovnání s Obrázkem 5.2.



Obrázek 5.3: Příklad Lagrangeovy interpolace funkce $y = \mathrm{abs}(x)$ na intervalu $[-1, 1]$, $n + 1 = 13$ Čebyševových interpolačních uzlů.

Závěr odstavce věnujeme problému vyčíslení hodnoty Lagrangeova interpolačního polynomu.

- Nejprve vypočítáme koeficienty $a_0, a_1, \ldots, a_n$ Lagrangeova interpolačního polynomu jako řešení soustavy lineárních rovnic (5.6). To je základní investice výpočtu.
- Pro každou zvolenou hodnotu nezávisle proměnné x z intervalu $[a, b]$ (v Obrázcích 5.1–5.3 bylo zvoleno 30 až 50 bodů) je nutné vyčíslit

$$y = \sum_{i=0}^{n} a_i \, x^i \,. \tag{5.20}$$

Výraz (5.20) je zapsán jako lineární kombinace mocnin proměnné x. Vyčíslení v této formě není efektivní. Ten samý výraz lze zapsat šikovněji pomocí tzv. Hornerova schematu. Algoritmus si ukážeme na příkladu polynomu třetího stupně

$$y = a_0 + a_1 x + a_2 x^2 + a_3 x^3 \,.$$

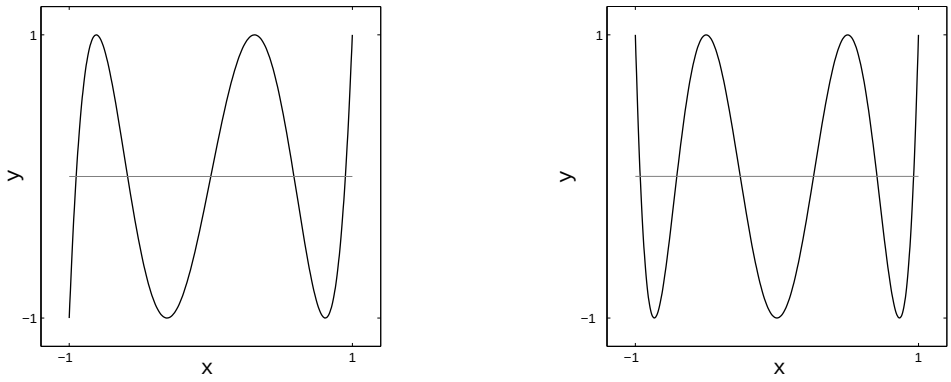
Tento výraz je ekvivalentní výrazu

$$y = a_0 + x(a_1 + x(a_2 + x a_3)) \,.$$

Položíme $b := a_3$. Proměnnou b budeme postupně aktualizovat:

$$b := a_2 + x\,b \,, \quad b := a_1 + x\,b \,, \quad y = a_0 + x\,b \,.$$

Lze dokázat, že Hornerovo schéma je optimálním algoritmem pro vyčíslení reálných polynomů.



Obrázek 5.4: Vlevo, funkce $y = T_5(x)$. Vpravo, funkce $y = T_6(x)$.

5.2 Čebyševovy polynomy

Začneme tím, že bez důkazu uvedeme formuli pro optimálně rozložené interpolační uzly:

Věta 5.2. *Řešením problému* (5.18)&(5.19) *jsou uzly*

$$x_i^{\text{opt}} = \frac{1}{2}\left((a+b) + (b-a)\,\cos\left(\frac{2i+1}{2n+2}\,\pi\right)\right), \quad i = 0, 1, \ldots, n\,. \tag{5.21}$$

K důkazu se vrátíme v závěru odstavce.

Definice 5.4. *Definujme funkci*

$$T_n(x) \;\equiv\; \cos(n\,\arccos x)\,, \qquad -1 \le x \le 1\,. \tag{5.22}$$

Dva příklady jsou na Obrázku 5.4.

Lemma 5.1. *Definiční obor funkce $T_n = T_n(x)$ lze rozšířit na $\mathbb{R}^1$ tak, že rozšířená funkce T_n je polynom stupně n t.j. $T_n \in \mathbb{P}_n$.*

Důkaz Uvažujme substituci

$$T_n(x) = \cos ny\,, \quad y = \arccos x\,, \quad -1 \le x \le 1\,, \quad 0 \le y \le \pi\,. \tag{5.23}$$

Upravujme funkci $\cos ny$, $0 \le y \le \pi$:

$$\cos ny = \Re(\cos ny + \mathrm{i}\sin ny) = \Re(\mathrm{e}^{\mathrm{i}ny}) = \Re(\mathrm{e}^{\mathrm{i}y})^n\,, \tag{5.24}$$

kde $\Re$ je reálná část komplexní funkce. Podle binomické věty

$$(\mathrm{e}^{\mathrm{i}y})^n = (\cos y + \mathrm{i}\sin y)^n \;= \sum_{k=0}^{n} \binom{n}{k} (\cos y)^{n-k} (\mathrm{i}\sin y)^k\,.$$

Vzhledem k (5.24) nás zajímá reálná část binomického rozvoje, tedy sudé indexy $k = 2l$. Upravme sudé mocniny:

$$(\mathrm{i}\sin y)^{2l} = (-1)^l \left((\sin y)^2\right)^l = (-1)^l \left(1 - (\cos y)^2\right)^l .$$

Proto

$$\cos ny = \sum_{\ell=0}^{[\frac{n}{2}]} (-1)^\ell \binom{n}{2\ell} (\cos y)^{n-2\ell} (1 - (\cos y)^2)^\ell .$$

Vrátíme se k původní proměnné x, viz (5.23). Potom funkce T_n,

$$T_n(x) = \sum_{\ell=0}^{[\frac{n}{2}]} (-1)^\ell \binom{n}{2\ell} x^{n-2\ell} (1 - x^2)^\ell , \tag{5.25}$$

původně definovaná na intervalu $[-1, 1]$, je polynom stupně n. □

Shrňme vlastnosti Čebyševových polynomů.

Lemma 5.2. *Platí:*

(i) *Polynom*

$$T_n(x) = a_n x^n + a_{n-1} x^{n-1} + \cdots + a_0$$

má celočíselné koeficienty a_i, speciálně $a_n = 2^{n-1}$

(ii) *Polynomy T_{2n+1} jsou liché funkce, polynomy T_{2n} jsou sudé funkce t.j.*

$$T_{2n+1}(-x) = -T_{2n+1}(x) , \quad T_{2n}(-x) = T_{2n}(x)$$

(iii) $T_n(1) = 1 , \quad T_n(-1) = (-1)^n$

(iv) $|T_n(x)| \le 1 \qquad \forall\ -1 \le x \le 1$

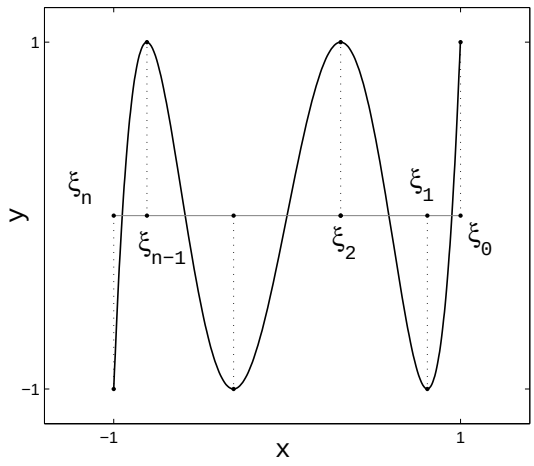
(v) $|T_n(x)| = 1 \quad \Longleftrightarrow \quad x = \cos \dfrac{k\pi}{n} , \ k = 0, \ldots, n$

(vi) $T_n(x) = 0 \quad \Longleftrightarrow \quad x = \cos \dfrac{2k-1}{2n}\pi , \ k = 1, \ldots, n$

Důkaz ad(i): plyne z (5.25), ad(ii)–(iv): je třeba využít substituce (5.23) □

V třídě polynomů $g(x) = a_n x^n + a_{n-1} x^{n-1} + \cdots + a_0$, které jsou škálovány podmínkou $a_n = 2^{n-1}$, je Čebyševův polynom T_n jediný polynom, který se vejde do krabičky s rozměry $[-1, 1] \times [-1, 1]$:

Věta 5.3. *Nechť $g \in \mathbb{P}_n$ je polynom $g(x) = a_n x^n + a_{n-1} x^{n-1} + \cdots + a_0$, pro který $a_n = 2^{n-1}$. Potom existuje $-1 \le \xi \le 1$ tak, že $|g(\xi)| \ge 1$. Jestliže $|g(x)| \le 1$ pro každé $-1 \le x \le 1$, potom $g = T_n$.*



Obrázek 5.5: K důkazu Věty 5.3: Střídání znamének extremálních bodů $1 = \xi_0 > \xi_1 > \ \ldots \ > \xi_n = -1$.

Důkaz Sporem : Nechť $|g(x)| < 1$ pro každé $-1 \le x \le 1$. Definujme $h(x) \equiv T_n(x) - g(x)$. Funkce h je polynom. Protože polynomy T_n a g mají stejný koeficient nejvyšší mocniny, potom

$$h = T_n - g \in \mathbb{P}_{n-1}\,. \tag{5.26}$$

Uvažujme extrémy Čebyševova polynomu T_n, viz Lemma 5.2(v). Platí:

$$|T_n(\xi_i)| = 1\,, \quad 1 = \xi_0 > \xi_1 > \ \ldots \ > \xi_n = -1\,,$$

viz Obrázek 5.5. Dále je si třeba uvědomit, že předpokládáme $|g(\xi_i)| < 1$, $i = 0, \ldots, n$. Proto posloupnost $\{h(\xi_i)\}_{i=0}^n$ střídá znaménko:

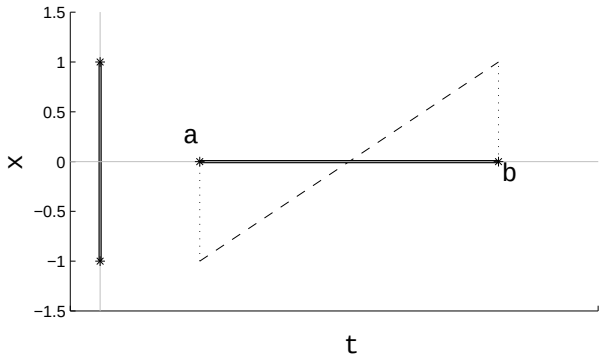
$$\begin{array}{rcll} T_n(\xi_0) & = & 1\,, & h(\xi_0) = T_n(\xi_0) - g(\xi_0) > 0 \\ T_n(\xi_1) & = & -1\,, & h(\xi_1) = T_n(\xi_1) - g(\xi_1) < 0 \\ T_n(\xi_2) & = & 1\,, & h(\xi_2) = T_n(\xi_2) - g(\xi_2) > 0 \\ & \vdots & & \end{array}$$

To znamená, že v každém z intervalů (ξ_i, ξ_{i+1}), $i = 0, 1, \ldots, n-1$ má polynom h alespoň jeden kořen. Celkem tedy alespoň $\underline{n}$ kořenů. Ale h je polynom stupně nejvýše $n-1$, viz (5.26). To je možné, jedině když $h \equiv 0$, tedy $T_n = g$. To je ve sporu s předpokladem $|g(x)| < 1$ pro každé $-1 \le x \le 1$.

Druhá část tvrzení, t.j. jednoznačnost $g = T_n$, se opět ukáže sporem. □

Uvažujme kořeny polynomu T_n, viz Lemma 5.2(vi), s novým označením a očíslováním:

$$x_j^{\rm opt} = \cos\,\frac{2j+1}{2n}\pi\,, \quad j = 0, 1, \ldots, n-1\,, \tag{5.27}$$



Obrázek 5.6: Lineární transformace intervalů $[-1,1]$ a $[a,b]$.

$$-1 \le x_0^{\mathrm{opt}} < x_1^{\mathrm{opt}} < \cdots < x_{n-1}^{\mathrm{opt}} \le 1\,. \tag{5.28}$$

Definujme $\omega_n^{\mathrm{opt}} \in \mathbb{P}_n$,

$$\omega_n^{\mathrm{opt}}(x) = \prod_{j=0}^{n-1} (x - x_j^{\mathrm{opt}})\,.$$

Poznamenejme, že $T_n = 2^{n-1}\,\omega_n^{\mathrm{opt}}$, viz Lemma 5.2(i).

Dále uvažujme libovolnou posloupnost interpolačních uzlů $\{x_j\}_{j=0}^{n-1}$,

$$-1 \le x_0 < x_1 < \cdots < x_{n-1} \le 1\,, \tag{5.29}$$

a příslušný uzlový polynom $\omega_n \in \mathbb{P}_n$

$$\omega_n(x) = \prod_{j=0}^{n-1} (x - x_j)\,.$$

Položme $g = 2^{n-1}\,\omega_n$. Polynom g vystupuje v předpokladech Věty 5.3. Z citované věty snadno plyne, že

$$||\omega_n^{\mathrm{opt}}||_\infty \le ||\omega_n||_\infty \tag{5.30}$$

pro každý uzlový polynom ω_n s uzly, které splňují požadavky (5.29).

Definujme lineární transformaci

$$x \longmapsto \frac{1}{2}\,((a+b) + (b-a)\,x) \tag{5.31}$$

intervalu $[-1, 1]$ na interval $[a, b]$, viz Obrázek 5.6. Lze ověřit, že lineární transformací (5.31) optimálních uzlů (5.27) dostaneme optimální uzly na intervalu $[a, b]$.

Vraťme se k důkazu Věty 5.2. Věta se týká podmínek optimality uzlových polynomů $\omega_{n+1} \in \mathbb{P}_{n+1}$ na intervalu $[a, b]$. Adaptujeme postup (5.27)–(5.31), které jsme použili při analýze uzlových polynomů $\omega_n \in \mathbb{P}_n$.

5.3 Kubický spline

Zformulujeme další interpolační úlohu (5.1)&(5.3). Půjde o interpolace typu spline. Interpolační funkce jsou vybírany z prostorů po částech hladkých funkcí. Jako typickou ukázku této techniky budeme definovat kubický B-spline.

Poznámka 5.1. (Původ slova spline) *V lodním stavitelství se při rýsování plánů používalo pomůcek (elastických šablon), kterým se říkalo anglicky splines. Tedy spline (jednotné č.), splines (množné č.). Zadanou množinou bodů (t.j. interpolačních uzlů) byly "prokládány" elastické šablony. Z matematického hlediska, šablony modelovaly nějaký typ elastického nosníku.*

Na intervalu $[a, b]$ budeme uvažovat $n + 1$ interpolačních uzlů

$$a = x_0 < x_1 < \cdots < x_n = b\,. \tag{5.32}$$

Vyhnene se konstrukci báze příslušného lineárního prostoru kubických B-splinů, (5.1). Budeme přímo definovat operátor $S(f; x_0, x_1, \ldots, x_n)$, který datům interpolační úlohy t.j. zadaným uzlům (5.32) a zadaným funkčním hodnotám

$$y_i = f(x_i)\,, \quad i = 0, 1, \ldots, n \tag{5.33}$$

přiřadí reálnou funkci

$$g = S(f; x_0, x_1, \ldots, x_n)\,, \tag{5.34}$$

$g = g(x)$, na intervalu $[a, b]$.

Funkce g musí splňovat následující požadavky:

1. Na každém intervalu $[x_j, x_{j+1}]$, $j = 0, \ldots, n-1$, je funkce $g = g(x)$ polynom třetího stupně, t.j.

$$g : [x_j, x_{j+1}] \longrightarrow \mathbb{R}^1\,, \quad g \in \operatorname{span}\{1, x, x^2, x^3\}\,, \tag{5.35}$$

s koeficienty $\alpha_{0,j}$, $\alpha_{1,j}$, $\alpha_{2,j}$, $\alpha_{3,j}$,

$$g(x) = \alpha_{0,j} + \alpha_{1,j}x + \alpha_{2,j}x^2 + \alpha_{3,j}x^3\,,$$

$j = 0, \ldots, n-1$.

2. Podmínky spojitosti:

$$\begin{aligned}\lim_{x\to x_i^-} g(x) &= \lim_{x\to x_i^+} g(x) \stackrel{\text{def}}{=} y_i \\ \lim_{x\to x_i^-} g'(x) &= \lim_{x\to x_i^+} g'(x) \stackrel{\text{def}}{=} y_i' \\ \lim_{x\to x_i^-} g''(x) &= \lim_{x\to x_i^+} g''(x) \stackrel{\text{def}}{=} y_i''\end{aligned}$$

$i = 1, \ldots, n-1$

3. Interpolační podmínky: $y_i = f(x_i)\,,\ i = 0, \ldots, n$, viz (5.33).

Udělejme si bilanci: Na funkci $g = g(x)$ je tedy kladeno $3\,(n-1)$ podmínek spojitosti a $n+1$ interpolačních podmínek. Celkem $4n-2$ požadavků. Na druhé straně, je třeba určit $4n$ koeficientů $\{\alpha_{0,j}, \alpha_{1,j}, \alpha_{2,j}, \alpha_{3,j}\}_{j=0}^{n-1}$. Můžeme tedy přidat dva požadavky. Obvykle chceme, aby

$$\lim_{x\to x_0^+} g''(x) \stackrel{\text{def}}{=} y_0'' = 0\,, \quad \lim_{x\to x_n^-} g''(x) \stackrel{\text{def}}{=} y_n'' = 0\,. \tag{5.36}$$

Operátor $S(f; x_0, x_1, \ldots, x_n)$ definuje funkci $g = g(x)$, která je dvakrát spojitě diferencovatelná, t.j. $g \in C^2[a,b]$, splňuje interpolační podmínku (5.3), a dvě podmínky navíc, např. (5.36). Funkci g říkáme kubický B-spline.

Nechť $g = S(f; x_0, x_1, \ldots, x_n)$. Při konstrukci kubického B-splinu, na každém z intervalů $[x_j, x_{j+1}]$ volíme místo kanonické báze (5.35) speciální bázi kubických polynomů

$$\operatorname{span}\{A_j(x), B_j(x), C_j(x), D_j(x)\}\,, \tag{5.37}$$

kde

$$\begin{aligned}A_j(x) &= \frac{x_{j+1}-x}{x_{j+1}-x_j} \quad \in \mathbb{P}_1\,, \\ B_j(x) &= \frac{x-x_j}{x_{j+1}-x_j} = 1 - A_j(x) \quad \in \mathbb{P}_1\,, \\ C_j(x) &= \tfrac{1}{6}(A_j^3(x) - A_j(x))(x_{j+1}-x_j)^2 \quad \in \mathbb{P}_3\,, \\ D_j(x) &= \tfrac{1}{6}(B_j^3(x) - B_j(x))(x_{j+1}-x_j)^2 \quad \in \mathbb{P}_3\,,\end{aligned}$$

$j = 0, \ldots, n-1$. Poznamenejme, že

$$C_j''(x) = A_j(x)\,, \quad D_j''(x) = B_j(x)\,.$$

V uzlech x_j, x_{j+1}:

$$\begin{array}{llll} A_j(x_j) = 1\,, & A_j(x_{j+1}) = 0\,, & A_j''(x_j) = 0\,, & A_j''(x_{j+1}) = 0\,; \\ B_j(x_j) = 0\,, & B_j(x_{j+1}) = 1\,, & B_j''(x_j) = 0\,, & B_j''(x_{j+1}) = 0\,; \\ C_j(x_j) = 0\,, & C_j(x_{j+1}) = 0\,, & C_j''(x_j) = 1\,, & C_j''(x_{j+1}) = 0\,; \\ D_j(x_j) = 0\,, & D_j(x_{j+1}) = 0\,, & D_j''(x_j) = 0\,, & D_j''(x_{j+1}) = 1\,. \end{array}$$

Volbou bází (5.37) zformulujeme snadno podmínky spojitosti a podmínky spojitosti druhých derivací. Položme

$$y_i \equiv g(x_i)\,, \quad y_i'' \equiv g''(x_i)\,, \quad i = 0, 1, \ldots, n\,.$$

Na každém z intervalů $[x_j, x_{j+1}]$, j=0,. . . ,n-1,

$$\begin{aligned} g(x) &= y_j A_j(x) + y_{j+1} B_j(x) + y_j'' C_j(x) + y_{j+1}'' D_j(x) \\ g'(x) &= \frac{y_{j+1} - y_j}{x_{j+1} - x_j} - \frac{3A_j^2(x) - 1}{6}(x_{j+1} - x_j) y_j'' + \frac{3B_j^2(x) - 1}{6}(x_{j+1} - x_j) y_{j+1}'' \\ g''(x) &= y_j'' C_j''(x) + y_{j+1}'' D_j''(x)\,. \end{aligned}$$

Potom

$$\begin{aligned} \lim_{x \to x_j^-} g(x) &= \lim_{x \to x_j^+} g(x) = y_j \\ \lim_{x \to x_j^-} g''(x) &= \lim_{x \to x_j^+} g''(x) = y_j'' \end{aligned}$$

Zformulujeme podmínky spojitosti prvních derivací: V každém uzlovém bodě x_j, restrikcí na interval $[x_j, x_{j+1}]$ resp. $[x_{j-1}, x_j]$, dostaneme

$$\begin{aligned} \lim_{x \to x_j^+} g'(x) &= \frac{y_{j+1} - y_j}{x_{j+1} - x_j} - \frac{2}{6}(x_{j+1} - x_j)\, y_j'' - \frac{1}{6}(x_{j+1} - x_j)\, y_{j+1}'' \\ \lim_{x \to x_j^-} g'(x) &= \frac{y_j - y_{j-1}}{x_j - x_{j-1}} + \frac{1}{6}(x_j - x_{j-1})\, y_{j-1}'' - \frac{2}{6}(x_j - x_{j-1})\, y_j'' \end{aligned}$$

Musí platit:

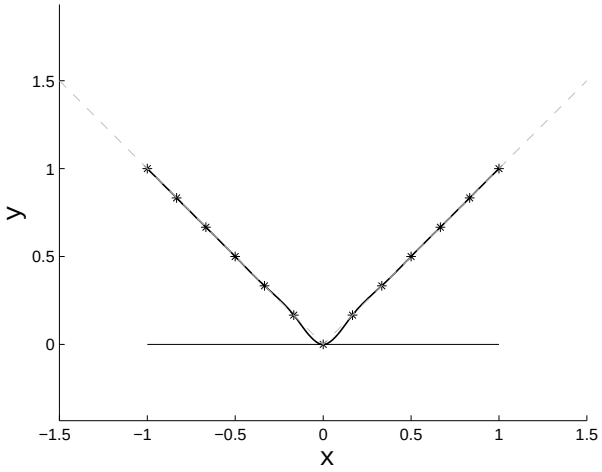
$$\lim_{x \to x_j^+} g'(x) = \lim_{x \to x_j^-} g'(x) \tag{5.38}$$

t.j.

$$\frac{1}{6}(x_j - x_{j-1})\, y_{j-1}'' + \frac{1}{3}(x_{j-1} - x_{j+1})\, y_j'' + \frac{1}{6}(x_{j+1} - x_j)\, y_{j+1}'' = \frac{y_{j+1} - y_j}{x_{j+1} - x_j} - \frac{y_j - y_{j-1}}{x_j - x_{j-1}}\,,$$

$j = 1, \ldots, n-1$. Pro $j = 1$ a $j = n-1$ uplatníme podmínku $y_0'' = y_n'' = 0$, viz (5.36). Podmínky (5.38) lze zformulovat jako soustavu $n-1$ lineárních rovnic pro neznámé $y_1'', \ldots, y_{n-1}''$ s pravou stranou, která závisí na zadaných interpolačních datech $y_0, y_1, \ldots, y_{n-1}, y_n$. Matice soustavy je třídiagonální (hvězdičky indikují nenulové prvky matice soustavy):

$$\begin{bmatrix} * & * & & & & \\ * & * & * & & & \\ & * & * & * & & \\ & & \ddots & \ddots & \ddots & \\ & & & * & * & * \\ & & & & * & * \end{bmatrix} \cdot \begin{bmatrix} y_1'' \\ \vdots \\ \\ \vdots \\ y_{n-1}'' \end{bmatrix} = \begin{bmatrix} \\ \vdots \\ \\ \vdots \\ \\ \end{bmatrix}, \tag{5.39}$$



Obrázek 5.7: Kubický B-spline, interpolace funkce $y = \mathrm{abs}(x)$ na intervalu $[-1, 1]$, $n + 1 = 13$ ekvidistantních interpolačních uzlů.

j-tý řádek soustavy je uveden za formulí (5.38).

