I. Základní pojmy numerické matematiky

1 Numerické úlohy a algoritmy

Technik, fyzik, ekonom, případně pracovník i jiných oborů se o matematiku zajímá především se dvou důvodů. Na jedné straně je pro něho matematika prostředkem k formulaci problémů a ke zkoumání vztahů mezi vyšetřovanými objekty, tedy jakýmsi jazykem, a na druhé straně je prostředkem k řešení formulovaných problémů, tedy metodou.

Uvědomme si, jak se takový reálný problém řeší. Bud' jej můžeme řešit prostředky daného oboru, tj. například pomocí experimentů, měření atd., nebo prostředky matematickými. K matematickému řešení reálného problému je nutné nejdříve formulovat matematický model daného problému, tj. formulovat *matematickou úlohu.*

Takovou úlohu chápeme jako funkční vztah mezi danými a hledanými objekty[1]) a pochopitelně požadujeme, aby tento vztah byl srozumitelný a jednoznačný.

1.1 Numerické úlohy a numerické metody.

Postupy (metody), jak na základě daných objektů a pomocí vztahů charaktererizujících danou úlohu stanovit objekty hledané, mohou být velmi různorodé. V této souvislosti hovoříme o e x a c t n í c h (teoretických), p ř i b l i ž n ý c h a n u m e r i c k ý c h m e t o d á c h.

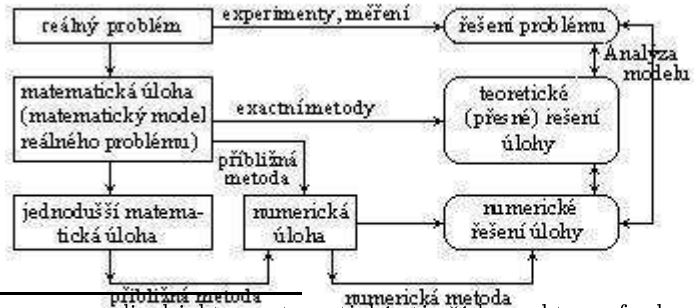


Schéma 1

[1]) Máme na mysli objekty matematické, tj. čísla, vektory, funkce, matice, operátory, křívky, plochy,... atd., obecně prvky nějakých funkcionálních prostorů.

Každá metoda nám danou úlohu převádí na úlohu (nebo systém úloh) jednodušší. Jednou z nich je úloha numerická.

Numerickou úlohou rozumíme jasný a jednoznačný popis funkčního vztahu mezi k o n e č n ý m počtem *vstupních* a *výstupních dat,* tj. mezi danými a hledanými objekty numerické úlohy. Zdůrazněme, že pro numerické úlohy je požadavek konečnosti souborů vstupních a výstupních dat podstatný. Data numerické úlohy musí být vyjádřitelná konečným počtem (reálných) čísel.

Numerická úloha je tedy takovým matematickým modelem reálného problému, který může být realizován na počítači (v konečném čase).

Základní matematickou disciplínou, která konstruuje a analyzuje metody a postupy pro realizaci numerických úloh na počítačích, je *numerická matematika.*

Struktura zkoumání reálných problémů je patrná ze schématu 1 na str. ??

1.2 Příklady

1.2.1

Úloha určit kořeny rovnice $x^3+a_1x^2+a_2x+a_3=0$ s reálnými koeficienty je numerická úloha. Vektor vstupních dat je $\mathbf{a}=(a_1,a_2,a_3)$. Výstupní data reprezentovaná vektorem $\mathbf{x}=(\alpha_1,\beta_1,\alpha_2,\beta_2,\alpha_3,\beta_3,)$ určují hledané (komplexní) kořeny $x_k=\alpha_k+\mathbf{i}\beta_k$, $k=1,2,3$. Vztah mezi vstupními a výstupními daty je rovnicí dán implicitně.

Danou úlohu můžeme řešit různými metodami. Např. můžeme užít metodu výpočtů podle Cardanových formulí nebo některou z metod uvedených v kap.III této publikace.

1.2.2

Matematická úloha najít řešení $y=y(x)$ diferenciální rovnice

$$y''=x^2+y^2,$$

vyhovující počátečním podmínkám $y(0)=0$, $y'(0)=1$, není numerickou úlohou, nebot' hledaná funkce y nemůže být určena konečným počtem reálných čísel. Tato matematická úloha může být prostřednictvím nějaké přibližné metody (viz [11],[12]) aproximována (tj. přibližně nahrazena) numerickou úlohou, jestliže specifikujeme výstupní údaje jako přiblížné hodnoty funkce y v bodech $x=h,2h,3h,...,Nh$. Vstupními údaji jsou čísla

$y(0) = 0$, $y'(0) = 1$, dále pak hodnoty funkce x^2 v bodech $x = h, 2h, ..., Nh$ a čísla h a N .

1.3 Algoritmy.

Algoritmem[2]) *numerické metody* rozumíme jasnou a jednoznačnou specifikaci (popis) konečné posloupnosti operací, jejichž prostřednictvím se m-tici čísel z určité množiny vstupních dat jednoznačně přiřazuje n-tici výsledků. "Operacemi" rozumíme aritmetické a logické operace, které může provádět počítač. Cíle numerické matematiky jsou patrně dvojího typu:

a) převod matematické úlohy na numerickou, případně převod numerické úlohy na jednodušší numercikou úlohu,

b) udání algoritmů pro řešení numerických úloh a vyšetřování vlastností těchto algoritmů.

Termínem *numerický algoritmus*, resp. *numerický proces* zdůrazňujeme, že nás u algoritmu zajímá pouze realizace aritmetických operací s čísly (nikoliv logické operace).

Jsou-li $(x_1, x_2, ..., x_m) = \mathbf{x}$ vstupní data, potom numerický algoritmus je dán popisem operací $r_1, r_2, ..., r_k$ a jejich výsledků

$$
\begin{aligned}
& z_1 = r_1(\mathbf{x}), \\
& z_2 = r_2(\mathbf{x}, z_1), \\
& z_3 = r_3(\mathbf{x}, z_1, z_2), \\
& \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots, \\
& z_k = r_k(\mathbf{x}, z_1, z_2, ..., z_{k-1}) .
\end{aligned}
\tag{1.3.1}
$$

Konečný výsledek $\mathbf{y} = (y_1, y_2, ..., y_n)$, $1 \leqq n \leqq k$, je dán speciálním výběrem mezivýsledků z_i .

Algoritmus (1.3.1) budeme stručně zapisovat takto:

$$
\mathbf{y} = \mathbf{r}(\mathbf{x}), \quad \mathbf{x} \in X, \quad \mathbf{y} \in Y,
\tag{1.3.2}
$$

kde X a Y jsou množiny (prostory) vstupních a výstupních dat.

V počítači se však aritmitické operace nerealizují přesně (viz čl. 2), proto místo mezivýsledků $\mathbf{z}$ dostáváme mezivýsledky $\tilde{\mathbf{z}}$; navíc při zavádění vstup-

[2]) V 9. století n.l. arabský matematik Muhamad ibn Músá al-Chvárizmí (pocházel z Chívy) napsal knihu, ve které vykládá indický početní systém (základ dekadického poziičního systému) a pravidla jeho používání. Latinskému překladu názvu knihy (včetně autorova jména) "Algorithmi de numero Indorum" vděčíme za termín algoritmus.

ních dat do počítače se zpravidla dopouštíme jistých vstupních chyb, takže algoritmus realizovaný počítačem, tzv. *strojový algoritmus*, dává mezivýsledky

$$\begin{aligned}
&\tilde{z_1} = \varrho_1\left(\tilde{\mathbf{x}}\right), \\
&\tilde{z_2} = \varrho_2\left(\tilde{\mathbf{x}}, \tilde{z_1}\right), \\
&\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots, \\
&\tilde{z_k} = \varrho_k\left(\tilde{\mathbf{x}}, \tilde{z_1}, \tilde{z_2}, ..., \tilde{z}_{k-1}\right) .
\end{aligned}$$

a strojový výsledek

$$\tilde{\mathbf{y}} = \varrho\left(\tilde{\mathbf{x}}\right), \tag{1.3.3}$$

který se může dost podtatně lišit od výsledku $\mathbf{y}$ (teoretického). Strojové operace $\varrho_1, \varrho_2, ..., \varrho_k$ se v počítači reakizují podle pravidel daných konstrukcí počítače. Několik nezbytných informací najde čtenář v odst. 2.3.