Jak vyčíslit $g = g(x)\,, \quad g = S(f; x_0, \ldots, x_n)$?

Algoritmus 5.1. (Kubický B-spline) *Nechť $x \in [a, b]$.*

1. *Nezávisle na zadaném x, řešíme soustavu* (5.39) *pro neznámé $y_1'', \ldots, y_{n-1}''$, $y_0'' = y_n'' = 0$, a zadaná interpolační data*

$$\begin{array}{llll} x_0\,, & x_1\,, & \ldots\,, & x_n\,, \\ f(x_0) = y_0\,, & f(x_1) = y_1\,, & \ldots\,, & f(x_n) = y_n\,. \end{array}$$

   Získáme tabulku

$$\begin{array}{llll} x_0\,, & x_1\,, & \ldots\,, & x_n\,, \\ y_0\,, & y_1\,, & \ldots\,, & y_n\,, \\ y_0''\,, & y_1''\,, & \ldots\,, & y_n''\,. \end{array} \tag{5.40}$$

2. *Pomocí algoritmů třídění (sorting) najdeme index j, pro který $x_j \le x \le x_{j+1}$. Potom*

$$g(x) = y_j A_j(x) + y_{j+1} B_j(x) + y_j'' C_j(x) + y_{j+1}'' D_j(x)\,.$$

Interpolovanou funkci $g = g(x)$ vyčíslujeme zpravidla pro celou řadu hodnot $a \le x \le b$. Aktualizujeme jenom druhý krok algoritmu. Na Obrázku 5.7 byla funkce $g = g(x)$ vyčíslena ve čtyřiceti bodech intervalu $[-1, 1]$.

Poznámka 5.2. (Srovnání interpolačních technik) *Experiment se stejnými daty nabízí srovnání Lagrangeovy interpolace (Obrázky 5.2 a 5.3) a interpolace typu spline*

(Obrázek 5.7). Lagrangeova interpolace bez optimalizace pozice interpolačních uzlů nemá šanci. Jak Lagrangeova interpolace, tak interpolace typu spline vyžadují vyřešit soustavu lineárních rovnic (5.6) *resp.* (5.39). *Pokud n je velké a pokud interpolační uzly se "zahušťují", potom při řešení soustavy* (5.6) *(s Vandermondovou maticí) lze očekávat problémy. Na druhou stranu, matice soustavy* (5.39) *je třídiagonální a má příznivou vlastnost: Je diagonálně dominantní. To například znamená, že při řešení soustavy nemusíme používat pivotaci.*

Spliny mají netušené aplikace např. v počítačové grafice a při vytváření virtuálních objektů. A co Čebyševovy polynomy? Když už nic jiného, je to krásná matematika.

Kapitola 6

Numerická integrace soustav obyčejných diferenciálních rovnic

6.1 Dva motivační příklady

Příklad 6.1. (Logistická rovnice) *Diferenciální rovnice*

$$x' = (a - bx)x - c \tag{6.1}$$

s počáteční podmínkou

$$x(t_0) = x_0 \tag{6.2}$$

a parametry $a \geq 0$, $b \geq 0$, $c \geq 0$, může modelovat časový vývoj populace nějakého hmyzího druhu (populární je např. banánová muška, lat. Drosophila).

Řešením úlohy (6.1) & (6.2) se rozumí spojitě diferencovatelná funkce $t \mapsto u(t)$ taková, že

$$\frac{du(t)}{dt} = (a - bu(t))\, u(t) - c$$

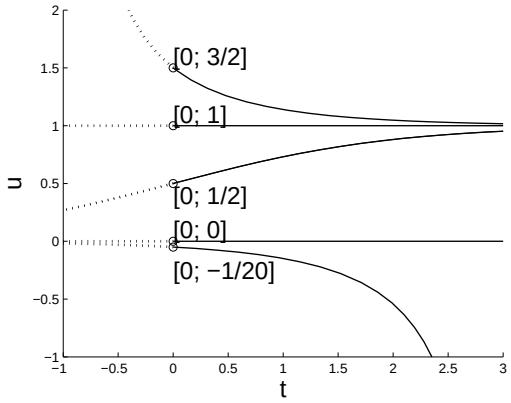
pro všechna $t \in \mathbb{R}^1$, přičemž $u(t_0) = x_0$.

Definujme operátor $\phi : \mathbb{R}^1 \times \mathbb{R}^1 \to \mathbb{R}^1$, který zadané počáteční podmínce (6.2) přiřazuje řešení $u(t)$ úlohy (6.1) & (6.2) v čase t. Tedy,

$$(t_0, x_0) \in \mathbb{R}^1 \times \mathbb{R}^1 \longmapsto u(t) \equiv \phi(t, t_0, x_0) \in \mathbb{R}^1\,. \tag{6.3}$$

Operátor ϕ definuje evoluci počáteční podmínky v čase.

Hodnotu $u(t) = \phi(t, t_0, x_0) \in \mathbb{R}^1$ lze interpretovat jako stav v čase $t \in \mathbb{R}^1$. Z hlediska uvažovaného modelu se stavem rozumí hustota populace v jednotkovém objemu. Model předpovídá budoucnost ($t \geq t_0$) a rekonstruuje minulost ($t \leq t_0$). Na Obrázku 6.1 jsou *trajektorie* vycházející z počátečních podmínek $(t_0, x_0) := (0, 3/2)$, $(0, 1)$, $(0, 1/2)$, $(0, 0)$ a $(0, -1/20)$ pro hodnoty $a = 1$, $b = 1$ a $c = 0$ parametrů úlohy.



Obrázek 6.1: Populační model: Logistická rovnice. Pět různých trajektorií, které odpovídají indikovaným počátečním podmínkám. Parametry $a = 1$, $b = 1$, $c = 0$.

Plná resp. čárkovaná křivka odpovídá stavům pro které $t \geq 0$ resp. $t \leq 0$. Obrázek samozřejmě znázorňuje jenom *části* trajektorií, které odpovídají zvolenému měřítku. Trajektorií se obecně rozumí zobrazení $t \in I \longmapsto (t, \phi(t, t_0, x_0))$, kde I je interval obsahující t_0. Času $t \in I$ je tedy přiřazena uspořádaná dvojice, kterou tvoří čas t a odpovídající stav $u(t) = \phi(t, t_0, x_0)$. Je možné si položit otázku, jaký je *maximální interval existence* řešení počáteční úlohy (6.1) & (6.2). Jinými slovy, jaký je největší možný interval I?

Trajektorie na Obrázku 6.1 byly získány *numericky*, tedy přibližně. Vzhledem k jednoduchosti modelu (6.1) & (6.2) je možné zkonstruovat operátor ϕ *explicitně* pomocí elementárních funkcí. Technice řešení počáteční úlohy (t.j. konstrukci operátoru ϕ) se říká integrace diferenciální rovnice, v našem případě rovnice (6.1). Numerickému (t.j. přibližnému) řešení se říká *numerická integrace (kvadratura)* diferenciální rovnice.

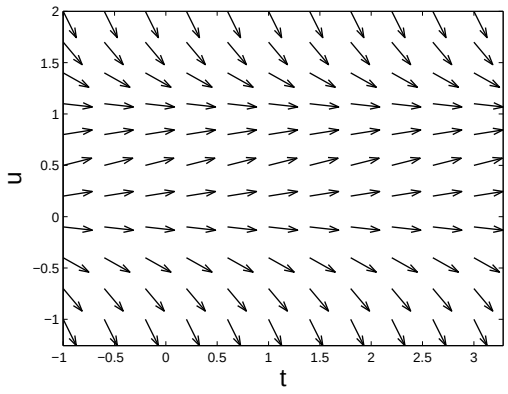
Pravou stranu obyčejné diferenciální rovnice (6.1) je možno chápat jako zobrazení $f : \mathbb{R}^1 \times \mathbb{R}^1 \to \mathbb{R}^1$,

$$(t, x) \longmapsto f(t, x) \equiv (a - bx)x - c\,. \tag{6.4}$$

Vidíme, že pravá strana nezávisí na t. (V následujícím Příkladu to nebude pravda.) Na základě f definujme zobrazení

$$\begin{bmatrix} t \\ x \end{bmatrix} \in \mathbb{R}^1 \times \mathbb{R}^1 \;\longmapsto\; \begin{bmatrix} 1 \\ f(t,x) \end{bmatrix} \in \mathbb{R}^1 \times \mathbb{R}^1\,. \tag{6.5}$$

Grafu zobrazení (6.5) se říká směrové pole. Směrové pole popisuje přemístěné zadaného vektoru (t, x) do pozice $\overline{(t, x) + (1, f(t, x))}$. Interpretujeme to tak, že výsledná pozice je superpozicí původního *vázaného vektoru* (t, x) a *směrového vektoru* $(1, f(t, x))$. Zobrazení (6.5) definuje právě směrový vektor tečny k příslušné trajektorii v bodě (t, x).



Obrázek 6.2: Směrové pole logistické rovnice. Parametry $a = 1$, $b = 1$, $c = 0$.

Směrový vektor má normalizovánu první složku. Můžeme zvolit jinou normalizaci: Nechť $K > 0$ je zvolená konstanta. Definujme směrové pole jako graf zobrazení

$$\begin{bmatrix} t \\ x \end{bmatrix} \in \mathbb{R}^1 \times \mathbb{R}^1 \longmapsto \frac{K}{\sqrt{(}1 + (f(t,x))^2)} \begin{bmatrix} 1 \\ f(t,x) \end{bmatrix} \in \mathbb{R}^1 \times \mathbb{R}^1 . \qquad (6.6)$$

Obě definice (6.5) a (6.6) jsou ekvivalentní. Poskytují stejnou informaci.

Na Obrázku 6.2 je směrové pole (přesněji, část směrového pole v měřítku, které odpovídá Obrázku 6.2). Je použita definice (6.2). Směr přesunu z pozice (t, x) do pozice $(t, x) + \frac{K}{\sqrt{(1+(f(t,x))^2)}} (1, f(t, x))$ je vyznačen šipkou. Normalizace, t.j. volba K v (6.2), se provádí automaticky, a v podstatě nás nezajímá. Důležité je, že všechny šipky mají stejnou délku. Směrové pole samozřejmě se počítá a zobrazuje v bodech zvolené mřížky (zde konkrétně 10 krát 10).

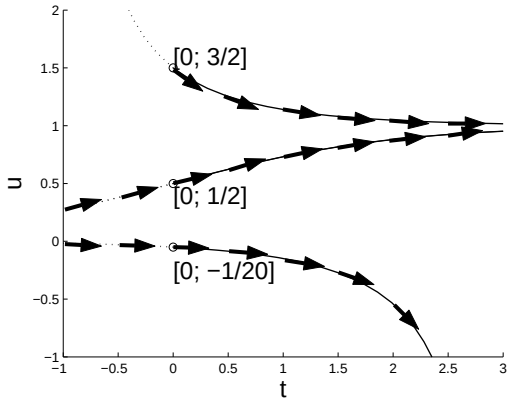
Podél trajektorií, které odpovídají počátečním podmínkám $(0, 1)$ a $(0, 0)$ se řešení s časem nevyvíjí, viz Obrázek 6.1. Tyto trajektorie jsou interpretovány jako *stacionární řešení* úlohy (6.1) &(6.2).

Pomocí směrového pole lze zformulovat alternativní geometrickou představu o řešení problému (6.1) &(6.2): Hledáme křivku (trajektorii) v $\mathbb{R}^2$ tak, aby

a) procházela počáteční podmínkou

b) v každém bodě směrnice tečny k trajektorii souhlasila se směrem vektorového pole.

Říkali jsme, že trajektorie na Obrázku 6.1 (a Obrázku 6.3) byly získány aplikací numerické metody. Konkrétně, tyto trajektorie byly aproximovány v systému `MATLAB` použitím funkce `ODE23`, při standartním nastavení všech parametrů řešiče.



Obrázek 6.3: Směrnice tečen k trajektorii. Parametry $a = 1$, $b = 1$, $c = 0$.

Poznamenejme, že numerické metody probírané v tomto učebním textu (včetně zmíněné `ODE23`) definují posloupnosti diskrétních časů a stavů. Tuto posloupnost kreslící program lineárně interpoluje a vzniká dojem spojité trajektorie.

Na Obrázku 6.4 je, v jistém smyslu, patologický případ: Plná křivka reprezentuje trajektorii s počáteční podmínkou $t_0 = 0$, $x_0 = 0.7233$ při volbě $a = 1$, $b = 1$, $c = 1/5$ parametrů úlohy. Bylo použito explicitní formule

$$u(t) = \sqrt{5}/10 \, \tanh\left((t - t_0)\,\sqrt{5}/10 + \operatorname{arctanh}\left((2x_0 - 1)\sqrt{5}\right)\right) + 1/2$$

Trajektorie byla aproximována použitím funkce `ODE23` (výstup viz symboly $\square$) a alternativní funkce `ODE15s` (výstup viz symboly $\diamond$). Čárkované křivky vznikly jako lineární interpolace výstupů funkcí `ODE23` a `ODE15s`. Je vidět, že dříve úspěšná `ODE23` hrubě zkreslila realitu, zatímco trajektorie počítaná pomocí `ODE15s` kvalitativně odpovídá. Z toho plyne poučení, že záleží na použité numerické metodě. V čem je vyšetřovaný případ patologický? Počáteční podmínka $x_0 = 0.7233$ je velmi blízko jednomu ze stacionárních řešení. Je to problém numerické povahy, a jak je vidět, existuje lék.

Příklad 6.2. (Lineární oscilátor) *Uvažujme soustavu dvou obyčejných rovnic*

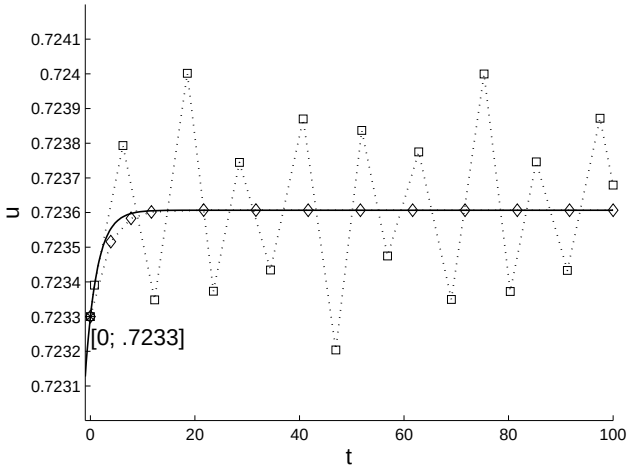
$$\begin{aligned} x_1' &= x_2 \\ x_2' &= -bx_1 + c\cos(\omega t)\,. \end{aligned}$$

kde $b \geq 0$, $c \geq 0$ a $\omega \in \mathbb{R}^1$ jsou parametry. Ve vektorové notaci,

$$x' = f(t, x) \equiv \begin{bmatrix} x_2 \\ -bx_1 + c\cos(\omega t) \end{bmatrix}. \tag{6.7}$$

Soustavu (6.7) *doplníme počáteční podmínkou*

$$x(t_0) = x_0 \in \mathbb{R}^2\,. \tag{6.8}$$



Obrázek 6.4: Trajektorie s počáteční podmínkou $[0; 0.7233]$. Numerické metody `ODE23` $\square$, `ODE15s` $\diamond$. Parametry $a = 1$, $b = 1$, $c = 1/5$.

Řešením úlohy (6.7) & (6.8) se rozumí spojitě diferencovatelná vektorová funkce $t \mapsto u(t) \in \mathbb{R}^2$ taková, že

$$\frac{du(t)}{dt} = f(t, u(t)) \equiv \begin{bmatrix} u_2(t) \\ -bu_1(t) + c\cos(\omega t) \end{bmatrix} \tag{6.9}$$

pro všechna t, přičemž $u(t_0) = x_0 \in \mathbb{R}^2$.

Zadáme-li počáteční podmínku $(t_0, x_0) \in \mathbb{R}^1 \times \mathbb{R}^2$, potom v každém čase $t \in \mathbb{R}^1$ existuje jednoznačně vektor řešení $u(t) \in \mathbb{R}^2$ úlohy (6.7) & (6.8). Tedy, existuje operátor

$$(t_0, x_0) \in \mathbb{R}^1 \times \mathbb{R}^2 \longmapsto u(t) \equiv \phi(t, t_0, x_0) \in \mathbb{R}^2\,. \tag{6.10}$$

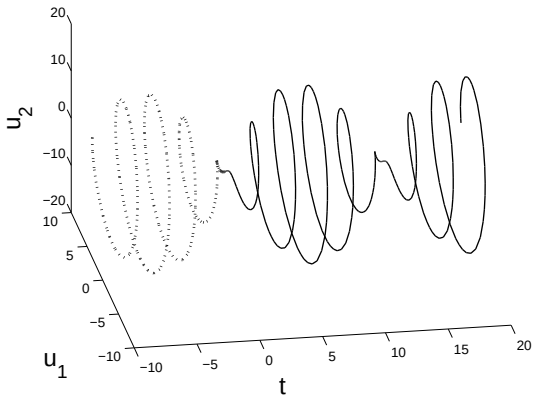
Tento operátor lze i v tomto případě zkonstruovat explicitně. Nicméně nám stačí výrok o existenci takového operátoru.

Úloha (6.7) & (6.8) modeluje oscilace pružiny. Složky $u_1(t)$ resp. $u_2(t)$ vektoru $u(t) = \phi(t, t_0, x_0) \in \mathbb{R}^2$ jsou interpretovány jako výchylka resp. zrychlení v čase t.

Proměnné $x = (x_1, x_2) \in \mathbb{R}^2$ v rovnici (6.7) se říká stavová proměnná; x_1 resp. x_2 představují okamžitou výchylku resp. okamžimžité zrychlení. Prostor $\mathbb{R}^2$ se, v souvislosti s úlohou (6.7) & (6.8), nazývá stavový prostor. Parametry b, c a ω mají tuto interpretaci: b je modul pružnosti, c resp. ω jsou amplituda resp. frekvence budící síly.

Problém lineárního osciláloru se obvykle formuluje jako řešení lineární diferenciální rovnice druhého řádu

$$x'' + b\,x = c\,\cos(\omega t) \tag{6.11}$$



Obrázek 6.5: Lineární oscilátor. Trajektorie s počáteční podmínkou $t_0 = 0$, $x_0 = [1; 0]$. Parametry $b = 9$, $c = 10$, $\omega = 2.5$.

s počáteční podmínkou $(x(t_0), x'(t_0)) = x_0 \in \mathbb{R}^2$. Uvedená počáteční úloha pro rovnici (6.11) je zřejmě ekvivalentní úloze (6.7) & (6.8). Tuto poznámku uvádíme jako příklad, že soustavy rovnic vyšších řádů lze převést na soustavy rovnic prvního řádu.

Trajektorie, t.j. zobrazení

$$t \in \mathbb{R}^1 \longmapsto [t, \phi(t, t_0, x_0)] \in \mathbb{R}^1 \times \mathbb{R}^2$$

na Obrázku 6.5 byla počítána numericky (funkce `ODE23`). Perioda T pohybu byla odhadnuta jako $T = 12.5664$ s přesností na pět platných cifer.

Projekci trajektorie do stavového prostoru, t.j. zobrazení

$$t \in \mathbb{R}^1 \longmapsto \phi(t, t_0, x_0) \in \mathbb{R}^2 \,,$$

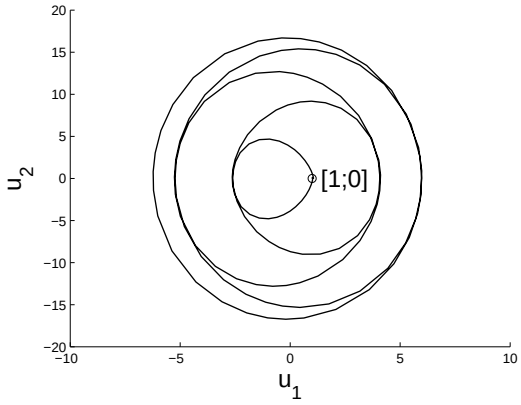
se říká fázová křivka. Příklad je na Obrázku 6.6.

6.2 Matematický model evoluce

Nejprve zformulujeme problém: *Počáteční úlohu* pro *soustavu obyčejných diferenciálních rovnic.*

Tohoto problému se týkaly dva motivační příklady z předchozího odstavce. Viděli jsme, že oba příklady modelují evoluci v konečně dimenzionálním prostoru (v $\mathbb{R}^1$ resp. $\mathbb{R}^2$): Stavová proměnná u (v $\mathbb{R}^1$ resp. $\mathbb{R}^2$) se vyvíjí v čase t. Časový vývoj začíná v zadaném čase t_0, pro zadanou počáteční podmínku u_0 (v $\mathbb{R}^1$ resp. $\mathbb{R}^2$).

Stavový prostor budeme identifikovat s lineárním prostorem $\mathbb{R}^n$. Dimenze stavového prostoru je tedy n. Stav systému je ztotožněn s prvkem stavového prostoru, tedy s aktuální hodnotou *stavové proměnné*. Čas je skalární parametr. Budeme jej standartně označovat jako t.



Obrázek 6.6: Lineární oscilátor. Projekci trajektorie do stavového prostoru $\mathbb{R}^2$ se říká fázová křivka. Parametry $b = 9$, $c = 10$, $\omega = 2.5$.

Data uvažované úlohy jsou

1. počáteční podmínka, t.j. zadaný stav $x_0 \in \mathbb{R}^n$ v čase t_0.
2. pravá strana soustavy, zadaná zobrazením

$$f : \mathbb{R}^1 \times \mathbb{R}^n \to \mathbb{R}^n \,. \tag{6.12}$$

Budeme předpokládat, že definiční obor pravé strany je otevřená množina $J \times D \subset \mathbb{R}^1 \times \mathbb{R}^n$ a že J je interval. Předpokládáme, že $t_0 \in J$ a $x_0 \in D$, tedy počáteční podmínka $(t_0, x_0) \in \mathbb{R}^1 \times \mathbb{R}^n$ patří do definičního oboru.

Počáteční úlohu zapisujeme, ve vektorové notaci, jako

$$x' = f(t, x)\,, \quad x(t_0) = x_0\,. \tag{6.13}$$

Jedná se o soustavu n obyčejných diferenciálních rovnic (angl. ordinary differential equations, ODEs)

$$\begin{aligned} x_1' &= f_1(t, x_1, \ldots, x_n) \\ &\vdots \\ x_n' &= f_n(t, x_1, \ldots, x_n) \end{aligned}$$

s počáteční podmínkou $x(t_0) = x_0 \in \mathbb{R}^n$.

Definujme, co znamená řešení problému (6.13). Za tím účelem budeme požadovat, aby pravá strana byla spojitá, t.j.

$$f \in C(J \times D, \mathbb{R}^n)\,. \tag{6.14}$$

Definice 6.1. (Řešení počáteční úlohy) *Předpokládejme, že $f \in C(J \times D, \mathbb{R}^n)$. Nechť existuje*

1. *$I \subset J$, interval obsahující t_0*

2. *vektorová funkce $t \mapsto u(t) \in \mathbb{R}^n$ která má spojitou první derivaci na I, t.j. $u \in C^1(I, \mathbb{R}^n)$.*

Nechť

$$u'(t) = f(t, u(t)) \tag{6.15}$$

pro všechna $t \in I$ a je splněna počáteční podmínka

$$u(t_0) = x_0\,. \tag{6.16}$$

Potom říkáme, že funkce u je řešením (6.13) *na intervalu I.*

Poznámka 6.1. (Integrální formulace) *Jestliže $f \in C(J \times D, \mathbb{R}^n)$, potom funkce u splňuje* (6.15)&(6.16) *právě když*

$$u(t) = x_0 + \int_{t_0}^{t} f(s, u(s))ds \tag{6.17}$$

pro každé $t \in I$.

Pravou stranu f soustavy v úloze (6.13) lze interpretovat jako směrové pole:

Definice 6.2. (Směrové pole) *Nechť $J \times D$ je definiční obor pravé strany f rovnice* (6.12). *Směrové pole je graf zobrazení*

$$\begin{bmatrix} t \\ x \end{bmatrix} \in \mathbb{R}^1 \times \mathbb{R}^n \;\longmapsto\; \begin{bmatrix} 1 \\ f(t,x) \end{bmatrix} \in \mathbb{R}^1 \times \mathbb{R}^n\,. \tag{6.18}$$

Směrové pole popisuje přemístění zadaného vektoru (t, x) do pozice $(t, x)+(1, f(t, x))$. Nová pozice je superpozicí původního *vázaného vektoru* (t, x) a *směrového vektoru* $(1, f(t, x))$. Směrový vektor má normalizovánu první složku.

Můžeme si myslet, že v každém bodě $(t, x) \in J \times D$ je k dispozici informace jakým směrem se dát. V numerických metodách dostáváme tuto informaci jenom čas od času (v diskrétních časech).

Potřebujeme tvrzení, které zaručí existenci a jednoznačnost řešení počáteční úlohy (6.15). Pouhá spojitost pravé strany f, t.j. předpoklad (6.14), nestačí:

Definice 6.3. *Nechť $f : J \times D \to \mathbb{R}^n$, $f \in C(J \times D, \mathbb{R}^n)$. Říkáme, že f je Lipschitzovsky spojitá na $J \times D$, jestliže existuje konstanta $L \geq 0$ tak, že*

$$\|f(t, x) - f(t, y)\| \leq L\, \|x - y\| \tag{6.19}$$

pro každé $t \in J$ a každé $x, y \in D$.

Věta 6.1. (Lokální existence a jednoznačnost) *Nechť f je Lipschitzovsky spojitá na $J \times D$. Potom počáteční úloha* (6.13) *je lokálně jednoznačně řešitelná t.j. pro každou počáteční podmínku $(t_0, x_0) \in J \times D$ platí:*

Existuje otevřený interval $I \subset J$ obsahující $t_0 \in I$, a funkce $u \in C^1(I, \mathbb{R}^n)$ tak, že vektorová funkce $t \mapsto u(t)$ je právě jediné řešení rovnice (6.15) *na intervalu $t \in I$, které splňuje počáteční podmínku* (6.16).

Důkaz Věty 6.1 lze najít např. v [7], Věta 11.1.1. na str.202 (lokální existence) a Věta 11.1.5. na str.211 (jednoznačnost). Důkaz první z citovaných vět je založen na konstrukci aproximativních přiblížení přesného řešení. Tedy na numerické metodě. Půjde o Eulerovu metodu, přesněji její geometrickou interpretaci: tzv. *Eulerův graf.* Eulerovu metodu budeme definovat v úvodu následujícího odstavce. □

Věta 6.1 tvrdí, že existuje operátor φ, který každé počáteční podmínce $(t_0, x_0) \in J \times D$ přiřazuje řešení

$$u(t) = \varphi(t, t_0, x_0) \in \mathbb{R}^n\,, \quad t \in I \tag{6.20}$$

počáteční úlohy (6.15) & (6.16) na otevřeném intervalu I, $I \subset J$. Interval I závisí na volbě počáteční podmínky $(t_0, x_0) \in J \times D$. Intervalu I říkáme interval existence.

Poznámka 6.2. *Je možné, za trochu silnějších předpokladů (lokálně Lipschitzovsky spojitá f), najít maximální interval existence, který obsahuje I. Viz [7], Věta 12.1.1., str. 218.*

Operátoru φ, viz (6.20), říkáme tok vektorového pole. Vektorové pole je synonymum pro směrové pole, a to je synonymum pro pravou stranu f diferenciální rovnice. Operátor toku vektorového pole modeluje evoluci v $\mathbb{R}^n$.

Definice 6.4. (Trajektorie, Fázová křivka) *Uvažujme počáteční úlohu* (6.13). *Nechť φ je příslušný tok vektorvého pole,* (6.20). *Nechť I je interval existence. Křivce*

$$t \in I \longmapsto (t, \phi(t, t_0, x_0)) \in \mathbb{R}^1 \times \mathbb{R}^n$$

říkáme trajektorie. Fázová křivka je projekce trajektorie do stavového prostoru:

$$t \in I \longmapsto \phi(t, t_0, x_0) \in \mathbb{R}^n\,.$$

Příklady jsou na Obrázcích 6.1, 6.5 a 6.6.

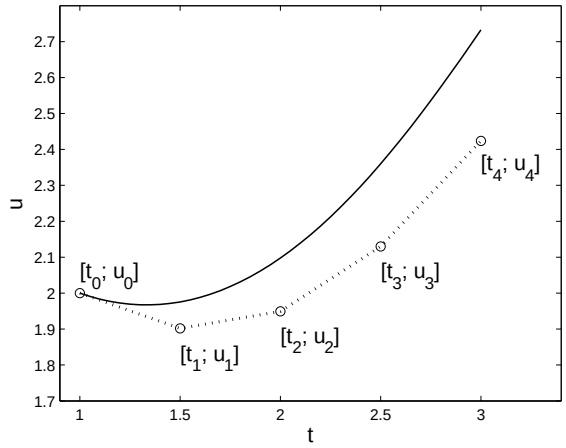
Víme, že za předpokladů Věty 6.1, operátor φ existuje. Lze φ zkonstruovat? Jenom ve vyjímečných případech:

Příklad 6.3. *Uvažujme jednodimenziální počáteční úlohu*

$$x' = f(t, x) \equiv ax\,, \quad x(t_0) = x_0\,,$$

kde $a \in \mathbb{R}^1$ je parametr. Ukážeme, že

$$u(t) \;=\; \varphi(t, t_0, x_0) \;\equiv\; \mathrm{e}^{a(t-t_0)} x_0\,.$$



Obrázek 6.7: Eulerova metoda s krokem $\tau = 0.5$. Eulerův graf.

Postup známe z analýzy: Řešíme rovnici $x' = ax$. Separací proměnných, a integrací,

$$\frac{\mathrm{d}x}{\mathrm{d}t} = at\,, \quad \int \frac{\mathrm{d}x}{x} = a \int \mathrm{d}t$$

dostaneme, že $\ln x = at + C$, kde C je integrační konstanta. Konstantu C určíme z počáteční podmínky a odvodíme formuli pro *obecné řešení* $x = \mathrm{e}^{a(t-t_0)}x_0$.

Říkáme, že rovnici $x' = ax$ integrujeme. Konstrukce operátoru φ, založená na integraci rovnice (6.15), může být komplikovaná nebo není prostě známa.

Budeme rozvíjet postupy, které tok vektorového pole pouze aproximují, ale s chybou, kterou lze předem odhadnout. Tyto techniky lze chápat jako numerickou integraci soustav obyčejných diferenciálních rovnic. V následujícím odstavci probereme tzv. jednokrokové metody, které tvoří důležitou třídu metod numerické integrace ODEs.

6.3 Jednokrokové metody

Naším cílem je numerické řešení počáteční úlohy (6.13): Hledáme řešení $u(t) = \phi(t, t_0, x_0)$ na zadaném *konečném* uzavřeném intervalu $t \in [t_0, T]$. Dále budeme předpokládat, že pravá strana f je dostatečně hladká.

Prototypem metod vyšetřovaných v této kapitole je *Eulerova metoda* (1768): Uvažujme dělení

$$\{t_j\}_{j=0}^{N}\,, \quad t_{j+1} > t_j\,, \quad t_N = T\,, \tag{6.21}$$

intervalu $[t_0, T]$. Čas t_0 je určen počáteční podmínkou (6.13); N je počet dělicích bodů.

Definujme posloupnost $\{u_j\}_{j=0}^N$ pomocí rekurence

$$u_{j+1} = u_j + (t_{j+1} - t_j) f(t_j, u_j)\,. \tag{6.22}$$

Přitom j-tý prvek posloupnosti, $u_j \in \mathbb{R}^n$, je interpretován jako aproximace stavu $u(t_j) \in \mathbb{R}^n$ v čase t_j.

Pro ilustraci, řešme rovnici $x' = 0.3\,x\,\sin(t - 4/3)$ s počáteční podmínkou $x(1) = 2$. Řešení $u(t)$ hledáme na intervalu $[1, 3]$, viz Obrázek 6.7. Uvažujme ekvidistantní (rovnoměrné) dělení intervalu. Tedy, $t_{j+1} - t_j \equiv \tau$, $j = 1, \ldots, N$. Položme $\tau = 0.5$. Potom $N = 4$. Numerické řešení představují dvojice $[t_j, u_j] \in \mathbb{R}^1 \times \mathbb{R}^n$ pro $j = 0, \ldots, N$, které jsou generovány rekurencí (6.22). Pokud potřebujeme aproximovat řešení v časech $t_j < t < t_{j+1}$, použijeme lineární interpolaci:

$$\begin{bmatrix} t \\ u(t) \end{bmatrix} \approx \frac{t_{j+1} - t}{t_{j+1} - t_j} \begin{bmatrix} t_j \\ u_j \end{bmatrix} + \frac{t - t_j}{t_{j+1} - t_j} \begin{bmatrix} t_{j+1} \\ u_{j+1} \end{bmatrix}\,. \tag{6.23}$$

Tato interpolace je na Obrázku 6.7 znázorněna tečkovaně. Příslušné křivce se říká *Eulerův graf*. Pokud nejsme s aproximací řešení spokojeni, pak je třeba zmenšit časový krok τ nebo zvolit přesnější metodu. A právě o tom bude tato kapitola.

Ale vraťme se ještě k Eulerově metodě. Iterace (6.22) ve stavovém prostoru $\mathbb{R}^n$ lze ekvivalentně zapsat také v "časoprostoru" $\mathbb{R}^1 \times \mathbb{R}^n$:

$$\begin{bmatrix} t_j \\ u_j \end{bmatrix} \longmapsto \begin{bmatrix} t_j \\ u_j \end{bmatrix} + (t_{j+1} - t_j) \begin{bmatrix} 1 \\ f(t_j, u_j) \end{bmatrix} \equiv \begin{bmatrix} t_{j+1} \\ u_{j+1}\,. \end{bmatrix} \tag{6.24}$$

Tyto iterace mají názornou geometrickou interpretaci. Připomeňme pojem směrové pole, viz Definice 6.2. Uvažujme přímku v $\mathbb{R}^1 \times \mathbb{R}^n$, která vychází z $\begin{bmatrix} t_j \\ u_j \end{bmatrix}$ ve směru $\begin{bmatrix} 1 \\ f(t_j, u_j) \end{bmatrix}$, viz (6.18). Přímka má parametrické vyjádření

$$\tau \in \mathbb{R}^1 \longmapsto \begin{bmatrix} t_j \\ u_j \end{bmatrix} + \tau \begin{bmatrix} 1 \\ f(t_j, u_j) \end{bmatrix} \in \mathbb{R}^1 \times \mathbb{R}^n\,.$$

Potom $j+1$-tý krok procesu (6.24) odpovídá volbě $\tau \equiv t_{j+1} - t_j$.

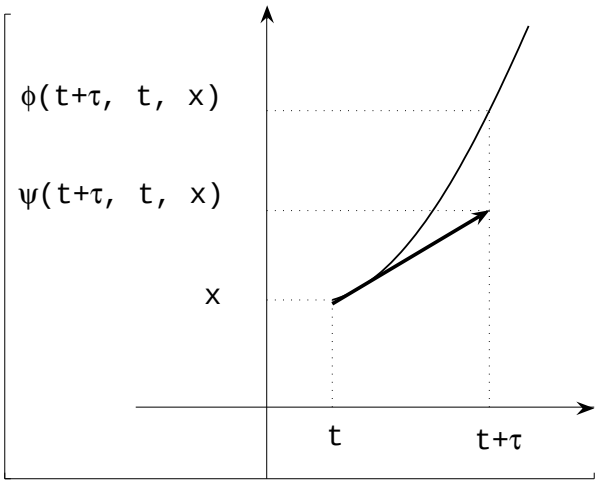
Budeme definovat jeden krok obecné metody. Srovnáme jí s tokem vektorového pole. Připomeňme tok vektorového pole φ, viz (6.20), $u(t) = \varphi(t, t_0, x_0)$, $t \in I$. Dvojici $(t_0, x_0) \in J \times D$ označme $(t, x) \in J \times D$. Existuje otevřený interval I tak, že

$$u(t + \tau) = \varphi(t + \tau, t, x)\,, \quad \tau \in I\,. \tag{6.25}$$

To je tvrzení Věty 6.1, formulované pomocí toku vektorového pole. Poznamenejme, že $\phi(t, t, x) = x$. Pro každou dvojici $(t, x) \in J \times D$ definujme $x' \in \mathbb{R}^n$:

$$x' = \lim_{\tau \to 0} \frac{1}{\tau} \left(\phi(t + \tau, t, x) - x\right) = f(t, x)\,. \tag{6.26}$$

Říkáme, že x' je okamžitá rychlost v bodě (t, x). Formální zápis (6.13) má teď přesný smysl.



Obrázek 6.8: Jednokroková metoda. Diskrétní tok $\psi(t+\tau, t, x)$ vs. přesné řešení $\varphi(t+\tau, t, x)$.

Definice 6.5. (Jednokroková metoda) *Nechť f je Lipschitzovsky spojitá na $J \times D$. Nechť zobrazení $\psi : J \times D \times \mathbb{R}^1 \to \mathbb{R}^n$,*

$$t \in J\,,\ x \in D\,,\ \tau \geq 0 \ \longmapsto\ \psi(t+\tau, t, x) \in \mathbb{R}^n\,, \tag{6.27}$$

v každém bodě $(t, x) \in J \times D$ splňuje podmínku konsistence:

$$\psi(t, t, x) = \lim_{\tau \to 0} \frac{\psi(t+\tau, t, x) - x}{\tau} = f(t, x)\,. \tag{6.28}$$

Potom operátoru ψ říkejme diskrétní tok vektorového pole f a parametru τ říkejme časový krok. Diskrétní tok definuje jednokrokovou metodu.

Na Obrázku 6.8 je idea jednokrokové metody. Definice 6.5 dává návod

$$\begin{bmatrix} t \\ x \end{bmatrix} \ \mapsto\ \begin{bmatrix} t+\tau \\ \psi(t+\tau, t, x) \end{bmatrix}\,, \tag{6.29}$$

jak přejít od aproximace řešení v čase t k aproximaci řešení v čase $t+\tau$. Rekurencí (6.29) vytvoříme posloupnost aproximací řešení v diskrétních časech $\{t_j\}_{j=0}^{N}$, $t_{j+1} > t_j$.

Budeme specifikovat tři jednokrokové metody:

Definice 6.6. (Eulerova metoda, 1768) *Nechť f je Lipschitzovsky spojitá na $J \times D$. Nechť $(t, x) \in J \times D$, $\tau \geq 0$. Položme $\kappa_1 = f(t, x)$. Definujme*

$$\psi(t+\tau, t, x) \equiv x + \tau\kappa_1\,. \tag{6.30}$$

To je jenom jiný zápis metody (6.22).

Definice 6.7. (Rungeho metoda, 1895) *Nechť f je Lipschitzovsky spojitá na $J \times D$. Nechť $(t,x) \in J \times D$, $\tau \geq 0$. Položme $\kappa_1 = f(t,x)$, $\kappa_2 = f(t+\frac{\tau}{2}, x+\frac{\tau}{2}\kappa_1)$. Definujme*

$$\psi(t+\tau, t, x) \equiv x + \tau\kappa_2 \,. \tag{6.31}$$

Definice 6.8. (Rungeho-Kuttova metoda, 1901) *Nechť f je Lipschitzovsky spojitá na $J \times D$. Nechť $(t,x) \in J \times D$, $\tau \geq 0$. Položme $\kappa_1 = f(t,x)\,,\ \kappa_2 = f(t+\frac{\tau}{2}, x+\frac{\tau}{2}\kappa_1)\,,\ \kappa_3 = f(t+\frac{\tau}{2}, x+\frac{\tau}{2}\kappa_2)\,,\ \kappa_4 = f(t+\tau, x+\tau\,\kappa_3)\,.$ Definujme*

$$\psi(t+\tau, t, x) \equiv x + \tau\left(\frac{1}{6}\kappa_1 + \frac{1}{3}\kappa_2 + \frac{1}{3}\kappa_3 + \frac{1}{6}\kappa_4\right) . \tag{6.32}$$

Poznámka 6.3. *Srovnejme: Jeden krok Eulerovy metody nás stojí jedno vyčíslení pravé strany. Jeden krok Rungeho metody nás stojí dvě vyčíslení pravé strany. Jeden krok Runge-Kuttovy metody nás stojí čtyři vyčíslení pravé strany. Co získáme aplikací "dražší" metody?*

Definice 6.9. (Lokální diskretizační chyba, řád metody) *Předpokládejme, že f je Lipschitzovsky spojitá na $J \times D$. Nechť φ je příslušný operátor toku vektorového pole f. Zvolme $(t,x) \in J \times D$ a časový krok $\tau > 0$. Uvažujme jednokrokovou metodu: Nechť $\psi(t+\tau, t, x)$ je diskrétní tok vektorového pole f, viz* (6.27).

Lokální diskretizační chyba metody v bodě (t,x) a pro zvolené τ je definována jako

$$d(t+\tau, t, x) \equiv \|\phi(t+\tau, t, x) - \psi(t+\tau, t, x)\| \,. \tag{6.33}$$

Pokud existuje přirozené číslo $p \geq 1$ tak, že

$$d(t+\tau, t, x) = O(\tau^{p+1}) \quad \text{pro } \tau \to 0\,, \tag{6.34}$$

říkáme, že metoda je řádu p v bodě (t,x).

Věta 6.2. *Platí:*

1. *Jestliže $f \in C^1(J \times D, \mathbb{R}^n)$, potom Eulerova metoda je řádu $p = 1$*
2. *Jestliže $f \in C^2(J \times D, \mathbb{R}^n)$, potom Rungeho metoda je řádu $p = 2$*
3. *Jestliže $f \in C^4(J \times D, \mathbb{R}^n)$, potom Runge-Kuttova metoda je řádu $p = 4$*

v každém bodě $(t,x) \in J \times D$.

Důkaz Nechť $f \in C^1(J \times D, \mathbb{R}^n)$. Nechť $(t,x) \in J \times D$. Uvažujme Taylorův rozvoj toku vektorového pole vzhledem k τ. Stačí uvažovat první dva členy:

$$\phi(t+\tau, t, x) = \ x + \tau f + O(\tau^2)\,, \quad f = f(t,x)\,. \tag{6.35}$$

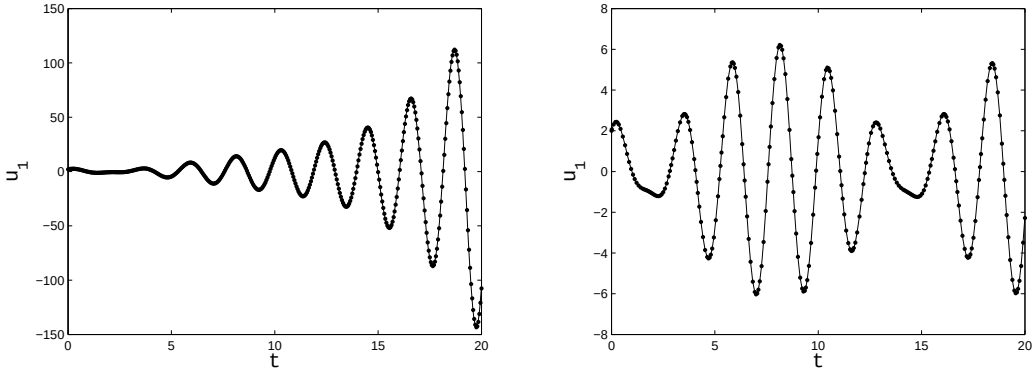
Z definice Eulerovy metody, viz (6.30),

$$\psi(t+\tau,t,x) = x + \tau\kappa_1 = x + \tau f\,, \quad f = f(t,x)\,.$$

Proto diskretizační chyba metody, viz (6.33), je řádu $d(t+\tau,t,x) = O(\tau^2)$.

Důkazy zbývajících tvrzení o řádu Rungeho metody a řádu Runge-Kuttovy metody vyžadují vyčíslení dalších členů Taylorova rozvoje (6.35). Např. v případě Runge-Kuttovy metody je nutné analyzovat dalších sedm členů rozvoje. K tomu je třeba dalších technických úprav (např. převodu soustavy na autonomní systém, který nezávisí explicitně na čase t). Omezíme se proto na citace výsledku: např. [8], Kap. 4, Theorem 4.18, str. 143. □

Na závěr uděláme experimentální srovnání Eulerovy metody a Runge-Kuttovy metody. Jako testovací úlohu volíme lineární oscilátor ((6.7) & (6.8), viz Obrázek 6.5) Použijeme ekvidistantní dělení s krokem $\tau = 0.05$ (Eulerova metoda) resp. $\tau = 0.2$ (Runge-Kuttova metoda). Vzhledem k Poznámce 6.3, soutěž obou metod má rovné podmínky. Srovnání je na Obrázku 6.9: Numerické řešení vlevo kvalitativně neodpovídá očekávání, protože netlumeně osciluje. Netlumených oscilací se zbavíme, pokud zvolíme τ dostatečně malé (např. $\tau = 0.01$). Z tohoto experimentálního srovnání vychází Runge-Kuttova metoda jako podstatně efektivnější.



Obrázek 6.9: Lineární oscilátor, první složka řešení u_1 vs. čas t. Eulerova metoda s krokem $\tau = 0.05$ (vlevo). Runge-Kuttova metoda s krokem $\tau = 0.2$ (vpravo). Parametry $b = 9$, $c = 10$, $\omega = 2.5$.

Vztah mezi přesným řešením a numerickým řešením počáteční úlohy (6.7) & (6.8) budeme zkoumat v následujícím odstavci. Chyba metody bude záviset na řádu použité metody, (6.34).

Poznámka 6.4. *Model lineárního oscilátoru (*(6.7) *&* (6.8)*) umíme integrovat t.j. nalézt explicitní formuli pro tok vektorového pole φ. Pro specifickou volbu parametrů*

$b = 9$, $c = 10$, $\omega = 3$, řešení počáteční úlohy netlumeně oscilují. Dochází k tzv. rezonanci. Takové řešení připomíná numerické řešení na Obrázku 6.9 vlevo.

Víme tedy, že netlumené oscilace jsou možným kandidátem na řešení. Jde o to, zda numericky zjištěné netlumené oscilace jsou numerickým artefaktem, jako na Obrázku 6.9 vlevo, nebo odpovídají realitě.

Z toho plyne obecné poučení: Numerické výpočty je třeba kriticky zkoumat ze všech stran.

6.4 Analýza konvergence

Na intervalu $[t_0, T]$, numericky řešíme (integrujeme) počáteční úlohu (6.13). Používáme jednokrokovou metodu (6.27).

Formulace hlavního výsledku si vyžádá zavedení nového pojmu:

Definice 6.10. (Přírůstková funkce) *Nechť f je Lipschitzovsky spojitá na $J \times D$. Nechť ψ je diskrétní tok vektorového pole* (6.27)*. Pro každé $t \in J$, $x \in D$, $\tau \geq 0$ položme*

$$\Psi(t, x, \tau) \equiv \frac{\psi(t + \tau, t, x) - x}{\tau} \in \mathbb{R}^n . \tag{6.36}$$

Funkci Ψ se říká přírůstková funkce metody.

Definiční obor přírůstkové funkce je odvozen z definičního oboru diskrétního toku vektorového pole. Rozeberme situaci podrobněji: Nejprve si uvědomíme, že lze zajistit existenci kompaktní podmnožiny $K \in D$ těchto vlastností:

1. $[t_0, T] \times K$ obsahuje celou trajektorii.
2. Pravá strana f je Lipschitzovsky spojitá na $[t_0, T] \times K$. Označme L konstantu Lipschitzovské spojitosti, viz (6.19).

Definiční obor přírůstkové funkce Ψ jistě obsahuje kompakt $[t_0, T] \times K \times [0, \tau_0]$, pokud $\tau_0 > 0$ je dostatečně malé. V tomto oboru je přírůstková funkce Ψ spojitá:

$$\Psi \in C([t_0, T] \times K \times [0, \tau_0]), \mathbb{R}^n) . \tag{6.37}$$

Předpokládejme, že $\Psi = \Psi(t, x, \tau)$ je dokonce Lipschitzovsky spojitá v proměnné x, t.j. existuje konstanta $\Lambda \geq 0$ tak, že

$$\|\Psi(t, x, \tau) - \Psi(t, y, \tau)\| \leq \Lambda \, \|x - y\| \tag{6.38}$$

pro každé x a y z K každé $t \in [t_0, T]$ a každé $\tau \in [0, \tau_0]$.

Později se ukáže souvislost mezi konstantou Λ a konstantou L Lipschitzovské spojitosti funkce f. Proto předpoklad (6.38) je často automaticky splněn.

Věta 6.3. (Konvergenční věta) *Uvažujme počáteční úlohu* (6.13). *Nechť řešení existuje na intervalu* $[t_0, T]$. *Uvažujme jednokrokovou metodu* (6.27) *s tím, že přírůstková funkce splňuje předpoklady* (6.37) *a* (6.38). *Nechť*

1. $u(t) = \varphi(t, t_0, x_0)\,, \quad t_0 \le t \le T$
2. $\{t_j\}_{j=0}^N\,, \quad t_{j+1} - t_j = \tau\,, \quad t_N \le T$
   $\{u_j\}_{j=0}^N \;:\quad u_{j+1} = u_j + \tau\,\Psi(t_j, u_j, \tau)$
3. $d(t+\tau, t, u(t)) \le C\,\tau^{p+1}\,, \quad t_0 \le t \le T$

Potom

$$||u(t_j) - u_j|| \le \frac{\Lambda(t_j - t_0) - 1}{\Lambda}\, C\tau^p\,, \quad t_0 \le t_j \le T\,, \quad j = 0, 1, \ldots, N\,. \tag{6.39}$$

Dříve než přistoupíme k důkazu, uveďme několik poznámek:

Poznámka 6.5. (k formulaci konvergenční věty)

ad 2. *Jeden krok metody* (6.29) *lze zapsat pomocí přírůstkové funkce*

$$\psi(t+\tau, t, x) = x + \tau\Psi(t, x, \tau)\,.$$

Předpokládáme ekvidistantní dělení s krokem $\tau > 0$. *Rekurzívním použitím metody dospějeme k posloupnosti přibližných řešení* $\{u_j\}_{j=0}^N$ *v diskrétních časech* $\{t_j\}_{j=0}^N$.

ad 3. *Předpoklad se týká odhadu diskretizační chybu podél celé trajektorie. Jde o kvalitativní přepoklad. Hodnotu konstanty* C *nelze zpravidla odhadnout. Jestliže např.* $f \in C^p(J \times D, \mathbb{R}^n)$, *potom taková konstanta existuje. To vyplývá z Věty 6.2, která je skriptu formulovaná pro tři konkrétní metody, ale lze ji dokázat pro velkou třídu jednokrokových metod.*

Důkaz Vyšetřeme vztah mezi dvěma po sobě jdoucími členy posloupnosti $u(t_{j+1})$, $u(t_j)$ a posloupnosti u_{j+1}, u_j. Z definice,

$$\begin{aligned} u(t_{j+1}) &= \varphi(t_j + \tau, t_j, u(t_j))\,, \\ u_{j+1} &= u_j + \tau\Psi(t_j, u_j, \tau)\,. \end{aligned}$$

Chceme odhadnout rozdíl $||u(t_{j+1}) - u_{j+1}||$ na základě rozdílu dvou předchůdců t.j., $||u(t_j) - u_j||$. Platí:

$$\begin{aligned} u(t_{j+1}) - u_{j+1} &= \varphi(t_j + \tau, t_j, u(t_j)) - \\ &\quad - u(t_j) - \tau\Psi(t_j, u(t_j), \tau) + \\ &\quad + u(t_j) + \tau\Psi(t_j, u(t_j), \tau) - \\ &\quad - u_j - \tau\Psi(t_j, u_j, \tau)\,. \end{aligned}$$

Použitím trojúhelníkových nerovností, a předpokladu (6.38), dospějeme k odhadu

$$\begin{aligned} ||u(t_{j+1}) - u_{j+1}|| &\leq ||\varphi(t_j+\tau, t_j, u(t_j)) - u(t_j) - \tau\Psi(t_j, u(t_j), \tau)|| + \\ &\quad + ||u(t_j) - u_j|| + \tau||\Psi(t_j, u(t_j), \tau) - \Psi(t_j, u_j, \tau)|| \leq \\ &\leq C\tau^{p+1} + (1+\tau\Lambda)\,||u(t_j) - u_j||\,. \end{aligned}$$

Položme

$$\varepsilon_j \equiv ||u(t_j) - u_j||\,, \quad j = 0, 1 \ldots, N\,.$$

Číslo ε_j je celková chyba aproximace trajektorie v čase t_j. Shrňme, že

$$\varepsilon_{j+1} \leq (1+\tau\Lambda)\varepsilon_j + C\tau^{p+1}\,, \quad j = 0, 1, \ldots, N-1\,. \tag{6.40}$$

Cílem je odhadnout ε_j. Rekurzivně použijeme (6.40) v tom smyslu, že odhadneme ε_1 na základě ε_0, ε_2 na základě ε_1, atd. až ε_j na základě ε_{j-1}. Dostaneme posloupnost nerovností:

$$\begin{aligned} \varepsilon_1 &\leq (1+\tau\Lambda)\varepsilon_0 + C\tau^{p+1} && |\cdot(1+\tau\Lambda)^{j-1} \\ \varepsilon_2 &\leq (1+\tau\Lambda)\varepsilon_1 + C\tau^{p+1} && |\cdot(1+\tau\Lambda)^{j-2} \\ &\vdots \\ \varepsilon_{j-2} &\leq (1+\tau\Lambda)\varepsilon_{j-3} + C\tau^{p+1} && |\cdot(1+\tau\Lambda)^2 \\ \varepsilon_{j-1} &\leq (1+\tau\Lambda)\varepsilon_{j-2} + C\tau^{p+1} && |\cdot(1+\tau\Lambda) \\ \varepsilon_j &\leq (1+\tau\Lambda)\varepsilon_{j-1} + C\tau^{p+1} \end{aligned}$$

Vytvoříme vpravo naznačené lineární kombinace, a nerovnosti sečteme. Připomeňme, že $\varepsilon_0 = 0$ v souladu s počáteční podmínkou. Dojdeme k závěru, že

$$\varepsilon_j \leq (1 + (1+\tau\Lambda) + \ \ldots\ + (1+\tau\Lambda)^{j-1})\ C\tau^{p+1} = \frac{(1+\tau\Lambda)^j - 1}{\tau\Lambda}\ C\tau^{p+1}\,,$$

kde vpravo jsme sečetli geometrickou řadu. K odhadu mocnin $(1+\tau\Lambda)^j$ využijeme exponenciely. Platí totiž, že

$$1 + \tau\Lambda \leq \mathrm{e}^{\tau\Lambda}\,, \quad \tau \geq 0\,.$$

Proto

$$\varepsilon_j \leq \frac{(1+\tau\Lambda)^j - 1}{\Lambda}\ C\tau^p \leq \frac{\mathrm{e}^{\tau j\Lambda} - 1}{\Lambda}\,C\tau^p = \frac{\mathrm{e}^{\Lambda(t_j - t_0)} - 1}{\Lambda}\,C\tau^p\,, \quad j = 0, 1, \ldots, N\,,$$

což je jiný zápis tvrzení (6.39). □

Věta 6.3 platí pro libovolnou jednokrokovou metodu ψ, pro kterou existuje přírůstková funkce Ψ viz (6.36), a Ψ je navíc Lipschitzovsky spojitá, viz (6.38). V Odstavci 6.3 jsme jako příklad jednokrokové metody uvedli Eulerovu metodu (6.30), Rungeho metodu (6.31) a Runge-Kuttovu metodu (6.32). Pro každou z nich existuje přírůstková funkce, která je Lipschitzovsky spojitá. Toto tvrzení dokážeme pro Rungeho metodu s tím, že pro další metody se postupuje analogicky.

Lemma 6.1. *Uvažujme Rungeho metodu, viz* (6.31). *Nechť* $f \in C([t_0, T], \mathbb{R}^n)$. *Nechť* f *je Lipschitzovsky spojitá t.j. existuje* $L > 0$ *tak, že*

$$\|f(t,x) - f(t,y)\| \leq L \, \|x - y\|$$

pro každé $t \in [t_0, T]$ *a každé* $x, y \in D$. *Potom přírůstková funkce* Ψ *je jednoznačně definovaná,*

$$\Psi(t,x,\tau) \equiv f\left(t + \frac{\tau}{2}, x + \frac{\tau}{2} f(t,x)\right).$$

Funkce $\Psi = \Psi(t,x,\tau)$ *je Lipschitzovsky spojitá v druhém argumentu s konstantou* $\Lambda = L\left(1 + \frac{\tau L}{2}\right)$.

Důkaz Formuli pro Ψ dostaneme přímo z definice Rungeho metody (6.31). Nechť $t \in [t_0, T]$, $x, y \in D$ a $\tau \geq 0$. Potom

$$\begin{aligned}
& ||\Psi(t,x,\tau) - \Psi(t,y,\tau)|| = \\
= \;& ||f\left(t + \frac{\tau}{2}, x + \frac{\tau}{2} f(t,x)\right) - f\left(t + \frac{\tau}{2}, y - \frac{\tau}{2} f(t,y)\right)|| \leq \\
\leq \;& L\,||x + \frac{\tau}{2} f(t,x) - y - \frac{\tau}{2} f(t,y)|| \leq \\
\leq \;& L\,||x - y|| + \frac{\tau}{2} L\,||f(t,x) - f(t,y)|| \leq \\
\leq \;& L\left(1 + \frac{\tau L}{2}\right) ||x - y||.
\end{aligned}$$

□

Na závěr uvedeme dvě poznámky:

Poznámka 6.6. (Apriorní odhad) *Věta 6.3 odhaduje chybu* $||u(t_j) - u_j||$ *numerické aproximace řešení v disktétních časech* $\{t_j\}_{j=0}^N$. *Od numerické integrace nemůžeme očekávat víc. Chceme-li odhadnout vzdálenost mezi přesným a přibližným řešením v zadaném čase* $t_j < t < t_{j+1}$, *můžeme použít lineární extrapolaci, viz* (6.23). *Programy, které graficky zbrazují numerické řešení, tuto možnost automaticky využívají. Větu 6.3 lze interpretovat jako apriorní odhad globální chyby řešení: Aniž bychom numerické řešení počítali, předem (t.j. apriori) jsme schopni vyslovit soud o kvalitě numerického řešení. Konstanty* C, Λ *a* L *sice fakticky neznáme, ale víme, že chyba řešení závisí na délce kroku* τ, *na řádu* p *zvolené metody, a také na hladkosti pravé strany* f.

Poznámka 6.7. (Aposteriorní odhad) *Je možné ovlivnit přesnost numerického řešení během výpočtu? Pomohlo by např., kdybychom v zadaném čase* t_j *dokázali odhadnout velikost lokální diskretizační chyby* $d(t_j + \tau, t_j, u_j)$. *To je možné za cenu dodatečné investice: Vyčíslení jednoho kroku metody, ale pro dvě volby délky kroku* τ *a* 2τ. *Z diference obou výsledků odhadneme požadovanou informaci o lokální diskretizační chybě. Tuto informaci jsme získali aposteriori, t.j. ze zkušenosti, v našem*

případě z porovnání dvou numerických experimentů. Na základě aposteriorní informace lze navrhnout strategii pro adaptivní volbu kroku τ, viz např. [6], str. 574, [8], Kap 5, Algorithm 5.2, str. 194. V algoritmech numerické integrace ODEs v systému `MATLAB` *se standartně využívá adaptivní volba délky integračního kroku τ.*

Kapitola 7

Problém vlastních čísel

Nechť $A \in \mathbb{R}^{n\times n}$. Připomeňme se základní pojmy:

Definice 7.1. (Charakteristický polynom) *Nechť $A \in \mathbb{R}^{n\times n}$. Definujme funkci $p = p(z)$,*

$$z \in \mathbb{C}^1 \longmapsto p(z) \equiv \det(zI - A) \in \mathbb{C}^1 \,,$$

kde $I \in \mathbb{R}^{n\times n}$ je identita. Funkci p říkáme charakteristický polynom.

Lze ukázat, že $p(z)$ je polynom stupně n s reálnými koeficienty. Koeficient u nejvyšší mocniny je jednička.

Definice 7.2. (Vlastní číslo, spektrum matice) *Nechť $A \in \mathbb{R}^{n\times n}$, nechť $p = p(z)$ je příslušný charakteristický polynom. Jestliže $\lambda \in \mathbb{C}^1$ je kořenem polynomu p t.j. $p(\lambda) = 0$, potom říkáme, že λ je vlastní číslo matice A. Označme $\sigma(A)$ množinu všech vlastních čísel matice A. Množině $\sigma(A)$ říkejme spektrum matice A.*

Protože polynom (s obecně komplexními koeficienty) stupně $n \geq 1$ má v komplexní rovině právě n kořenů (počítáme-li každý kořen tolikrát, kolikrát je jeho násobnost), potom charakteristický polynom $p = p(z)$ má n kořenů, a tedy

$$\sigma(A) = \{\lambda_1, \ldots, \lambda_n\} \,, \quad \lambda_j \in \sigma(A) \,, \quad j = 1, \ldots, n \,.$$

Navíc, polynom $p = p(z)$ má reálné koeficienty. Proto

$$\lambda \in \sigma(A) \Rightarrow \bar{\lambda} \in \sigma(A) \,.$$

To znamená, že komplexní vlastní čísla jsou ve spektru $\sigma(A)$ zastoupena v komplexně sdružených párech.

Definice 7.3. (Vlastní vektor) *Nechť $A \in \mathbb{R}^{n\times n}$. Nechť $\lambda \in \sigma(A)$. Nenulovému vektoru $\xi \in \mathbb{C}^n$, který splňuje podmínku*

$$A\xi = \lambda\xi$$

se říká vlastní vektor příslušný vlastnímu číslu λ.

V této kapitole probereme některé numerické metody řešení problému vlastních čísel. Tím se rozumí problém nalezení vlastních čísel a vlastních vektorů zadané matice $A \in \mathbb{R}^{n\times n}$. Je to velmi obtížný problém numerické lineární algebry, viz např. [4, 3].

Omezíme se na tzv. symetrický problém vlastních čísel. Budeme předpokládat, že matice $A \in \mathbb{R}^{n\times n}$ je symetrická, t.j. $A = A^{\mathrm{T}}$. Tento předpoklad znamená podstatné zjednodušení problému.

Věta 7.1. (Schurova věta) *Nechť $A \in \mathbb{R}^{n\times n}$, $A = A^{\mathrm{T}}$. Potom existuje ortonormální matice $Q \in \mathbb{R}^{n\times n}$ tak, že*

$$AQ = QD\,, \tag{7.1}$$

kde $D \in \mathbb{R}^{n\times n}$ je diagonální matice

$$D = \begin{bmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_n \end{bmatrix}. \tag{7.2}$$

Důkaz viz např. [4], Theorem 8.1.1, str. 393. □

Připomeňme, že ortonormální matice Q je regulární čtvercová matice, pro kterou $Q^{-1} = Q^{\mathrm{T}}$, viz Definice 2.1.

Nechť $Q(:, j) = \varphi_j \in \mathbb{R}^n$ je j-tý sloupec matice Q. Z (7.1) plyne, že

$$A\varphi_j = \lambda_j \varphi_j \tag{7.3}$$

pro $j = 1, 2, \ldots, n$. Z vlastností matice Q plyne, viz Definice 2.1, že

$$\varphi_i^{\mathrm{T}} \varphi_j = \delta_{ij}\,. \tag{7.4}$$

To znamená, že $||\varphi_j|| = 1$. Podle Definice 7.3, φ_j je vlastní vektor, který odpovídá vlastnímu číslu λ_j. Navíc, $\varphi_j \in \mathbb{R}^n$ je reálný vlastní vektor, který odpovídá reálnému vlastnímu číslu λ_j.

Ukázali jsme, že

$$\sigma(A) = \{\lambda_1, \ldots, \lambda_n\}\,, \quad \lambda_j \in \mathbb{R}^1\,, \quad j = 1, \ldots, n\,. \tag{7.5}$$

Bez újmy na obecnosti lze vlastní čísla λ_j (a odpovídající vlastní vektory φ_j, t.j. sloupce matice Q) uspořádat tak, aby

$$|\lambda_1| \geq |\lambda_2| \geq \cdots \geq |\lambda_{n-1}| \geq |\lambda_n|\,. \tag{7.6}$$

Jinými slovy, platí tvrzení Věty 7.1 s dodatečnou podmínkou (7.6).

Díky vlastnosti (7.4), vektory $\varphi_j \in \mathbb{R}^n$, $j = 1, \ldots, n$, tvoří ortonormální bázi prostoru $\mathbb{R}^n$,

$$\mathbb{R}^n = \mathrm{span}\,\{\varphi_1, \varphi_2, \ldots, \varphi_{n-1}, \varphi_n\}\,. \tag{7.7}$$

Pro každý vektor $x \in \mathbb{R}^n$, $x = (x_1, \ldots, x_n)^{\mathrm{T}}$, existuje právě jeden $a \in \mathbb{R}^n$, $a = (a_1, \ldots, a_n)^{\mathrm{T}}$ tak, že

$$x = \sum_{j=1}^{n} a_j \, \varphi_j \, . \tag{7.8}$$

Matice Q reprezentuje převodní vztah mezi kartézskými souřednicemi $x \in \mathbb{R}^n$ a novými souřadnicemi $a \in \mathbb{R}^n$:

$$Qa = x \, , \quad Q^{-1}x = Q^{\mathrm{T}}x = a \, .$$

V celé kapitole předpokládáme, že $A \in \mathbb{R}^{n \times n}$, $A = A^{\mathrm{T}}$. Vlastní čísla $\lambda_j \in \mathbb{R}^1$ jsou uspořádána podle velikosti, viz (7.6). Odpovídající vlastní vektory $\varphi_j \in \mathbb{R}^n$ jsou normalizovány, t.j. $||\varphi_j|| = 1$.

7.1 Mocninná metoda a její modifikace

Definice 7.4. (Dominatní vlastní číslo) *Nechť*

$$|\lambda_1| > |\lambda_2| \geq \cdots \geq |\lambda_{n-1}| \geq |\lambda_n| \, . \tag{7.9}$$

Potom říkáme, že $\lambda_1 \in \sigma(A)$ je dominatní vlastní číslo. Příslušnému vlastnímu vektoru $\varphi_1 \in \mathbb{R}^n$ říkáme dominatní vlastní vektor.

Odvodíme metodu, která aproximuje dominantní vlastní číslo a příslušný vlastní vektor. Analyzujme odděleně případy, kdy dominatní vlastní číslo λ_1 bude kladné (Lemma 7.1) resp. záporné (Lemma 7.2).

Lemma 7.1. *Nechť*

$$\lambda_1 > |\lambda_2| \geq \cdots \geq |\lambda_{n-1}| \geq |\lambda_n| \, . \tag{7.10}$$

Nechť je dáno $x^{(0)} \in \mathbb{R}^n$, $x^{(0)} \neq 0$. Uvažujme rozvoj (7.8) t.j., $x^{(0)} = \sum_{j=1}^{n} a_j \, \varphi_j$. Nechť $a_1 \neq 0$. Definujme posloupnosti

$$\left\{x^{(k)}\right\}_{k=0}^{+\infty} , \quad x^{(k)} \in \mathbb{R}^n \; : \quad x^{(k)} = Ax^{(k-1)} \tag{7.11}$$

$$\left\{y^{(k)}\right\}_{k=0}^{+\infty} , \quad y^{(k)} \in \mathbb{R}^n \; : \quad y^{(k)} = \frac{x^{(k)}}{||x^{(k)}||} \, . \tag{7.12}$$

Platí:

$$y^{(k)} \to \frac{a_1}{|a_1|} \, \varphi_1 \, , \quad \left(y^{(k)}\right)^{\mathrm{T}} A y^{(k)} \to \lambda_1 \tag{7.13}$$

pro $k \to +\infty$.

Důkaz Z definice posloupnosti (7.11) indukcí odvodíme, že

$$x^{(k)} = Ax^{(k-1)} = A^2x^{(k-2)} = \cdots = A^kx^{(0)}\,.$$

Využitím rozkladu (7.8),

$$x^{(k)} = A^kx^{(0)} = A^k\sum_{j=1}^{n} a_j\varphi_j = \sum_{j=1}^{n} a_jA^k\varphi_j = \sum_{j=1}^{n} a_j\lambda_j^k\varphi_j\,.$$

Zde jsme využili definice vlastního čísla $A\varphi_j = \lambda_j\varphi_j$. Je třeba se uvědomit, že $A^2\varphi_j = A(A\varphi_j) = A(\lambda_j\varphi_j) = \lambda_j^2\varphi_j$. Indukcí, $A^k\varphi_j = \lambda_j^k\varphi_j$.

Formuli pro $x^{(k)}$ dále upravujme:

$$\begin{aligned} x^{(k)} &= \textstyle\sum_{j=1}^{n} a_j\lambda_j^k\varphi_j = a_1\lambda_1^k\varphi_1 + \sum_{j=2}^{n} a_j\lambda_j^k\varphi_j = \\ &= a_1\lambda_1^k\left(\varphi_1 + \textstyle\sum_{j=2}^{n}\frac{a_j}{a_1}\left(\frac{\lambda_j}{\lambda_1}\right)^k\varphi_j\right) = \\ &\equiv a_1\lambda_1^k\,z_k\,, \end{aligned}$$

kde $z_k \in \mathbb{R}^k$. Poznamenejme, že z předpokladu dominance (7.9) plyne, že pokud $2 \le j \le n$, potom

$$\left(\frac{\lambda_j}{\lambda_1}\right)^k \to 0 \quad \text{pro} \quad k \to +\infty\,. \tag{7.14}$$

Z definice vektorů z_k potom plyne, že

$$z_k \to \varphi_1 \quad \text{pro} \quad k \to +\infty\,. \tag{7.15}$$

Z definice (7.12),

$$y^{(k)} \equiv \frac{x^{(k)}}{||x^{(k)}||} = \frac{a_1\lambda_1^k\,z_k}{|a_1|\,|\lambda_1^k|\,||z_k||}\,. \tag{7.16}$$

V důsledku předpokladu (7.10) je $\frac{\lambda_1^k}{|\lambda_1^k|} = 1$. Z konvergence (7.15) plyne konvergence

$$y^{(k)} \to \frac{a_1}{|a_1|}\,\varphi_1 \quad \text{pro} \quad k \to +\infty\,. \tag{7.17}$$

Tím je dokázána první část tvrzení.

Z (7.17) dostaneme, že $Ay^{(k)} \to \lambda_1\,\frac{a_1}{|a_1|}\,\varphi_1$. Tedy, že i

$$(y^{(k)})^{\mathrm{T}}Ay^{(k)} \to \lambda_1\,\varphi_1^{\mathrm{T}}\varphi_1 = \lambda_1 \quad \text{pro} \quad k \to +\infty\,. \tag{7.18}$$

□

Lemma 7.2. *Nechť*

$$|\lambda_1| > |\lambda_2| \geq \cdots \geq |\lambda_{n-1}| \geq |\lambda_n|\,, \quad \lambda_1 < 0\,. \tag{7.19}$$

Nechť je dáno $x^{(0)} \in \mathbb{R}^n$, $x^{(0)} \neq 0$. *Uvažujme rozvoj* (7.8) *t.j.,* $x^{(0)} = \sum_{j=1}^{n} a_j\,\varphi_j$. *Nechť* $a_1 \neq 0$. *Definujme posloupnosti* $\left\{x^{(k)}\right\}_{k=0}^{+\infty}$ *a* $\left\{y^{(k)}\right\}_{k=0}^{+\infty}$ *podle* (7.11) *a* (7.12). *Platí:*

$$y^{(2l)} \to \frac{a_1}{|a_1|}\,\varphi_1\,, \quad y^{(2l+1)} \to -\frac{a_1}{|a_1|}\,\varphi_1 \tag{7.20}$$

pro $l \to +\infty$ *a*

$$\left(y^{(k)}\right)^{\mathrm{T}} A y^{(k)} \to \lambda_1 \tag{7.21}$$

pro $k \to +\infty$.

Důkaz Generujeme posloupnost $\left\{x^{(k)}\right\}_{k=0}^{+\infty}$ rekurencí (7.11). Postupujeme jako v případě důkazu předchozího lemmatu: Definujeme $z_k \in \mathbb{R}^n$ tak, že $x^{(k)} = a_1\lambda_1^k z_k$. Z (7.14) vyplývá (7.15).

Generujeme $\left\{y^{(k)}\right\}_{k=0}^{+\infty}$ podle (7.12). Zjistíme, že platí (7.16). Při vyhodnocení $y^{(k)}$ záleží na tom, zda index k je sudý nebo lichý: V důsledku předpokladu (7.19) je $\dfrac{\lambda_1^{2l}}{|\lambda_1^{2l}|} = 1$ a $\dfrac{\lambda_1^{2l+1}}{|\lambda_1^{2l+1}|} = -1$. K výroku (7.20) o konvegenci dospějeme analogickou argumentací jako k výrokům (7.17) a (7.18). □

Algoritmus 7.1. (Mocninná metoda) *Dáno:* $v^{(0)} \in \mathbb{R}^n$, $||v^{(0)}|| = 1$. *Definujeme posloupnosti* $\left\{v^{(k)}\right\}_{k=0}^{\infty}$ *a* $\left\{\lambda^{(k)}\right\}_{k=1}^{\infty}$ *rekurencí*

$$\begin{aligned} w^{(k-1)} &= A v^{(k-1)} \\ v^{(k)} &= \frac{w^{(k-1)}}{||w^{(k-1)}||} \\ \lambda^{(k)} &= (v^{(k)})^{\mathrm{T}} A v^{(k)}\,. \end{aligned} \tag{7.22}$$

Posloupnost $\left\{w^{(k)}\right\}_{k=0}^{\infty}$ nás nezajímá. Je to pomocná posloupnost. Ukážeme souvislost mezi posloupností $\left\{v^{(k)}\right\}_{k=0}^{\infty}$, kterou generuje Algoritmus 7.1, a posloupnostmi $\left\{x^{(k)}\right\}_{k=0}^{\infty}$ a $\left\{y^{(k)}\right\}_{k=0}^{\infty}$, kterých se týká Lemma 7.1 resp. Lemma 7.2. Nejprve jedno připomenutí:

Poznámka 7.1. *Nechť* $A \in \mathbb{R}^{n\times n}$, $q \in \mathbb{R}^n$, $q \neq 0$. *Nechť* $r \in \mathbb{R}^1$, $r > 0$. *Potom* $A(rq) = rAq$ *a* $||A(rq)|| = r||Aq||$. *To znamená, že*

$$\frac{A(rq)}{||A(rq)||} = \frac{rAq}{r||Aq||} = \frac{Aq}{||Aq||}\,.$$

Nechť $x^{(0)} \in \mathbb{R}^n$, $x^{(0)} \neq 0$. Nechť $v^{(0)} = \dfrac{x^{(0)}}{||x^{(0)}||}$. Definujme posloupnosti $\left\{x^{(k)}\right\}_{k=0}^{\infty}$ a $\left\{y^{(k)}\right\}_{k=0}^{\infty}$ podle (7.11) a (7.12). Definujme posloupnost $\left\{v^{(k)}\right\}_{k=0}^{\infty}$ podle (7.22). Indukcí lze dokázat, že

$$v_k = y_k\,, \quad k = 0, 1, \ldots\,. \tag{7.23}$$

Při důkazu uplatníme Poznámku 7.1.

Věta 7.2. (Konvergence mocninné metody) *Nechť $v^{(0)} \in \mathbb{R}^n$, $||v^{(0)}|| = 1$. Předpokládejme, že $(v^{(0)})^{\mathrm{T}}\varphi_1 \neq 0$. Uvažujme posloupnosti $\left\{v^{(k)}\right\}_{k=0}^{\infty}$ a $\left\{\lambda^{(k)}\right\}_{k=0}^{\infty}$ generované metodou* (7.22).

Potom $\lambda^{(k)} \to \lambda_1$. Jestliže

$\lambda_1 > 0$, potom $v^{(k)} \to \varphi_1$ pro $k \to +\infty$

$\lambda_1 < 0$, potom $v^{(2l)} \to \varphi_1$ a $v^{(2l+1)} \to -\varphi_1$ pro $l \to +\infty$.

Důkaz Položme $x^{(0)} = v^{(0)}$. Definujme posloupnosti $\left\{x^{(k)}\right\}_{k=0}^{\infty}$ a $\left\{y^{(k)}\right\}_{k=0}^{\infty}$ podle (7.11) a (7.12). Víme, že $\left\{v^{(k)}\right\}_{k=0}^{\infty} = \left\{y^{(k)}\right\}_{k=0}^{\infty}$, viz (7.23). Tvrzení Věty 7.2 vyplývá z Lemmatu 7.1 resp. z Lemmatu 7.2. Ukážeme že předpoklad $x^{(0)} = v^{(0)} = \sum_{j=1}^{n} a_j\,\varphi_j$, $a_1 \neq 0$ (viz Lemma 7.1 resp. Lemma 7.2) je ekvivalentní předpokladu $(v^{(0)})^{\mathrm{T}}\varphi_1 \neq 0$ (viz Věta 7.2). Platí: $\varphi_1^{\mathrm{T}} v^{(0)} = \varphi_1^{\mathrm{T}} \sum_{j=1}^{n} a_j\,\varphi_j = a_1$. Tedy $a_1 \neq 0$ právě když $\varphi_1^{\mathrm{T}} v^{(0)} = (v^{(0)})^{\mathrm{T}}\varphi_1 \neq 0$. □

Poznámka 7.2.

1. *Na základě vlastnosti* (7.14) *lze odhadnou rychlost konvergence iteračního procesu $\left\{v^{(k)}\right\}_{k=0}^{\infty}$, viz [3], Theorem 5.6, str. 194: Chybu k-té iterace lze zhora odhadnout jako $C\left|\dfrac{\lambda_2}{\lambda_1}\right|^k$, kde $C = \left(\sum_{i=2}^{n}(a_i/a_1)\right)^{1/2}$.*

   Kvalitativně řečeno, čím větší resp. menší bude poměr $|\lambda_2|/|\lambda_1|$, tím rychlejší resp. pomalejší bude konvergence.

2. *Numerickou metodu často aplikujeme, aniž bychom ověřovali předpoklady příslušné konvergenční věty. Např. předpoklad $(v^{(0)})^{\mathrm{T}}\varphi_1 \neq 0$ Věty 7.2 nelze ověřit, aniž bychom předem znali řešení, t.j. vektor φ_1. Nicméně by to byla velká náhoda, kdyby tento předpoklad nebyl splněn. Říkáme, že předpoklad $(v^{(0)})^{\mathrm{T}}\varphi_1 \neq 0$ je splněn generically.*

3. *Zakončovací kriterium: Během iterací sledujeme např. residuum $r^{(k)} \equiv Av^{(k)} - \lambda^{(k)}v^{(k)}$. Jestliže norma $||r^{(k)}||$ poklesne pod zadanou mez, iterace ukončíme s tím, že $\lambda^{(k)}$ je aproximací dominatního vlastního čísla λ_1 a vektor $v^{(k)}$ je aproximací dominatního vlastního vektoru φ_1.*

Uvedeme metodu inverzní iterace, která umožní aproximovat libovolné vlastní číslo $\lambda_j \in \sigma(A)$ (a příslušný vlastní vektor $\varphi_j \in \mathbb{R}^n$) na které si ukážeme. Jak si na vlastní číslo ukážeme, když spektrum $\sigma(A)$ neznáme?

Nechť $\mu \in \mathbb{R}^1$, $\mu \notin \sigma(A)$. Uvažujme $\lambda_j \in \sigma(A)$. Nechť $\varphi_j \in \mathbb{R}^n$ je odpovídající normalizovaný vlastní vektor, t.j.

$$A\varphi_j = \lambda_j \varphi_j \,, \quad ||\varphi_j|| = 1 \,. \tag{7.24}$$

Zřejmě platí:

$$(A - \mu I)\varphi_j = (\lambda_j - \mu)\varphi_j \,, \tag{7.25}$$

kde $I \in \mathbb{R}^{n\times n}$ je idetita. Parametru μ se říká posun spektra.

Matice $A - \mu I \in \mathbb{R}^{n\times n}$ je regulární matice. Dokážeme to sporem: Nechť $\det(A - \mu I) = 0$. Potom existuje $x \in \mathbb{R}^n$, $x \neq 0$ tak, že $(A - \mu I)x = 0$ t.j. $Ax = \mu x$. To znamená, že vektor x je vlastní vektor, který odpovídá vlastnímu číslu μ matice A t.j., $\mu \in \sigma(A)$. Ale to je spor (s předpokladem $\mu \notin \sigma(A)$).

Proto

$$(A - \mu I)^{-1}\varphi_j = (\lambda_j - \mu)^{-1}\varphi_j \,. \tag{7.26}$$

Lze tedy shrnout, že

$$\lambda_j \in \sigma(A) \iff \lambda_j - \mu \in \sigma(A - \mu I) \iff (\lambda_j - \mu)^{-1} \in \sigma(A - \mu I)^{-1}$$

s tím, že odpovídající vlastní vektor $\varphi_j \in \mathbb{R}^n$ je stejný.

Vyjdeme z (7.26). Platí:

$$(A-\mu I)^{-2}\varphi_j = (A-\mu I)^{-1}((A-\mu I)^{-1}\varphi_j) = (A-\mu I)^{-1}((\lambda_j-\mu)^{-1}\varphi_j) = (\lambda_j-\mu)^{-2}\varphi_j \,.$$

Indukcí lze ukázat, že

$$(A - \mu I)^{-k}\varphi_j = (\lambda_j - \mu)^{-k}\varphi_j \,, \tag{7.27}$$

kde k je libovolné celé číslo.

Budeme definovat dva iterační procesy, které jsou analogické (7.11) a (7.12): Nechť je dáno $x^{(0)} \in \mathbb{R}^n$, $x^{(0)} \neq 0$. Uvažujme rozvoj (7.8) t.j., $x^{(0)} = \sum_{j=1}^{n} a_j \varphi_j$. Definujme posloupnosti

$$\left\{x^{(k)}\right\}_{k=0}^{+\infty} \,, \quad x^{(k)} \in \mathbb{R}^n \,: \quad x^{(k)} = (A - \mu I)^{-1}x^{(k-1)} \tag{7.28}$$

$$\left\{y^{(k)}\right\}_{k=0}^{+\infty} \,, \quad y^{(k)} \in \mathbb{R}^n \,: \quad y^{(k)} = \frac{x^{(k)}}{||x^{(k)}||} \,. \tag{7.29}$$

Na základě (7.27) lze ukázat, že

$$x^{(k)} = (A - \mu I)^{-k}x^{(0)} = \sum_{j=1}^{n}(\lambda_i - \mu)^{-k}a_i\varphi_i \,. \tag{7.30}$$

Parametr $\mu \notin \sigma(A)$, t.j. posun spektra, je pevně zvolen. Předpokládejme, že existuje $\lambda_j \in \sigma(A)$ tak, že

$$|\lambda_j - \mu| < |\lambda_i - \mu| \qquad \forall \lambda_i \in \sigma(A)\,, \quad i \neq j\,. \tag{7.31}$$

Předpokládejme, že $x^{(0)}$ je zvolen tak, že $a_j \neq 0$ t.j.

$$(x^{(0)})^{\mathrm{T}} \varphi_j \neq 0\,. \tag{7.32}$$

Budeme postupovat podobně jako v důkazu Lemmatu 7.1. Budeme upravovat formuli pro $x^{(k)}$:

$$\begin{aligned} x^{(k)} &= \textstyle\sum_{j=1}^{n} (\lambda_i - \mu)^{-k} a_i \varphi_i = (\lambda_j - \mu)^{-k} a_j \varphi_j + \sum_{j=1, i\neq j}^{n} (\lambda_i - \mu)^{-k} a_i \varphi_i = \\ &= a_j (\lambda_j - \mu)^{-k} \left(\varphi_j + \textstyle\sum_{i=1, i\neq j}^{n} \frac{a_i}{a_j} \left(\frac{\lambda_j - \mu}{\lambda_i - \mu} \right)^k \varphi_i \right) = \\ &\equiv a_j (\lambda_j - \mu)^{-k}\, z_k\,, \end{aligned}$$

kde $z_k \in \mathbb{R}^k$.

Z předpokladu (7.31) plyne, že pro každý index $i = 1, \ldots, n$, $i \neq j$,

$$\left(\frac{\lambda_j - \mu}{\lambda_i - \mu} \right)^k \to 0 \quad \text{pro} \quad k \to +\infty\,. \tag{7.33}$$

Z definice vektorů z_k potom plyne, že

$$z_k \to \varphi_1 \quad \text{pro} \quad k \to +\infty\,. \tag{7.34}$$

Podle definice (7.29),

$$y^{(k)} \equiv \frac{x^{(k)}}{||x^{(k)}||} = \frac{a_j (\lambda_j - \mu)^{-k}\, z_k}{|a_j|\, |(\lambda_j - \mu)^{-k}|\, ||z_k||}\,. \tag{7.35}$$

Máme dvě varianty:

1. Jestliže $\lambda_j > \mu$, potom $\dfrac{(\lambda_j - \mu)^k}{|(\lambda_j - \mu)^k|} = 1$, a tedy

$$y^{(k)} \to \frac{a_j}{|a_j|} \varphi_j \quad \text{pro} \quad k \to +\infty\,. \tag{7.36}$$

2. Jestliže $\lambda_j < \mu$, potom $\dfrac{(\lambda_j - \mu)^{2l}}{|(\lambda_j - \mu)^{2l}|} = 1$ a $\dfrac{(\lambda_j - \mu)^{2l+1}}{|(\lambda_j - \mu)^{2l+1}|} = -1$, a proto

$$y^{(2l)} \to \frac{a_j}{|a_j|} \varphi_j\,, \quad y^{(2l+1)} \to -\frac{a_j}{|a_j|} \varphi_j \quad \text{pro} \quad l \to +\infty\,. \tag{7.37}$$

Pro obě varianty platí, že

$$\left(y^{(k)}\right)^{\mathrm{T}} A y^{(k)} \to \lambda_j \quad \text{pro} \quad k \to +\infty\,. \tag{7.38}$$

Algoritmus 7.2. (Inverzní iterace) *Dáno:* $\mu \in \mathbb{R}^1$, $v^{(0)} \in \mathbb{R}^n$, $||v^{(0)}|| = 1$. *Definujeme posloupnosti* $\left\{v^{(k)}\right\}_{k=0}^{\infty}$ *a* $\left\{\lambda^{(k)}\right\}_{k=0}^{\infty}$ *rekurencí*

$$\begin{aligned} w^{(k-1)} &= (A-\mu I)^{-1}\, v^{(k-1)} \\ v^{(k)} &= \frac{w^{(k-1)}}{||w^{(k-1)}||} \\ \lambda^{(k)} &= (v^{(k)})^{\mathrm{T}} A v^{(k)}\,. \end{aligned} \tag{7.39}$$

Poznámka 7.3. (Implementace kroku (7.39)) *Při výpočtu* $w^{(k-1)}$ *nepočítáme inverzi* $(A-\mu I)^{-1}\, v^{(k-1)}$. *Místo toho, řešíme soustavu lineárních rovnic pro neznámou* $w^{(k-1)}$*:*

$$(A-\mu I)\, w^{(k-1)} = v^{(k-1)}\,.$$

Věta 7.3. (Konvergence metody inverzní iterace) *Nechť je splněn předpoklad* (7.31). *Nechť* $(v^{(0)})^{\mathrm{T}}\varphi_j \neq 0$. *Potom platí* (7.36)–(7.38).

Důkaz Položme $x^{(0)} = v^{(0)}$. Definujme posloupnosti $\left\{x^{(k)}\right\}_{k=0}^{\infty}$ a $\left\{y^{(k)}\right\}_{k=0}^{\infty}$ podle (7.28) a (7.29). Víme, že $\left\{v^{(k)}\right\}_{k=0}^{\infty} = \left\{y^{(k)}\right\}_{k=0}^{\infty}$, viz (7.23). Z analýzy (7.27)–(7.35) těchto posloupností dospějeme k tvrzení (7.36)–(7.38). □

Poznámka 7.4.

1. *Obecně (t.j. genericky), iterace konvergují právě k tomu vlastnímu číslu* $\lambda_j \in \sigma(A)$, *které je nejblíže zadanému posunu* $\mu \in \mathbb{R}^1$ *spektra matice A. Samozřejmě v takové pozici může být více prvků spektra.*

2. *Podobně jako v Poznámce 7.2(3) budeme v každé iteraci* (7.39) *sledovat reziduum* $r^{(k)} \equiv Av^{(k)} - \lambda^{(k)}v^{(k)}$. *Jestliže norma* $||r^{(k)}||$ *poklesne pod zadanou mez, iterace ukončíme s tím, že* $\lambda^{(k)}$ *je aproximací vlastního čísla* λ_j *a vektor* $v^{(k)}$ *je aproximací vlastního vektoru* φ_j.

Poznámka 7.5. (Aposteriorní odhad chyby) *Lze odhadnout vzdálenost* $|\lambda_j - \lambda^{(k)}|$*? Naše přání vypadá jako nesmysl, vždyť* λ_j *počítáme a tedy je neznáme. Vzdálenost* $|\lambda_j - \lambda^{(k)}|$ *lze odhadnout na základě rezidua* $r^{(k)}$ *a dalšího dodatečného výpočtu. Viz [3] str. 193–196. Takovému odhadu říkáme aposteriorní odhad. Můžeme garantovat chybu výpočtu.*

Příklad 7.1. *Nechť*

$$A = \begin{bmatrix} -261 & 209 & -49 \\ -530 & 422 & -98 \\ -800 & 631 & -144 \end{bmatrix}$$

Je známo ([4], Example 7.3.1 str. 331), že $\sigma(A) = \{\lambda_1 = 10, \lambda_2 = 4, \lambda_3 = 3\}$.

—	$\lambda^{(17)} = 10.00000001080428$	17	err = 0.00000455687055	λ_1
$\mu = 0$	$\lambda^{(46)} = 3.00000011474064$	46	err = 0.00000846848200	λ_3
$\mu = 3.8$	$\lambda^{(12)} = 3.99999921065176$	12	err = 0.00000349633125	λ_2
$\mu = 5$	$\lambda^{(23)} = 3.99999842130463$	23	err = 0.00000699266061	λ_2
$\mu = 8$	$\lambda^{(22)} = 10.00000009866281$	22	err = 0.00000632478817	λ_1

Tabulka 7.1: Mocninná metoda. Inverzní iterace pro $\mu = 0, 3.8, 5, 8$.
Veličina err udává aposteriorní odhad chyby. Je menší, než garantovaná chyba 10^{-5}.

Budeme testovat Algoritmus 7.1 a Algoritmus 7.2. Poznamenejme, že matice A není symetrická. Data:

$$v^{(0)} = [1, 0, 0]^{\mathrm{T}}, \quad \text{garantovaná chyba: } 10^{-5}.$$

Výsledky experimentu jsou shrnuty v Tabulce 7.1.

7.2 QR metoda

Připomeňme předpoklady celé kapitoly: Nechť $A \in \mathbb{R}^{n\times n}$, $A = A^{\mathrm{T}}$. Označme $\sigma(A)$ spektrum matice A. Vlastní čísla $\lambda_j \in \sigma(A)$ jsou reálná. Bez újmy na obecnosti je lze seřadit podle velikosti v absolutní hodnotě, viz (7.6).

Zkonstruujeme iterační proces

$$\{A^{(k)}\}_{k=0}^{+\infty}, \quad A^{(k)} \in \mathbb{R}^{n\times n}, \quad A^{(0)} = A,$$

který bude konvergovat k diagonální matici

$$A^{(k)} \longrightarrow \begin{bmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_n \end{bmatrix},$$

přičemž $\lambda_j \in \sigma(A)$, $j = 1, \ldots, n$. Vlastní čísla λ_j dokonce respektují uspořádání (7.6). Budeme schopni aproximovat <u>všechna</u> vlastní čísla λ_j a odpovídající vlastní vektory $\varphi_j \in \mathbb{R}^n$, $j = 1, \ldots, n$.

V dalším budeme potřebovat následující pojem:

Definice 7.5. (Ortonormální ekvivalence) *Nechť $A \in \mathbb{R}^{n\times n}$ a $B \in \mathbb{R}^{n\times n}$. Nechť existuje ortonormální matice $Q \in \mathbb{R}^{n\times n}$ tak, že*

$$AQ = QB. \tag{7.40}$$

Potom říkáme, že A a B jsou <u>ortonormálně ekvivalentní</u>. Pro tento vztah matic A a B zavedeme označení

$$A \sim B. \tag{7.41}$$

Zřejmě $A \sim B$ právě když $B \sim A$. Pojem ortonormální ekvivalence zavádíme pro obecně nesymetrické matice.

Poznámka 7.6. *Nechť $A \sim B$, $AQ = QB$, kde $Q \in \mathbb{R}^{n\times n}$ je ortonormální matice. Potom*

$$\sigma(A) = \sigma(B)\,. \tag{7.42}$$

Nechť $\lambda \in \sigma(A)$. Nechť $\xi \in \mathbb{C}^n$, $||\xi|| = 1$, je odpovídající vlastní vektor, t.j.

$$A\xi = \lambda\xi\,.$$

Potom

$$B\,Q^{\mathrm{T}}\xi = \lambda Q^{\mathrm{T}}\xi\,.$$

To znamená, že $\zeta = Q^{\mathrm{T}}\xi$, $||\zeta|| = 1$ je vlastní vektor odpovídající vlastnímu číslu $\lambda \in \sigma(B)$.

7.2.1 QR iterace

Uvažujme následující

Algoritmus 7.3. *Nechť $A \in \mathbb{R}^{n\times n}$, $A = A^{\mathrm{T}}$. Generujeme posloupnosti matic*

$$\{A^{(k)}\}_{k=0}^{+\infty}\,, \quad A^{(k)} \in \mathbb{R}^{n\times n}\,, \quad \{Q^{(k)}\}_{k=1}^{+\infty}\,, \quad Q^{(k)} \in \mathbb{R}^{n\times n}$$

pomocí rekurence: $A^{(0)} = A$,

- *definujeme QR-rozklad matice $A^{(k-1)}$:*

$$[Q^{(k)}, R^{(k)}] \;=\; \mathrm{qr}(A^{(k-1)})\,, \tag{7.43}$$

  kde $Q^{(k)} \in \mathbb{R}^{n\times n}$ je ortonormální a $R^{(k)} \in \mathbb{R}^{n\times n}$ je horní trojúhelníková matice,

- *položme*

$$A^{(k)} = R^{(k)}\,Q^{(k)}\,. \tag{7.44}$$

Požadovaný QR-rozklad matice $A^{(k-1)}$, viz Věta 2.2, není jednoznačně určen. Rovnice (7.43) má formu příkazového řádku v jazyce `MATLAB` kde qr označuje funkci příslušného rozkladu. V důsledku rozkladu (7.43) je $Q^{(k)}R^{(k)} = A^{(k-1)}$. To znamená, že $R^{(k)} = (Q^{(k)})^{\mathrm{T}}A^{(k-1)}$. Z definice kroku (7.44) dostaneme, že $A^{(k)} = R^{(k)}\,Q^{(k)} = (Q^{(k)})^{\mathrm{T}}A^{(k-1)}\,Q^{(k)}$. Shrňme:

$$A^{(k)} = (Q^{(k)})^{\mathrm{T}}A^{(k-1)}\,Q^{(k)}\,, \quad A^{(k)} \sim A^{(k-1)}\,. \tag{7.45}$$

Příklad 7.2. *Uvažujme (symetrickou) matici A, viz* (7.47)*. Aplikujme Algoritmus 7.3. Po padesáti iteracích dostaneme matici* $A^{(50)}$*, viz* (7.48)*. Mimodiagonální prvky jsou malé,*

$$A^{(50)}(i,j) \leq 3e{-}4\,, \quad i \neq j\,.$$

Diagonální prvky $A^{(50)}(j,j)$*,* $j = 1, \ldots, n$*,*

$$3.9787\,, \quad 2.9925\,, \quad 2.4883\,, \quad -1.9906\,, \quad 1.4914\,, \quad 1.0027\,, \quad -0.4930 \tag{7.46}$$

jsou seřazeny sestupně v absolutní hodnotě. Z monotónní konvergence diagonálních prvků matic $A^{(k)}$*,* $k = 1, \ldots, 50$ *lze usoudit, že posloupnost* (7.46) *aproximuje vlastní čísla* $\lambda_1, \ldots \lambda_7$ *matice A s přesností na čtyři platné cifry.*

$$A = \begin{bmatrix} 1.83 & 0.75 & -0.01 & 0.85 & -0.13 & -0.64 & 1.12 \\ 0.75 & 0.40 & 0.06 & 0.74 & 0.45 & -1.29 & 0.23 \\ -0.01 & 0.06 & 1.73 & 0.63 & 0.80 & -0.70 & 1.63 \\ 0.85 & 0.74 & 0.63 & 1.75 & -0.20 & -0.32 & -1.01 \\ -0.13 & 0.45 & 0.80 & -0.20 & 2.42 & -0.04 & -0.22 \\ -0.64 & -1.29 & -0.70 & -0.32 & -0.04 & 0.84 & 0.43 \\ 1.12 & 0.23 & 1.63 & -1.01 & -0.22 & 0.43 & 0.50 \end{bmatrix} \tag{7.47}$$

$$A^{(50)} = \begin{bmatrix} 3.9787 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 \\ 0.0000 & 2.9925 & 0.0003 & 0.0000 & 0.0000 & 0.0000 & 0.0000 \\ 0.0000 & 0.0003 & 2.4883 & -0.0000 & -0.0000 & 0.0000 & -0.0000 \\ 0.0000 & 0.0000 & -0.0000 & -1.9906 & -0.0000 & 0.0000 & -0.0000 \\ 0.0000 & 0.0000 & -0.0000 & -0.0000 & 1.4914 & -0.0000 & -0.0000 \\ -0.0000 & -0.0000 & -0.0000 & 0.0000 & -0.0000 & 1.0027 & -0.0000 \\ 0.0000 & -0.0000 & 0.0000 & -0.0000 & 0.0000 & 0.0000 & -0.4930 \end{bmatrix} \tag{7.48}$$

Jak aproximovat příslušné vlastní vektory? Využijeme posloupnost $\{Q^{(k)}\}_{k=1}^{+\infty}$. Rekurzivní aplikací rovnice (7.45) dostaneme

$$A^{(k)} = (Q^{(k)})^{\mathrm{T}} A^{(k-1)}\, Q^{(k)}\,, \quad A^{(k-1)} = (Q^{(k-1)})^{\mathrm{T}} A^{(k-2)}\, Q^{(k-1)}\,,$$

a tedy

$$A^{(k)} = (Q^{(k)})^{\mathrm{T}} (Q^{(k-1)})^{\mathrm{T}} A^{(k-2)} Q^{(k-1)} Q^{(k)}\,, \quad A^{(k)} \sim A^{(k-2)}\,.$$

Indukcí zjistíme, že $A^{(k)} \sim A^{(0)}$. Konkrétně,

$$A^{(k)} = (Q^{(k)})^{\mathrm{T}} \ \ldots \ (Q^{(1)})^{\mathrm{T}} A^{(0)} Q^{(1)} \ \ldots \ Q^{(k)}\,.$$

Přitom $A = A^{(0)}$. Položme

$$M^{(k)} \equiv Q^{(1)} \ \ldots \ Q^{(k)}\,, \quad k = 1, 2, \ldots\,. \tag{7.49}$$

Matice $M^{(k)}$ jsou ortonormální, protože jsou definovány jako součiny ortonormálních matic. Dospěli jsme k závěru, že

$$A\, M^{(k)} = M^{(k)}\, A^{(k)}\,, \quad A \sim A^{(k)}\,, \tag{7.50}$$

pro každé $k = 1, 2, \ldots$. Jestliže posloupnost $A^{(k)}$ konverguje k diagonální matici,

$$A^{(k)} \to \begin{bmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_n \end{bmatrix} \quad \text{pro} \quad k \to +\infty\,, \tag{7.51}$$

jak ukazuje Příklad 7.2, potom posloupnost ortonormálních matic $\{M^{(k)}\}_{k=1}^{+\infty}$, $M^{(k)} \in \mathbb{R}^{n\times n}$, konverguje k nějaké ortonormální matici $Q \in \mathbb{R}^{n\times n}$,

$$M^{(k)} \to Q \quad \text{pro} \quad k \to +\infty\,. \tag{7.52}$$

Sloupce matice Q jsou potom vlastní vektory, viz (7.3). Navíc, vlastní čísla respektují uspořádání (7.6).

Pro generování posloupnosti $\{M^{(k)}\}_{k=1}^{+\infty}$ můžeme použít rekurence $M^{(0)} = I$,

$$M^{(k)} = M^{(k-1)}\, Q^{(k)}\,,$$

která vychází z definice (7.49).

Algoritmus 7.4. (QR iterace) *Nechť $A \in \mathbb{R}^{n\times n}$, $A = A^{\mathrm{T}}$. Definujme posloupnosti matic*

$$\{A^{(k)}\}_{k=0}^{+\infty}\,, \quad A^{(k)} \in \mathbb{R}^{n\times n}\,, \quad \{M^{(k)}\}_{k=0}^{+\infty}\,, \quad M^{(k)} \in \mathbb{R}^{n\times n}$$

pomocí rekurence: $A^{(0)} = A$, $M^{(0)} = I$,

- *definujeme QR-rozklad matice $A^{(k-1)}$:*

$$[Q^{(k)}, R^{(k)}] \;=\; \mathrm{qr}(A^{(k-1)})\,, \tag{7.53}$$

  kde $Q^{(k)} \in \mathbb{R}^{n\times n}$ je ortonormální a $R^{(k)} \in \mathbb{R}^{n\times n}$ je horní trojúhelníková matice,

- *položme*

$$M^{(k)} = M^{(k-1)}\, Q^{(k)}\,. \tag{7.54}$$

Jestliže

$$|\lambda_1| > |\lambda_2| > \cdots > |\lambda_n|\,,$$

potom Algoritmus 7.4 konverguje, t.j. platí (7.51) a (7.52), viz [3], Property 5.9, str. 202. Nicméně základní algoritmus lze vylepšit.

7.2.2 Redukce na třídiagonální tvar

Nechť $A \in \mathbb{R}^{n\times n}$, $A = A^{\mathrm{T}}$. Zkonstruujeme (finitním algoritmem, s omezeným počtem aritmetických operací) matici $G \in \mathbb{R}^{n\times n}$, $G = G^{\mathrm{T}}$, která je třídiagonální, a která

je ortonormálně ekvivalentní s A t.j. $A \sim G$. To podle Definice 7.5 znamená, že existuje ortonormální matice $N \in \mathbb{R}^{n\times n}$ tak, že

$$A\,N = N\,G\,. \tag{7.55}$$

Konstrukci třídiagonální matice G uvedených vlastností se říká redukce zadané symetrické matice A na třídiagonální tvar. Představíme (numerickou) metodu, která využívá matic Givensových rotací (viz Odst. 2.4).

Příklad 7.3. *Uvažujme matici $A \in \mathbb{R}^{7\times 7}$, viz* (7.47)*. Matice $G \in \mathbb{R}^{7\times 7}$, viz* (7.56)*, je příslušná redukce. Prvky matice G na diagonále a obou vedlejších diagonálách jsou standartně zaokrouhleny na pět platných cifer. Ostatní prvky matice G by měly být nulové (v přesné aritmetice), ale vhledem k zaokroulovacím chybám (t.j. v konečné aritmetice) jsou rovny řádově 10^{-15}. Ortonormální matici $N \in \mathbb{R}^{7\times 7}$, která vystupuje v podmínce* (7.55)*, nebudeme uvádět.*

$$G = \begin{bmatrix} 1.8300 & 1.7222 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 \\ 1.7222 & 0.9567 & 1.9944 & 0.0000 & 0.0000 & 0.0000 & 0.0000 \\ 0.0000 & 1.9944 & 0.7416 & 1.3705 & 0.0000 & 0.0000 & 0.0000 \\ 0.0000 & 0.0000 & 1.3705 & 0.7861 & 1.4819 & 0.0000 & 0.0000 \\ 0.0000 & 0.0000 & 0.0000 & 1.4819 & 1.2965 & 0.6190 & 0.0000 \\ 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.6190 & 1.8255 & 0.7173 \\ 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.7173 & 2.0337 \end{bmatrix} \tag{7.56}$$

Pro porozumění dalšího výkladu je třeba, aby si čtenář připomenul

- Definici 2.3 Givensovy rotace (pro matice $n \times n$):

$$G_{ij} \in \mathbb{R}^{n\times n}\,, \quad 1 \le i < j \le n\,, \quad \text{s parametry } c, s\,:\; c^2 + s^2 = 1\,.$$

- Tvrzení 2.2 o vlastnostostech lineární transformace, která je definována maticí G_{ij}:

$$x \in \mathbb{R}^n \longmapsto G_{ij}\,x = y \in \mathbb{R}^n\,.$$

  Pro volbu $c = \frac{x_i}{r}$, $c = \frac{x_j}{r}$, $r = \sqrt{(x_i^2 + x_j^2)}$ dostaneme, že $y_j = 0$, $y_i = r$.

- Příklad 2.1 konstrukce QR-rozkladu zadané matice A.

Modifikujme uvedený příklad na případ čtvercové matice $A \in \mathbb{R}^{3\times 3}$. Hledáme $Q \in \mathbb{R}^{3\times 3}$ ortonormální, a $R \in \mathbb{R}^{3\times 3}$ horní trojúhelníkovou matici tak, aby $Q\,R = A$. V rozkladu vystupují matice $G_{23} \in \mathbb{R}^{3\times 3}$ a $G_{12} \in \mathbb{R}^{3\times 3}$,

$$G_{23} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & c & s \\ 0 & -s & c \end{bmatrix}\,, \quad G_{12} = \begin{bmatrix} c & s & 0 \\ -s & c & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

pro konkrétní volby parametrů c a s. Konstruujeme tři transformace

$$G_{23}^{(1)} \in \mathbb{R}^{3\times 3}\,, \quad G_{12}^{(1)} \in \mathbb{R}^{3\times 3}\,, \quad G_{23}^{(2)} \in \mathbb{R}^{3\times 3}$$

tak, aby jejich složením vznikla horní trojúhelníková matice $R \in \mathbb{R}^{3\times 3}$,

$$R = G_{23}^{(2)}\, G_{12}^{(1)}\, G_{23}^{(1)}\, A\,.$$

Potom

$$Q = (G_{23}^{(1)})^{\mathrm{T}}\, (G_{12}^{(1)})^{\mathrm{T}}\, (G_{23}^{(2)})^{\mathrm{T}}\,, \quad Q\, R = A\,.$$

QR-rozklad matice A přímo nesouvisí s redukcí symetrické matice na třídiagonální tvar. Nicméně, algoritmus redukce pracuje se součiny Givensových transformací. Takže výše uvedený příklad lze chápat jako užitečnou rozcvičku.

Příklad 7.4. *Nechť $A \in \mathbb{R}^{3\times 3}$, $A = A^{\mathrm{T}}$,*

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}\,.$$

Definujme $G_{23} \in \mathbb{R}^{3\times 3}$, s parametry

$$c = \frac{a_{21}}{r}\,, \quad s = \frac{a_{31}}{r}\,, \quad r = \sqrt{(a_{21}^2 + a_{31}^2)}\,.$$

Nechť $r \neq 0$. Potom

$$G \equiv G_{23}\, A\, (G_{23})^{\mathrm{T}} = \begin{bmatrix} * & * & 0 \\ * & * & * \\ 0 & * & * \end{bmatrix}$$

je symetrická, $G = G^{\mathrm{T}}$, třídiagonální. Položíme $N \equiv (G_{23})^{\mathrm{T}}$, viz (7.55).

Rozeberme si celou konstrukci podrobněji: Vhodnou volbou parametrů c a s,

$$G_{23}A = \begin{bmatrix} 1 & 0 & 0 \\ 0 & c & s \\ 0 & -s & c \end{bmatrix} \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ r & * & * \\ 0 & * & * \end{bmatrix}\,.$$

Transpozicí, a využitím symetrie dostaneme

$$(G_{23}A)^{\mathrm{T}} = A^{\mathrm{T}}(G_{23})^{\mathrm{T}} = A(G_{23})^{\mathrm{T}} = \begin{bmatrix} a_{11} & r & 0 \\ a_{12} & * & * \\ a_{13} & * & * \end{bmatrix}\,.$$

Další aplikací transformace G_{23},

$$G_{23}A(G_{23})^{\mathrm{T}} = G_{23} \begin{bmatrix} a_{11} & r & 0 \\ a_{12} & * & * \\ a_{13} & * & * \end{bmatrix} = \begin{bmatrix} a_{11} & r & 0 \\ r & \boxed{*} & \boxed{*} \\ 0 & \boxed{*} & \boxed{*} \end{bmatrix}$$

dostaneme nakonec symetrickou matici, což je zdůrazněno symboly $\boxed{*}$.

Příklad 7.5. *Nechť $A \in \mathbb{R}^{4\times 4}$, $A = A^{\mathrm{T}}$. Redukce proběhne ve třech krocích: Položme $A^{(1)} = A$ a $N = I$, kde I je identita 4×4. Definujme*

$$
\begin{aligned}
G_{34}^{(1)} A^{(1)} (G_{34}^{(1)})^{\mathrm{T}} &= \begin{bmatrix} * & * & * & 0 \\ * & * & * & * \\ * & * & * & * \\ 0 & * & * & * \end{bmatrix} = A^{(2)}, \qquad N := N\,(G_{34}^{(1)})^{\mathrm{T}}, \\
G_{23}^{(1)} A^{(2)} (G_{23}^{(1)})^{\mathrm{T}} &= \begin{bmatrix} * & * & 0 & 0 \\ * & * & * & * \\ 0 & * & * & * \\ 0 & * & * & * \end{bmatrix} = A^{(3)}, \qquad N := N\,(G_{23}^{(1)})^{\mathrm{T}}, \\
G_{34}^{(2)} A^{(2)} (G_{34}^{(2)})^{\mathrm{T}} &= \begin{bmatrix} * & * & 0 & 0 \\ * & * & * & 0 \\ 0 & * & * & * \\ 0 & 0 & * & * \end{bmatrix} = G, \qquad N := N\,(G_{34}^{(2)})^{\mathrm{T}}.
\end{aligned}
$$

Potom

$$A \sim G, \quad AN = NG, \quad N = (G_{34}^{(1)})^{\mathrm{T}} (G_{23}^{(1)})^{\mathrm{T}} (G_{34}^{(2)})^{\mathrm{T}}.$$

Poznámka 7.7. *Alternativní technikou (které se v současnosti dává přednost) je Householderova třídiagonalizace, viz [4], Algorithm 8.3.1, str. 415., s kalkulací nákladů* #flops $= 4\frac{n^3}{3}$. *Náklady redukce na třídiagonální tvar pomocí Givensových rotací jsou zhruba stejné.*

7.2.3 QR algoritmus

Budeme sledovat tuto strategii: Zadanou symetrickou matici $A \in \mathbb{R}^{n\times n}$

1. budeme redukovat na třídiagonální matici $G \in \mathbb{R}^{n\times n}$ tak, že $A \sim G$,
2. na matici $G \in \mathbb{R}^{n\times n}$ budeme aplikovat QR-iterace.

Výhoda tohoto postupu: V prvním kroku platíme fixní náklady na redukci G, viz Poznámka 7.7. V druhém kroku aplikujeme QR-iterace na matici G. Jestliže označíme

$$\{G^{(k)}\}_{k=0}^{+\infty}, \quad G^{(k)} \in \mathbb{R}^{n\times n}, \quad G^{(0)} = G \tag{7.57}$$

posloupnost QR-iterací matice G, potom každá z matic $G^{(k)}$ bude třídiagonální a symetrická (v přesné aritmetice, přičemž v konečné aritmerice tento výrok platí s malou chybou, řádově rovnou 10^{-15}). Proto stačí monitorovat konvergenci diagonálních prvků a prvků na jedné z vedlejších diagonál matic $G^{(k)}$.

Příklad 7.6. *Uvažujme symetrickou matici $A \in \mathbb{R}^{7\times 7}$ z* (7.47). *V Příkladu 7.3 jsme matici A redukovali na třídiagonální matici $G \in \mathbb{R}^{7\times 7}$, viz* (7.56) *tak, že $G \sim A$. Uvažujme iterační proces* (7.57). *Jako příklad uvádíme iteraci $G^{(10)}$, viz* (7.59)

a $G^{(40)}$*, viz* (7.60)*. Teorie zaručuje konvergenci diagonálních prvků* $\mathrm{diag}(G^{(k)}) \in \mathbb{R}^7$ *matic* $G^{(k)}$ *k vlastním číslům matice* A*. Registrujeme sice lokálně monotónní konvergenci, ale aproximace různých vlastních čísel konvergují s různou rychlostí. V tabulce* (7.58) *jsou uvedeny složky vektoru* $\mathrm{diag}(G^{(40)}) \in \mathbb{R}^7$ *s tím, že hvězdička* $*$ *označuje první nespolehlivou platnou cifru aproximace vlastního čísla* $\lambda_j \in \sigma(A)$*,* $i = 1, 2, \ldots, n$*.*

$$\begin{aligned} \lambda_1 &= 3.9786934451* \\ \lambda_2 &= 2.992506* \\ \lambda_3 &= 2.4881* \\ \lambda_4 &= -1.9904* \\ \lambda_5 &= 1.49143329834* \\ \lambda_6 &= 1.0026766470466* \\ \lambda_7 &= -0.49298943862350* \end{aligned} \tag{7.58}$$

$$G^{(10)} = \begin{bmatrix} 3.9787 & -0.0369 & -0.0000 & 0.0000 & -0.0000 & 0.0000 & -0.0000 \\ -0.0369 & 2.9288 & 0.4306 & -0.0000 & -0.0000 & 0.0000 & -0.0000 \\ -0.0000 & 0.4306 & -1.6062 & -1.2007 & 0.0000 & -0.0000 & -0.0000 \\ 0.0000 & -0.0000 & -1.2007 & 2.1677 & 0.0376 & 0.0000 & 0.0000 \\ 0.0000 & -0.0000 & 0.0000 & 0.0376 & 1.4923 & -0.0139 & 0.0000 \\ -0.0000 & 0.0000 & -0.0000 & -0.0000 & -0.0139 & 1.0030 & -0.0117 \\ 0.0000 & -0.0000 & 0.0000 & 0.0000 & -0.0000 & -0.0117 & -0.4929 \end{bmatrix} \tag{7.59}$$

$$G^{(40)} = \begin{bmatrix} 3.9787 & -0.0000 & 0.0000 & 0.0000 & -0.0000 & 0.0000 & -0.0000 \\ -0.0000 & 2.9925 & -0.0000 & 0.0000 & -0.0000 & -0.0000 & -0.0000 \\ 0.0000 & -0.0000 & 2.4883 & -0.0000 & 0.0000 & 0.0000 & 0.0000 \\ -0.0000 & 0.0000 & -0.0000 & -1.9906 & -0.0000 & 0.0000 & 0.0000 \\ 0.0000 & -0.0000 & -0.0000 & -0.0000 & 1.4914 & -0.0000 & 0.0000 \\ -0.0000 & 0.0000 & 0.0000 & 0.0000 & -0.0000 & 1.0027 & -0.0000 \\ 0.0000 & -0.0000 & -0.0000 & -0.0000 & -0.0000 & -0.0000 & -0.4930 \end{bmatrix} \tag{7.60}$$

Zformulujeme algoritmus, který umožní aproximovat všechna vlastní čísla a vlastní vektory zadané matice:

Algoritmus 7.5. (QR algoritmus) *Nechť* $A \in \mathbb{R}^{n\times n}$*,* $A = A^{\mathrm{T}}$*.*

Krok 1. *Redukce matice* A *na třídiagonální tvar: Konstrukce matic* $G \in \mathbb{R}^{n\times n}$ *a* $N \in \mathbb{R}^{n\times n}$ *vlastností:*

$$A\,N = N\,G\,, \quad G \text{ je symetrická a třídiagonální}\,, \quad N \text{ je ortonormální}.$$

Krok 2. QR iterace *matice* G*: Generujeme posloupnosti matic*

$$\{G^{(k)}\}_{k=0}^{+\infty}\,,\ G^{(k)} \in \mathbb{R}^{n\times n}\,,\ G^{(0)} = G\,, \quad \{M^{(k)}\}_{k=0}^{+\infty}\,,\ M^{(k)} \in \mathbb{R}^{n\times n}\,,\ M^{(0)} = I$$

podle (7.53) *&* (7.54)*.*

Jestliže

$$|\lambda_1| > |\lambda_2| > \cdots > |\lambda_n|\,,$$

potom Algoritmus 7.5 konverguje:

$$\mathrm{diag}(G^{(k)}) \;\to\; [\lambda_1, \ldots, \lambda_n]^{\mathrm{T}} \quad \text{pro} \;\; k \to +\infty\,, \tag{7.61}$$

kde $\lambda_j \in \sigma(A)$, $j = 1, 2, \ldots, n$. Navíc platí:

$$M^{(k)} \;\to\; Q \in \mathbb{R}^{n\times n} \quad \text{pro} \;\; k \to +\infty\,.$$

Sloupce součinu matic $NQ \in \mathbb{R}^{n\times n}$ jsou vlastní vektory matice A:

$$NQ = [\varphi_1, \ldots, \varphi_j, \ldots, \varphi_n]\,, \quad \varphi_j \in \mathbb{R}^n\,, \quad A\,\varphi_j = \lambda_j \varphi_j\,, \quad j = 1, \ldots, n\,. \tag{7.62}$$

Při verifikaci výroku (7.62) uplatníme Poznámku 7.6.

Poznámka 7.8. *Příklad 7.6 naznačuje, že Algoritmus 7.5 nefunguje za všech okolností spolehlivě. Zejména, aproximuje různá vlastní čísla s různou přesností. O různých modifikacích QR algoritmu si lze přečíst např. v [4], Chapter 8, §8.3 The Symmetric QR Algorithm.*

Všechny tři základní algoritmy této kapitoly t.j. Mocninná metoda (Algoritmus 7.9), Inverzní iterace (Algoritmus 7.11) a QR algoritmus (Algoritmus 7.59) předpokládají, že matice $A \in \mathbb{R}^{n\times n}$ je symetrická. Za jistých okolností lze předpoklad symetrie matice zmírnit:

Definice 7.6. (Normální matice) *Nechť $A \in \mathbb{R}^{n\times n}$. Jestliže*

$$A^{\mathrm{T}}A = A^{\mathrm{T}}A\,,$$

potom říkáme, že A je normální matice.

Každá symetrická matice $A \in \mathbb{R}^{n\times n}$ je normální.

Výše uvedené algoritmy fungují i pro matice, které jsou normální. Musíme ovšem počítat s tím, že vlastní čísla $\lambda \in \sigma(A)$ mohou být komplexní, vystupují v komplexně sdružených párech, a příslušné vlastní vektory mohou být komplexní.

Kapitola 8

Iterační metody řešení soustav lineárních rovnic

Nechť $A \in \mathbb{R}^{n\times n}$, $b \in \mathbb{R}^n$. Nechť $\det A \neq 0$. Hledáme $x \in \mathbb{R}^n$ tak, aby

$$Ax^* = b\,. \tag{8.1}$$

Vracíme se tedy k úloze, kterou jsme řešili v Kapitole 1. Používali jsme eliminační techniky, které byly spojeny s LU-rozkladem nebo s Choleského rozkladem matice A. Co se týče kalkulace ceny výpočtů, dospěli jsme k odhadu #flops $\frac{2}{3}n^3$ (pro Gaussovu eliminaci).

V této Kapitole 8 budeme uvažovat iterační metody řešení úlohy (8.1): Nechť $x^{(0)} \in \mathbb{R}^n$ je aproximace řešení x^*. Budeme definovat celou posloupnost $\{x^{(k)}\}_{k=0}^{+\infty}$ aproximací $x^{(k)} \in \mathbb{R}^n$ hledaného řešení. Za jistých okolností $x^{(k)} \to x^*$ pro $k \to +\infty$. Výpočet se ukončí, jakmile

$$||x^{(k)} - x^*|| < \mathrm{tol}\,, \tag{8.2}$$

kde tol je předem zadaná tolerance. Důležitou výhodou iteračních metod je možnost zadat si toleranci předem. Nemusí nás přece zajímat všechny platné cifry numerického řešení. Z hlediska kalkulace ceny výpočtu můžeme podstatně ušetřit, pokud se spokojíme s velkou tolerancí výpočtu.

K podmínce (8.2) poznamenejme, že řešení x^* není samozřejmě známo. Nicméně v každém iteračním kroku se můžeme pokusit vzdálenost $||x^{(k)} - x^*||$ alespoň odhadnout aposteriori, doslova ze zkušenosti. Přesněji, na základě dodatečných výpočtů. S aposteriorními odhady jsme se již setkali v jiných souvislostech, viz Poznámku 6.7 resp. Poznámku 7.5. Přirozeným kandidátem aposteriorního odhadu je residuum, t.j. zbytek,

$$||b - Ax^{(k)}|| < \mathrm{tol}\,. \tag{8.3}$$

Toto kriterium se běžně používá. Nelze jej však používat nekriticky. Více na toto téma viz [5], 8.6 Zastavovací kriteria, str. 203.

8.1 Klasické iterační metody

Řešíme soustavu (8.1). V celém odstavci předpokádejme, že $\det A \neq 0$.

Nechť

$$B \in \mathbb{R}^{n\times n}\,, \quad \det B \neq 0 \tag{8.4}$$

je pevně zvolená matice. Potom $x^* \in \mathbb{R}^n$ bude řešením soustavy (8.1) právě když

$$Ax^* = b \quad \Longleftrightarrow \quad B^{-1}(b - Ax^*) = 0\,.$$

Tedy $x^* \in \mathbb{R}^n$ bude pevným bodem

$$\begin{aligned} x^* &= x^* + B^{-1}(b - Ax^*)\,, \\ x^* &= \underbrace{(I - B^{-1}A)}_{\equiv G}\, x^* + \underbrace{B^{-1}b}_{\equiv c}\,. \end{aligned}$$

Položme

$$G = I - B^{-1}A\,, \quad c = B^{-1}b\,. \tag{8.5}$$

Definice 8.1. (Iterační matice) *Matici $G \in \mathbb{R}^{n\times n}$, viz* (8.5)*, se říká iterační matice.*

Definujme zobrazení $g : \mathbb{R}^n \longrightarrow \mathbb{R}^n$

$$x \in \mathbb{R}^n \longmapsto g(x) \equiv Gx + c \in \mathbb{R}^n\,. \tag{8.6}$$

Ukázali jsme, že

$$Ax^* = b \quad \Longleftrightarrow \quad x^* = g(x^*)\,. \tag{8.7}$$

To znamená, že $x^* \in \mathbb{R}^n$ je řešením soustavy lineárních rovnic (8.1) právě když $x^* \in \mathbb{R}^n$ je pevným bodem zobrazení (8.6).

Připomeňme Odstavec 3.2, Věta o pevném bodě:

1. Pevný bod x^* lze najít jednoduchou numerickou metodou (tzv. iterace zobrazení g):

$$x^{(0)} \in \mathbb{R}^n\,, \quad \{x^{(k)}\}_{k=0}^{+\infty} \;:\quad x^{(k+1)} = g(x^{(k)})\,, \tag{8.8}$$

kde $x^{(0)}$ je zadaná aproximace pevného bodu x^*. S uvážením konkrétní definice g, viz (8.6), dostaneme iterační formuli

$$x^{(k+1)} = Gx^{(k)} + c = (I - B^{-1}A)\,x^{(k)} + B^{-1}b\,, \tag{8.9}$$

což lze přepsat jako

$$Bx^{k+1} = (B - A)\,x^{(k)} + b\,. \tag{8.10}$$

V každém iteračním kroku musíme vyřešit soustavu lineárních rovnic s maticí $B \in \mathbb{R}^{n\times n}$, viz (8.4). Iterační metoda má rozumný smysl jenom v tom případě, kdy řešení soustavy (8.10) bude laciné: Matice B bude v nějakém speciálním tvaru.

2. Zobrazení g, viz (8.6), musí splňovat jisté předpoklady. Musí být kontrahující, viz Definice 3.2: Nechť existuje $\theta > 0$ tak, že

$$||g(x) - g(y)|| \leq \theta ||x - y|| \qquad \forall x, y \in \mathbb{R}^n \,. \tag{8.11}$$

Jestliže $\theta < 1$, potom g je kontrahující.

Zobrazení (8.6) je afinní. Proto

$$g(x) - g(y) = Gx - Gy = G(x - y) \,.$$

Podmínka (8.11) je evivalentní s podmínkou

$$||Gz|| \leq \theta ||z|| \qquad \forall z \in \mathbb{R}^n \,.$$

Snadno dospějeme k následující postačující podmínce kontraktivnosti zobrazení g:

$$\max_{||z||=1} ||Gz|| < 1 \,. \tag{8.12}$$

Podmínka (8.12) na iterační matici G zaručí, že iterační proces (8.8) bude konvergovat pro libovolnou volbu počáteční aproximace $x^{(0)} \in \mathbb{R}^n$.

Podmínku kontraktivnosti lze zformulovat nezávisle na použité normě. Budeme ji charakterizovat pomocí spektrálních vlastností iterační matice $G \in \mathbb{R}^{n \times n}$. Připomeneme pojem spektra matice G, viz Definice 7.2,

$$\sigma(G) = \{\lambda \in \mathbb{C} \ : \ \det(\lambda I - G) = 0\} \,. \tag{8.13}$$

Definice 8.2. (Spektrální poloměr matice) *Položme*

$$\varrho(G) = \max_{\lambda \in \sigma(G)} |\lambda| \,. \tag{8.14}$$

Číslu $\varrho(G)$ se říká spektrální poloměr dané matice G.

Následující tvrzení uvedeme bez důkazu:

Věta 8.1. *Iterační proces* (8.8) *bude konvergovat pro libovolnou volbu počáteční aproximace $x^{(0)} \in \mathbb{R}^n$ právě když $\varrho(G) < 1$.*

Důkaz Viz např. [2], Věta 5.4, str. 81, nebo [4], Theorem 10.1.1, str. 511. □

Konkrétní volbou matice $B \in \mathbb{R}^{n \times n}$ dostaneme tři klasické iterační metody: Richardsonovu metodu, Jacobiovu metodu a Gaussovu-Seidelovu metodu. Pro každou z nich uvedeme iterační krok (8.10) s případnou ekvivalentní modifikací, a příslušnou iterační maticí $G \in \mathbb{R}^{n \times n}$. K definici těchto metod potřebujeme následující rozklad matice $A \in \mathbb{R}^{n \times n}$:

$$A = L + D + R \,, \tag{8.15}$$

kde

$D \in \mathbb{R}^{n\times n}$ je diagonální matice,

$L \in \mathbb{R}^{n\times n}$ je ostře dolní trojúhelníková matice (mnemotechnicky "left"),

$R \in \mathbb{R}^{n\times n}$ je ostře horní trojúhelníková matice (mnemotechnicky "right").

Definice 8.3. (Richardsonova metoda) *Položme $B = I$, kde $I \in \mathbb{R}^{n\times n}$ je identita. Potom*

$$x^{(k+1)} = (I - A)x^{(k)} + b\,. \tag{8.16}$$

Iterační matice:

$$G = I - A\,. \tag{8.17}$$

Definice 8.4. (Jacobiova metoda) *Položme $B = D$. Potom*

$$Dx^{(k+1)} = (D - A)x^{(k)} + b\,,$$

t.j.

$$Dx^{(k+1)} = -(L + R)x^{(k)} + b\,. \tag{8.18}$$

Iterační matice:

$$G = -D^{-1}(L + R)\,. \tag{8.19}$$

Definice 8.5. (Gaussova-Seidelova metoda) *Položme $B = D + L$. Potom*

$$(D + L)x^{(k+1)} = (D + L - A)x^{(k)} + b\,,$$

t.j.

$$(D + L)x^{(k+1)} = -Rx^{(k)} + b\,. \tag{8.20}$$

Iterační matice:

$$G = -(D + L)^{-1}R\,. \tag{8.21}$$

Poznámka 8.1. (Náklady na jednu iteraci) *Diskutujme řešení soustavy* (8.20)*. Jedná se řešení soustavy s dolní trojúhelníkovou maticí. Lze použít algoritmu typu substituce vpřed, se kterým jsme se např. setkali v analýze Algoritmu 1.1 Gaussovy eliminace. Došli jsme k závěru, že substituce vpřed potřebuje* $\#\text{flops} = n^2 + O(n)$*. To je tedy cena jednoho iteračního kroku.*

Gaussova eliminace, jako přímá metoda, potřebuje $\#\text{flops} = \frac{2}{3}n^3 - n^2/2 + O(n)$*. Z toho vyplývá, že použití iterační metody je výhodné, pokud k dosažení pořadované přesnosti (viz* (8.2)*) budeme potřebovat méně (obvykle podstatně méně) iteračních kroků než n.*

Uvedeme dvě tvrzení, která se týkají konvergence Richardsonovy metody (Věta 8.2) a Gaussovy-Seidelovy metody (Věta 8.4). Výroky obou tvrzení se budou týkat spekter iteračních matic $G \in \mathbb{R}^{n\times n}$ příslušných metod. Díky Větě 8.1 si je můžeme interpretovat jako konvergenční věty.

Věta 8.2. *Nechť $A \in \mathbb{R}^{n\times n}$, $A = A^{\mathrm{T}}$. Uvažujme Richardsonovu metodu, kde $G = I - A$ je příslušná iterační matice, viz* (8.17). *Označme $\lambda_{\min}(A)$ resp. $\lambda_{\max}(A)$ nejmenší resp. největší vlastní číslo matice A. Potom*

$$\varrho(G) < 1 \quad \Longleftrightarrow \quad 0 < \lambda_{\min}(A) \leq \lambda_{\max}(A) < 2\,. \tag{8.22}$$

Důkaz

1. Protože A je symetrická, potom je i matice G symetrická.

2. Vlastní čísla obou matic A a G jsou reálná. Vlastní vektory obou matic jsou reálné.

3. Nechť $\lambda \in \sigma(A)$. Nechť $x \in \mathbb{R}^n$, $||x|| = 1$, $Ax = \lambda x$, je příslušný vlastní vektor. Potom
$$Gx = (I - A)x = (1 - \lambda)x\,.$$
Tedy $\mu = 1 - \lambda \in \sigma(G)$ a x je odpovídající vlastní vektor. Zřejmě platí ekvivalence:
$$\lambda \in \sigma(A) \quad \Longleftrightarrow \quad \mu = 1 - \lambda \in \sigma(G)\,.$$
Příslušné vlastní vektory jsou stejné.

Proto

$$\begin{aligned} \varrho(G) \;&\equiv\; \max_{\mu\in\sigma(G)} |\mu| \;= \\ &=\; \max_{\lambda_{\min}(A)\leq\lambda\leq\lambda_{\max}(A)} |1-\lambda| = \max(|1-\lambda_{\min}(A)|, |1-\lambda_{\max}(A)|) \end{aligned} \tag{8.23}$$

Tvrzení věty vyplývá z (8.23). □

Věta 8.3. *Nechť $A \in \mathbb{R}^{n\times n}$, A je s.p.d. (symetrická, positivně definitní), viz Definice 1.1. Uvažujme Gaussovu-Seidelovu metodu, kde $G = -(D+L)^{-1}R$ je příslušná iterační matice, viz* (8.21). *Potom $\varrho(G) < 1$.*

Důkaz viz [3], Theorem 4.5, str. 129, nebo [1], Theorem 8.4, str. 241. □

Poznámka 8.2. (Spektrální charakterizace podmínky s.p.d.) *Lze ukázat, že matice $A \in \mathbb{R}^{n\times n}$ je s.p.d. podle Definice 1.1 právě když matice A je symetrická a všechna její vlastní čísla jsou kladná.*

Konvergenci iteračních metod lze urychlit zavedením parametru, tzv. relaxací. Uvažujme iterační metodu $x \longmapsto Gx+c$, viz (8.6). Nechť $\omega > 0$ je relaxační parametr. Pro zadané $x^{(k)} \in \mathbb{R}^n$ definujme "mezikrok"

$$\overline{x}^{(k+1)} \;=\; Gx^{(k)} + c\,.$$

Na základě $x^{(k)}$ a $\overline{x}^{(k+1)}$ definujme

$$x^{(k+1)} = x^{(k)} + \omega(\overline{x}^{(k+1)} - x^{(k)})\,.$$

Geometrická interpretace: V bodě $x^{(k)}$ definujeme polopřímku se směrovým vektorem $\overline{x}^{(k+1)} - x^{(k)}$, a parametrem $\omega > 0$. Pro zadané $\omega > 0$ definujeme novou aproximaci $x^{(k+1)}$.

Vyloučením $\overline{x}^{(k+1)}$ dostaneme iterační proces

$$x^{(k+1)} = ((1-\omega)I + \omega G)\, x^{(k)} + \omega c\,, \tag{8.24}$$

který závisí na volbě relaxačního parametru $\omega > 0$. Volbou $\omega = 1$ dostaneme původní iterační proces (8.9). Iterační matice relaxovaného procesu (8.24) je

$$G_\omega \equiv (1-\omega)I + \omega G = I - \omega B^{-1}A\,. \tag{8.25}$$

Proces (8.24) je ekvivalentní iteracím

$$Bx^{(k+1)} = (B - \omega A)x^{(k)} + \omega b\,. \tag{8.26}$$

V každém kroku řešíme soustavu lineárních rovnic s maticí B.

Každá z klasických iteračních metod, viz Definice 8.3–8.5, má svoji relaxovanou verzi. Shrňme to do jedné definice:

Definice 8.6. (Relaxované klasické iterační metody)

1. *Relaxovaná Richardsonova metoda (B=I):*

$$x^{(k+1)} = (I - \omega A)x^{(k)} + \omega b\,, \tag{8.27}$$

2. *Relaxovaná Jacobiova metoda (B=D):*

$$Dx^{(k+1)} = (D - \omega A)x^{(k)} + \omega b\,, \tag{8.28}$$

3. *Relaxovaná Gaussova-Seidelova metoda (B=D+L):*

$$(L + D)x^{(k+1)} = (L + D - \omega A)x^{(k)} + \omega b\,. \tag{8.29}$$

Příslušné relaxované iterační matice $G_\omega \in \mathbb{R}^{n\times n}$ lze snadno odvodit z (8.25).

Jaké jsou výhody relaxace? Uveďme několik tvrzení.

Lemma 8.1. *Nechť $A = A^{\mathrm{T}}$. Označme $\lambda_{\min}(A)$ resp. $\lambda_{\max}(A)$ nejmenší resp. největší vlastní číslo matice A. Uvažujme relaxovanou Richardsonovu metodu s iterační maticí $G_\omega = I - \omega A$, $\omega > 0$. O spektrálním poloměru $\varrho(G_\omega)$ iterační matice platí:*

$$\begin{aligned}\varrho(G_\omega) &\equiv \max_{\mu\in\sigma(G_\omega)} |\mu| \;= \\ &= \max_{\lambda_{\min}(A)\le\lambda\le\lambda_{\max}(A)} |1-\omega\lambda| = \max(|1-\omega\lambda_{\min}(A)|, |1-\omega\lambda_{\max}(A)|)\end{aligned} \tag{8.30}$$

Důkaz Postupujeme analogicky jako v důkazu Věty 8.2: Dokazovaná formule (8.30) je analogická formuli (8.23). □

Věta 8.4. *Nechť $A = A^{\mathrm{T}}$. Označme $\lambda_{\min}(A)$ resp. $\lambda_{\max}(A)$ nejmenší resp. největší vlastní číslo matice A. Uvažujme relaxovanou Richardsonovu metodu s iterační maticí $G_\omega = I - \omega A$, $\omega > 0$. Potom*

$$\varrho(G_\omega) < 1 \quad \Longleftrightarrow \quad 0 < \lambda_{\min}(A) \leq \lambda_{\max}(A) < \frac{2}{\omega}\,. \tag{8.31}$$

Důkaz Tvrzení (8.31) plyne z formule (8.30). □

Richardsonova metoda konverguje, pokud jsou splněny značně restriktivní požadavky na spektrum matice A, viz (8.22). Relaxací Richardsonovy metody požadavky na spektrum matice A významně oslabíme. Stačí předpokládat, že A je s.p.d., a tedy podle Poznámky 8.2, $0 < \lambda_{\min}(A)$. Jakkoliv velké bude $\lambda_{\max}(A)$, vždycky najdeme dostatečně malý relaxační parametr ω,

$$0 < \omega < \frac{2}{\lambda_{\max}(A)}\,, \tag{8.32}$$

že relaxovaná Richardsonova metoda bude konvergovat pro libovolnou volbu počáteční aproximace $x^{(0)}$, viz Věta 8.1 v aplikaci na spektrum matice G_ω.

Máme tedy možnost výběru parametru ω. Volba ω ovlivní rychlost konvergence iteračního procesu. V obecné analýze iteračních procesů jsme tomu věnovali Poznámku 3.3 (vliv faktoru kontrakce) a Poznámku 3.4 (lineární konvergence). Roli faktoru kontrakce přebírá spektální poloměr $\varrho(G_\omega)$: Uvažujme iterační proces (8.8), $x^{(k+1)} = g(x^{(k)})$, kde g odpovídá volbě relaxované Richarsonovy metody, (8.27). Potom

$$||x^{(k)} - x^*|| \leq (\varrho(G_\omega))^k \, ||x^{(0)} - x^*||\,, \quad k = 1, 2, \ldots \tag{8.33}$$

Přirozená strategie je, volit ω tak, aby odpovídající $\varrho(G_\omega)$ bylo co nejmenší.

Uvažujme funkci

$$\omega \;\longmapsto\; \varrho(G_\omega)\,, \tag{8.34}$$

která je definována na otevřeném intervalu (8.32). Zformulujeme optimalizační úlohu.

Problém 8.1.

$$\omega_{\mathrm{opt}} = \arg \min_{0<\omega<\frac{2}{\lambda_{\max}(A)}} \varrho(G_\omega) \tag{8.35}$$

Věta 8.5. *Nechť A je s.p.d. Označme $\lambda_{\min}(A)$ resp. $\lambda_{\max}(A)$ nejmenší resp. největší vlastní číslo matice A. Uvažujme relaxovanou Richardsonovu metodu s iterační maticí $G_\omega = I - \omega A$, $\omega > 0$. Potom existuje právě jediné řešení Problému 8.1, kde*

$$\omega_{\mathrm{opt}} = \frac{2}{\lambda_{\max}(A) + \lambda_{\min}(A)}\,. \tag{8.36}$$

Spektrální poloměr příslušné iterační matice je

$$\varrho(G_{\text{opt}}) = \frac{\lambda_{\max}(A) - \lambda_{\min}(A)}{\lambda_{\max}(A) + \lambda_{\min}(A)}\,. \tag{8.37}$$

Důkaz viz [3], Theorem 4.9, str. 137, resp. [1], Example 8.9, str. 243. □

Na závěr uvedeme stručný souhrn faktů, které se týkají relaxované Gaussovy-Seidelovy metody.

Věta 8.6. *Nechť A je s.p.d. Uvažujme relaxovanou Gaussovu-Seidelovu metodu. Nechť G_ω je příslušná iterační matice. Potom $\varrho(G_\omega) < 1$ právě když $0 < \omega < 2$.*

Poznamenejme, že i když matice A je s.p.d., iterační matice G_ω nemusí být symetrická, což ztěžuje analýzu.
Důkaz viz [1], Property 4.3, str. 131. □

- Řeší se úloha optimálního výběru relaxačního parametru. Lze ukázat, že

$$1 < \omega_{\text{opt}} < 2\,. \tag{8.38}$$

- Pro speciální třídy matic existují formule pro ω_{opt}, viz [1], Property 4.4, str. 131.
- Pro obecné matice A s.p.d. lze ω_{opt} experimentálně zjistit (dalším iteračním procesem) .

Poznámka 8.3. (Relaxovaná Gaussova-Seidelova metoda = SOR) *Tato metoda je známa pod zkratkou SOR-metoda. Zkratka znamená* successive overrelaxation. *Pracuje v režimu, kdy $\omega > 1$ t.j., kdy je "overrelaxed", viz* (8.38).

Metody uvedené v tomto odstavci vznikly v šedesátých letech minulého století, viz např. [10]. Bylo to v době, kdy vznikla potřeba řešit rozsáhlé úlohy matematické fyziky, např. reaktorové fyziky. V této souvislosti bylo třeba řešit velké soustavy lineárnách rovnic. Z hlediska aplikací byl předpoklad "A je s.p.d." přirozený. Kromě praktické nezbytnosti, vznikla potřeba kultivovat teorii těchto metod. Dnes jsou to už tedy "klasické iterační metody". Nicméně periodicky se vracejí do módy. Naposledy v souvislosti s technikou více sítí (tzv. multigrid).

8.2 Metoda sdružených gradientů

Úlohu (8.1) budeme formulovat jako úlohu minimalizace reálné funkce více proměnných,

$$x \in \mathbb{R}^n \longmapsto f(x) \in \mathbb{R}^1\,,$$

viz Problém 4.1. Pro numerické řešení použijeme metody sestupu, viz Odstavec 4.2, zejména metodu sdružených gradientů, viz Algoritmus 4.6.

Příslušná funkce f souvisí s daty úlohy (8.1), t.j. maticí $A \in \mathbb{R}^{n\times n}$ a pravou stranou $b \in \mathbb{R}^n$:

Definice 8.7. *Pro zadanou dvojici $A \in \mathbb{R}^{n\times n}$ a $b \in \mathbb{R}^n$ definujeme funkci*

$$x \in \mathbb{R}^n \longmapsto f(x) \equiv \frac{1}{2} x^{\mathrm{T}} A x - b^{\mathrm{T}} x \ \in \mathbb{R}^1 \,. \tag{8.39}$$

Funkce f je kvadratický funkcionál.

Souvislost úlohy (8.1) s minimalizační úlohou funguje pouze za předpokladu, že matice soustavy $A \in \mathbb{R}^{n\times n}$ je s.p.d. (symetrická, pozitivně definitní), viz Poznámka 8.2.

Připomeneme pojem gradient $\nabla f(x)$ funkce $f = f(x)$ v bodě $x \in \mathbb{R}^n$, viz Definice 4.6:

$$\nabla f(x) \equiv (\frac{\partial f}{\partial x_1}(x), \ldots, \frac{\partial f}{\partial x_n}(x))^{\mathrm{T}} \in \mathbb{R}^n \,.$$

Lemma 8.2. *Uvažujme kvadratický funkcionál* (8.39)*. Potom*

$$\nabla f(x) = Ax - b \,. \tag{8.40}$$

Důkaz Je třeba vypočítat složky parciálních derivací $\frac{\partial}{\partial x_k}\left(\frac{1}{2}\, x^{\mathrm{T}} A x - b^{\mathrm{T}} x\right)$ pro $k = 1, \ldots, n$. Klíčovým výpočtem je kalkulace $\frac{\partial}{\partial x_k}\left(x^{\mathrm{T}} A x\right)$:

$$\frac{\partial}{\partial x_k}\left(x^{\mathrm{T}} A x\right) = \frac{\partial}{\partial x_k} \sum_{i,j=1}^{n} a_{ij} x_i x_j = \sum_{i,j=1}^{n} a_{ij} \frac{\partial}{\partial x_k}(x_i x_j) =$$

$$= \sum_{i,j=1}^{n} a_{ij}(\delta_{ki} x_j + x_i \delta_{kj}) = \sum_{j=1}^{n} a_{kj} x_j + \sum_{i=1}^{n} \underbrace{a_{ik}}_{= a_{ki}} x_i = 2 \sum_{j=1}^{n} a_{kj} x_j \,,$$

přičemž v předposledním kroku je využito symetrie $a_{jk} = a_{kj}$ matice. Dojdeme k závěru, že

$$\frac{\partial}{\partial x_k}\left(\frac{1}{2}\, x^{\mathrm{T}} A x - b^{\mathrm{T}} x\right) = \sum_{j=1}^{n} a_{kj} x_j - b_k \,, \quad k = 1, \ldots, n \,.$$

□

Věta 8.7. *Nechť A je s.p.d., $b \in \mathbb{R}^n$. Potom*

$$Ax^* = b \quad \Longleftrightarrow \quad x^* = \arg\min_{x\in\mathbb{R}^n} \ f(x) \,, \quad f(x) \equiv \frac{1}{2} x^{\mathrm{T}} A x - b^{\mathrm{T}} x \,. \tag{8.41}$$

Důkaz 1) Nechť $Ax^* = b$. Analyzujme funkci

$$x \;\mapsto\; h(x) \equiv (x - x^*)^{\mathrm{T}} A(x - x^*)\,. \tag{8.42}$$

Budeme ji postupně upravovat:

$$h(x) = (x - x^*)^{\mathrm{T}} A(x - x^*) = x^{\mathrm{T}}Ax - \underbrace{x^{\mathrm{T}}Ax^*} - (x^*)^{\mathrm{T}}Ax + (x^*)^{\mathrm{T}}Ax^* \;\;.$$

Protože A je symetrická, $x^{\mathrm{T}}Ax^* = (x^*)^{\mathrm{T}}A^{\mathrm{T}}x = (x^*)^{\mathrm{T}}Ax$. Potom

$$\begin{aligned} h(x) &= x^{\mathrm{T}}A^{\mathrm{T}}x - 2(x^*)^{\mathrm{T}}Ax + (x^*)^{\mathrm{T}}Ax^*\,, \\ &= x^{\mathrm{T}}A^{\mathrm{T}}x - 2b^{\mathrm{T}}x + (x^*)^{\mathrm{T}}Ax^*\,, \\ &= 2\left(\tfrac{1}{2}x^{\mathrm{T}}Ax - b^{\mathrm{T}}x\right) + (x^*)^{\mathrm{T}}Ax^*\,, \\ &= 2f(x) + (x^*)^{\mathrm{T}}Ax^*\,. \end{aligned}$$

Ve funkci $h = h(x)$ zaveďme substituci $y = x - x^*$. Transformovaná funkce

$$y \;\mapsto\; y^{\mathrm{T}}Ay \tag{8.43}$$

má jediné minimum

$$0 = \arg\min_{y\in\mathbb{R}^n}\; y^{\mathrm{T}}Ay\,,$$

protože A je s.p.d., viz Definice 1.1. V původních souřadnicích,

$$x^* = \arg\min_{x\in\mathbb{R}^n}(x - x^*)^{\mathrm{T}}A(x - x^*) = \arg\min_{x\in\mathbb{R}^n}\left(2f(x) + (x^*)^{\mathrm{T}}Ax^*\right)\,.$$

Pozice x^* minima funkce $x \mapsto 2f(x) + (x^*)^{\mathrm{T}}Ax^*$ je stejná jako pozice minima funkce $x \mapsto f(x)$. Proto

$$x^* = \arg\min_{x\in\mathbb{R}^n} f(x) = \arg\min_{x\in\mathbb{R}^n}\left(\frac{1}{2}x^{\mathrm{T}}Ax - b^{\mathrm{T}}x\right)\,.$$

2. Nechť $x^* = \arg\min\limits_{x\in\mathbb{R}^n}\left(\dfrac{1}{2}x^{\mathrm{T}}Ax - b^{\mathrm{T}}x\right)$. Nutnou podmínkou existence minima je splnění rovnice $\nabla f(x^*) = 0$. Z Lemmatu 8.2 vyplývá, že $Ax^* = b$.

□

Aplikujeme metodu sestupu t.j. Algoritmus 4.2. Ze speciálního tvaru kvadratického funkcionálu (8.39) vyplývá řada zjednodušení.

Lemma 8.3. (Přípustný směr sestupu) *Uvažujme kvadratický funkcionál* (8.39). *Nechť* $x \in \mathbb{R}^n$. *Nechť* $d \in \mathbb{R}^n$ *je zadaný směr sestupu. Potom* d *je přípustným směrem sestupu právě když*

$$(Ax - b)^{\mathrm{T}}d < 0\,. \tag{8.44}$$

Důkaz Podmínka přípustnosti (4.7) z Definice 4.3 znamená, že $\sum_{i=1}^{n} \frac{\partial f}{\partial x_i}(x) d_i < 0$ t.j., $(\nabla f(x))^{\mathrm{T}} d < 0$. Protože $\nabla f(x) = Ax - b$, potom podmínka (4.7) platí právě když $(Ax - b)^{\mathrm{T}} d < 0$. □

Lemma 8.4. (Line-search) *Uvažujme kvadratický funkcionál* (8.39)*, t.j.* $x \in \mathbb{R}^n \mapsto f(x) \equiv \frac{1}{2} x^{\mathrm{T}} A x - b^{\mathrm{T}} x \in \mathbb{R}^1$*. Nechť* $x \in \mathbb{R}^n$*. Nechť* $d \in \mathbb{R}^n$ *je přípustný směr sestupu. Definujme reálnou funkci* $\alpha \in \mathbb{R}^1 \mapsto g(\alpha) \equiv f(x + \alpha d) \in \mathbb{R}^1$*. Potom existuje právě jediné* $\alpha_{\min} > 0$ *tak, že*

$$\alpha_{\min} = \arg \min_{\alpha \in \mathbb{R}^1} g(\alpha) = \frac{r^{\mathrm{T}} d}{d^{\mathrm{T}} A d}, \quad r = b - Ax. \tag{8.45}$$

Komentář: Funkce g je je kvadratická funkce. Je konvexní díky předpokladu přípustnosti směru d. Pozici minima $\alpha_{\min}$ lze vypočítat z jednoduché formule.
Důkaz Z definice funkce g,

$$g(\alpha) = f(x + \alpha d) = \frac{1}{2}(x + \alpha d)^{\mathrm{T}} A (x + \alpha d) - b^{\mathrm{T}}(x + \alpha d) =$$

$$= \frac{1}{2}\, \alpha^2\, d^{\mathrm{T}} A d + \alpha \left(x^{\mathrm{T}} A d - b^{\mathrm{T}} d \right) + \frac{1}{2} x^{\mathrm{T}} A x - d^{\mathrm{T}} x .$$

Nutná podmínka pro extrém funkce g:

$$\frac{\mathrm{d}}{\mathrm{d}t} g(\alpha) = \alpha\, d^{\mathrm{T}} A d + x^{\mathrm{T}} A d - b^{\mathrm{T}} d = \alpha\, d^{\mathrm{T}} A d + (Ax - b)^{\mathrm{T}} d = 0 .$$

Protože $(Ax - b)^{\mathrm{T}} d < 0$, extrém je minimum. □

Jeden krok metody sestupu, viz (4.8), t.j.

$$x^{(k+1)} = x^{(k)} + \alpha_k d^{(k)} \tag{8.46}$$

je dán explicitní formulí, kde

$$\alpha_k = \frac{(r^{(k)})^{\mathrm{T}} d^{(k)}}{(d^{(k)})^{\mathrm{T}} A d^{(k)}}, \quad r^{(k)} = b - A x^{(k)},$$

nebo, použitím skalárního součinu,

$$\alpha_k = \frac{\left(r^{(k)}, d^{(k)}\right)}{\left(d^{(k)}, A d^{(k)}\right)}, \quad r^{(k)} = b - A x^{(k)} . \tag{8.47}$$

Krok "line-search" nemusíme tedy řešit numericky pomocí metod z Odstavce 4.2.1.

Poznámka 8.4. *Připomeneme, co se rozumí skalárním součinem ve formuli* (8.47)*. Pro každou dvojici* $x \in \mathbb{R}^n$*,* $y \in \mathbb{R}^n$ *definujme*

$$(x, y) \equiv x^{\mathrm{T}} y .$$

Euklidovská norma $||x||$ vektoru $x \in \mathbb{R}^n$ je generována uvedeným skalárním součinem: $||x||^2 = (x,x)$. Říkáme, že vektory $x \neq 0$ a $y \neq 0$ jsou kolmé, jestliže $(x,y) = 0$.

Uvádíme tato známá fakta jenom proto, že později zavedeme jiný skalární součin, který bude znamenat jinou definici kolmosti dvou vektorů.

Algoritmy metody největšího spádu (Algoritmus 4.5) a metody sdružených gradientů (Algoritmus 4.6) z Odstavce 4.2.2 přeformulujeme. Využijeme vyjádření explicitního vyjádření pro největší spád $-\nabla f(x^{(k)}) = b - Ax^{(k)}$, a zavedeme novou proměnnou $r^{(k)} = b - Ax^{(k)}$. Mnemotechnicky, $r^{(k)}$ je residuum, t.j. zbytek při řešení soustavy $b - Ax^{(k)}$.

Algoritmus 8.1. (Metoda největšího spádu) *Dáno: $x^{(0)} \in \mathbb{R}^n$, $r^{(0)} = b - Ax^{(0)} \in \mathbb{R}^n$, $d^{(0)} = r^{(0)} \in \mathbb{R}^n$. Definujeme posloupnosti $\left\{x^{(k)}\right\}_{k=0}^{\infty}$, $\left\{r^{(k)}\right\}_{k=0}^{\infty}$ a $\left\{d^{(k)}\right\}_{k=0}^{\infty}$ rekurencí*

- $x^{(k+1)} = x^{(k)} + \alpha_k d^{(k)}\,, \quad \alpha_k = \dfrac{\left(r^{(k)}, d^{(k)}\right)}{\left(d^{(k)}, Ad^{(k)}\right)}\,,$
- $r^{(k+1)} = b - Ax^{(k+1)}\,,$
- $d^{(k+1)} = r^{(k+1)}\,.$

Algoritmus 8.2. (Metoda sdružených gradienů) *Dáno: $x^{(0)} \in \mathbb{R}^n$, $r^{(0)} = b - Ax^{(0)} \in \mathbb{R}^n$, $d^{(0)} = r^{(0)} \in \mathbb{R}^n$. Definujeme posloupnosti $\left\{x^{(k)}\right\}_{k=0}^{\infty}$, $\left\{r^{(k)}\right\}_{k=0}^{\infty}$ a $\left\{d^{(k)}\right\}_{k=0}^{\infty}$ rekurencí*

- $x^{(k+1)} = x^{(k)} + \alpha_k d^{(k)}\,, \quad \alpha_k = \dfrac{\left(r^{(k)}, d^{(k)}\right)}{\left(d^{(k)}, Ad^{(k)}\right)}\,,$
- $r^{(k+1)} = b - Ax^{(k+1)}\,,$
- $d^{(k+1)} = r^{(k+1)} + \beta_k d^{(k)}\,, \quad \beta_k = \dfrac{\left(r^{(k+1)}, r^{(k+1)}\right)}{\left(r^{(k)}, r^{(k)}\right)}\,.$

Dlužíme motivaci pro definici parametru β_k v Algoritmu 8.2, t.j. parametru γ z formule (4.25). Je dobré mít na mysli Obrázek 4.18. Parametr γ se s iteračním krokem mění. Správně by měl být označen jako γ_k a tedy $\beta_k = \gamma_k$.

Definice 8.8. (A-kolmost) *Nechť A je s.p.d. Pro každou dvojici $x \in \mathbb{R}^n$, $y \in \mathbb{R}^n$ definujme*

$$(x,y)_A \equiv x^{\mathrm{T}} A y\,.$$

Říkáme, že vektory $x \neq 0$, $y \neq 0$ jsou A-kolmé, jestliže $(x,y)_A = 0$.

Lze ukázat, že zobrazení $x \in \mathbb{R}^n, y \in \mathbb{R}^n \mapsto (x,y)_A \in \mathbb{R}^1$ definuje skalární součin na $\mathbb{R}^n$. Říkáme mu energetický skalární součin. Normě $||x||_A \equiv \sqrt{(x,y)_A}$, generované energetickým skalárním součinem se říká energerická norma.

Výběr parametru β_k je diktován požadavkem, aby $(d^{(k)}, d^{(k+1)})_A = 0$: Definujme $d^{(k+1)} = r^{(k+1)} + \beta_k d^{(k)}$, kde β_k je dosud neurčený parametr. Požadujeme, aby

$$0 = (d^{(k)}, d^{(k+1)})_A = (d^{(k)}, r^{(k+1)})_A + \beta_k (d^{(k)}, d^{(k)})_A \,.$$

Potom

$$\beta_k = -\frac{\left(d^{(k)}, r^{(k+1)}\right)_A}{(d^{(k)}, d^{(k)})_A} = \frac{\left(r^{(k+1)}, r^{(k+1)}\right)}{(r^{(k)}, r^{(k)})} \,,$$

přičemž dobrat se k poslední identitě představuje delší počítání. Lze ukázat, že

$$x^* - x^{(0)} = \sum_{k=0}^{n-1} \alpha_k d^{(k)} \,, \quad x^* = x^{(n)} \,.$$

K nalezení řešení x^* je třeba nevýše n kroků algoritmu.

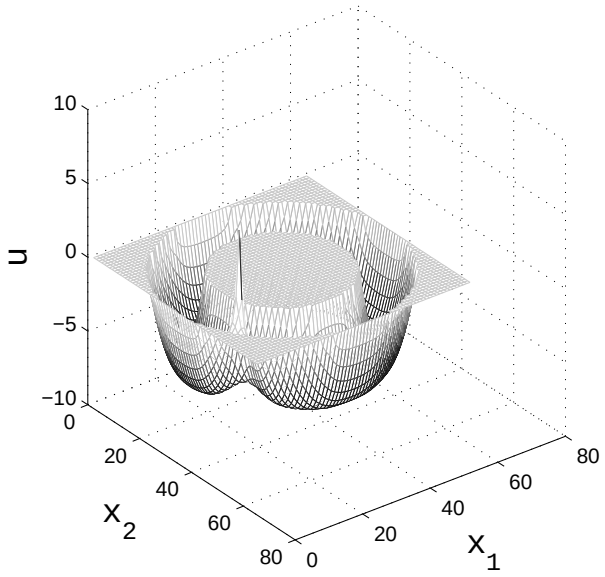
Algoritmus 8.2 tedy vypadá, jako finitní metoda s pevně vyčíslitelnými náklady (#flops $n^3 + O(n^2)$), o trochu dražší než Gaussova eliminace. Tak se algoritmus jeví pro nízký počet rovnic (malá n), viz Příklad 4.1 a Obrázek 4.19. Nicméně, metoda sdružených gradientů je velmi efektivní jako iterační metoda pro velkou třídu problémů.

8.3 Příklad

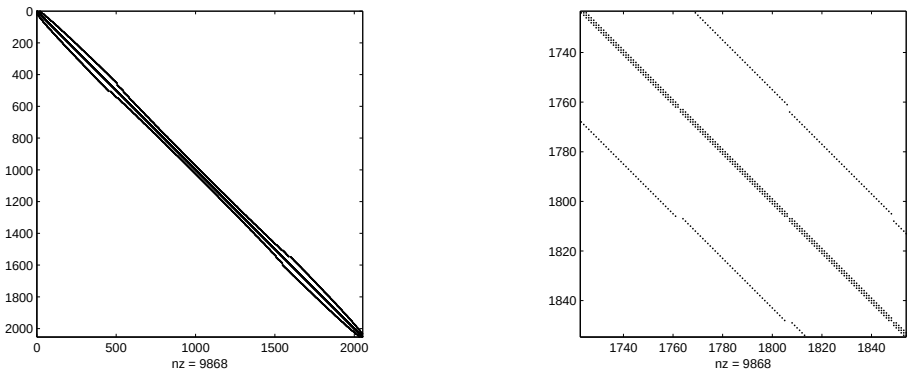
Budeme uvažovat model elastické deformace membrány (např. z telefonního sluchátka) ve tvaru mezikruží. Na membránu působí konstantní objemové síly a velká bodová síla opačné orientace. Na Obrázku 8.1 je numerické řešení problému jako funkce $u = u(x_1, x_2)$ výchylky v bodě (x_1, x_2) z mezikruží. Numerické řešení je založeno na diskretizaci původně spojitého modelu. Problém lze ekvivalentně formulovat jako řešení soustavy lineárnách rovnic $Ax = b$, kde $A \in \mathbb{R}^{2052 \times 2052}$ je symetrická, $b \in \mathbb{R}^{2052}$ a $x \in \mathbb{R}^{2052}$ je řešení soustavy.

Matice A má naprostou většinu prvků nulových. Pouze asi 0.25 procent prvků matice A jsou nenulové prvky. Na Obrázku 8.2 je znázorněno zaplnění matice (angl. sparsity pattern): Každý tmavý pixel odpovídá pozici nenulového prvku matice. Říkáme, že matice A je řídká (angl. sparse).

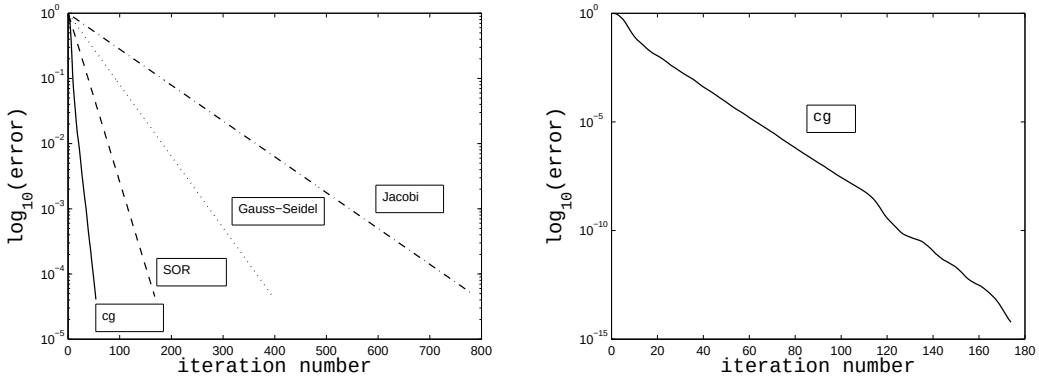
Poznámka 8.5. (Řídká matice) *Definice řídké matice vlastně neexistuje. Bylo by samozřejmě možné limitovat (v procentech) zaplnění matice. Co je ale možné definovat, je formát zadání matice: Budeme požadovat, aby matice byla zadána pouze výčtem pozic a hodnot nenulových prvků. V systému* `MATLAB` *to lze udělat prostřednictvím maticového formátu* `sparse`*. V tomto formátu lze učinit dotaz, kolik má matice nenulových prvků.*



Obrázek 8.1: Průhyb membrány $u = u(x_1, x_2)$ ve tvaru mezikruží, pod vlivem bodové síly.



Obrázek 8.2: Matice soustavy $A \in \mathbb{R}^{2052\times 2052}$: zaplnění matice (angl. sparsity pattern), počet nenulových prvků nz=9868. Vpravo: detail zaplnění.



Obrázek 8.3: Vpravo, srovnání konvergence čtyř iteračních metod, tol $= 10^{-4}$. Vlevo, konvergence metody cg, tol $= 10\,\mathrm{eps}\,||b||$.

Řídké matice vznikají např. při numerickém řešení problémů, které jsou popsány pomocí parciálních diferenciálních rovnic, viz model membrány. Příkladem skutečně obrovské řídké matice je tzv. Google-matrix,

`http://en.wikipedia.org/wiki/Google_matrix`

Budeme experimenálně zkoumat konvergenci

1. Jacobiovy metody (Definice 8.4)
2. Gaussovy-Seidelovy metody (Definice 8.18)
3. metody SOR (Poznámka 8.3, s volbou $\omega = 1.4$.)
4. metody cg, t.j. metody sdružených gradienů, conjugate gradients (Algoritmus 8.1)

na příkladu deformace elastické membrány na Obrázku 8.1.

Příslušné iterační procesy startovaly z počáteční aproximace $x^{(0)} = 0 \in \mathbb{R}^{2052}$. Jako zakončovací kriterium bylo použito vyčíslení normy rezidua, viz podmínka (8.3), s volbou tol $= 10^{-4}$. Pro odhad přesnosti výpočtu potřebujeme znát pevný bod $x^* \in \mathbb{R}^{2052}$ příslušného iteračního procesu. Ten neznáme, nicméně víme, že x^* je řešením soustavy $Ax^* = b$. Proto definujeme numerické řešení soustavy lineárních rovnic $Ax^* = b$ přímou metodou (viz `MATLAB` , A backslash b) jako referenční x^*. Řešení z Obrázku 8.1 je, v příslušně konvertovaném formátu, referenční vektor x^*. V každém kroku $x^{(k)}$ iteračního procesu měříme relativní chybu

$$\text{error} = \frac{||x^{(k)} - x^*||}{||x^{(k)}||}\,.$$

Na Obrázku 8.3 vlevo jsou shrnuty výsledky všech čtyř experimentů. Numerické testy odpovídají na otázku, kolika iterací je potřeba, abychom získali řešení s přesností na čtyři platné cifry. Zároveň se (v tomto konkrétním experimentu) ukazuje, že zadaná tolerance tol odpovídá v podstatě získané přesnosti. Cena jedné iterace pro každou z uvedených metod je zhruba stejná. Proto metoda cg vychází ze srovnání jako nejlepší. Na Obrázku 8.3 vpravo je analyzována konvergence metody cg, kde $\mathrm{tol} = 10\,\mathrm{eps}\,||b||$, přičemž eps je strojová přesnost, $\mathrm{tol} \sim 2.220446049250313e-016$. Iterace sledujeme až do (rozumného) konce. Metoda cg potřebovala asi 180 iterací, aby dosáhla maxima svých možností. Srovnejme to s nesmyslným nápadem, použít cg jako finitní metodu, která potřebuje 2052 kroků.

Znovu je třeba zopakovat varování z preambule kapitoly: spolehlivá zastavovací kriteria jsou předmětem aktuálního výzkumu, viz [5], 8.6 Zastavovací kriteria, str. 203.

Literatura

[1] Deuflhard P. and Hohmann A., *Introduction to Scientific Computing*, the 2nd edition, Springer, 2002

[2] Segethová J., *Základy numerické matematiky*, MFF UK, 2002

[3] Quarteroni A., Sacco R. and Saleri F., *Numerical Mathematics*, Springer, 2000

[4] Golub G.H., van Loan Ch.F., *Matrix computations*, The Johns Hopkins University Press, Third Edition, 1996

[5] Duintjer Tebbens J., Hnětynková I., Plešinger M., Strakoš Z., Tichý P., *Analýza metod pro maticové výpočty. Základní metody*, MatfyzPress, 2012

[6] Press W.H., Flannery B.P., Teukolsky S.A., Vetterling W.T., *Numerical recepies in C, The Art of Scientific Computing*, Cambridge Uviv. Press, 1989

[7] Kurzweil J., *Obyčejné diferenciální rovnice*, SNTL, Praha, 1978

[8] Deuflhart P., Bornemann F., *Scientific Computing with Ordinary Differential Equations*, Springer-Verlag, Texts in Applied Mathematis 42, New York, 2002

[9] Hairer E., Norset S.P., Wanner G., *Solving Ordinary Differential Equations I (Nonstiff Problems)*, Springer Verlag, second revised edition, New York, 1993

[10] Varga R., *Matrix Iterative Analysis*, Prentice-Hall, Englewood Cliffs, New York, 1962