1.3.1 Příklad.

Mějme úlohu stanovit kořeny x_1, x_2 kvadratické rovnice $x^2 - 2bx + c = 0$ s reálnými koeficienty, když $b^2 - c > 0$.

Metodou doplnění na čtverec dostaneme

$$x_1 = b + \sqrt{(b_2 - c)}, \qquad x_2 = b - \sqrt{(b_2 - c)}$$

Algoritmus A1:

$$\begin{aligned}
z_1 &= b^2, \\
z_2 &= z_1 - c, \\
z_3 &= \sqrt{z_2}, \\
z_4 &= b + z_3, \\
\underline{z_5} &= \underline{c/z_4,} \\
x_1 &= z_4, \\
x_2 &= z_5 .
\end{aligned}$$

Algoritmus A2:

$$\begin{aligned}
z_1 &= b^2, \\
z_2 &= z_1 - c, \\
z_3 &= \sqrt{z_2}, \\
z_4 &= b - z_3, \\
\underline{z_5} &= \underline{b + z_3,} \\
x_1 &= z_4, \\
x_2 &= z_5 .
\end{aligned}$$

Algoritmus A1 pro výpočet kořene x_2 vychází ze vzorce

$$x_2 = \frac{c}{b + \sqrt{(b^2 - c)}}.$$

Později (v odst. 4.3) uvidíme, že i když jsou algoritmy A1, A2 pro určení kořene x_2 teoreticky zcela ekvivalentní, mohou pro různá vstupní data b, c dávat na počítači zcela rozdílné výsledky. Pro výpočet x_1 se tyto algoritmy neliší.

1.4 Příklady algoritmů.

1.4.1 Hornerův algoritmus.

Chceme stanovit $p(\alpha)$, kde α je dané číslo a p daný polynom stupně n:

$$p(x) = a_0 x^n + a_1 x^{n-1} + ... + a_{n-1} x + a_n, \qquad a_0 \neq 0.$$

Tento polynom můžeme přepsat na ekvivalentní tvar

$$p(x) = \underbrace{((...}_{n-1}(a_0 x + a_1)x + a_2)x + ... + a_{n-1})x + a_n.$$

Algoritmus výpočtu $p(\alpha)$ bude dán postupným vypočítáváním členů v jednotlivých závorkách pro $x = \alpha$. Položíme-li $b_0 = a_0$, vidíme, že jde o opakované vyčíslování výrazů typu

$$b_i = \alpha b_{i-1} + a_i, \qquad i = 1, 2, ..., n, \tag{1.4.1}$$

kde hodnota vypočtená na levé straně se v dalším kroku vyskytne na straně pravé. Jde o algoritmus typu (1.3.1), kde $b_1, b_2, ..., b_{n-1}$ jsou mezivýsledky a $b_n = p(\alpha)$ je výsledek (výstupní údaj), který nás zajímá. Operace r_i z (1.3.1) je zde součin čísla α a předchozího výsledku a přičtení čísla a_i. Vidíme, že tento algoritmus obsahuje n násobení a n sčítání, a tedy celkem $2n$ operací[3]).

Výraz typu (1.4.1) nazýváme *rekurentním vztahem (rekurencí)* a uvedený výpočet *rekurentním procesem.*

Výpočet podle vztahu (1.4.1) se dá zapsat do tabulky, které se říká *Hornerovo schéma*:

[3]Kdybychom výpočet čísla $p(\alpha)$ prováděli "přirozeným" způsobem. tj. výpočtem jednotlivých členů typu $a_k x^{n-k}$ a sečtením, potřebovali bychom obecně $n + (n-1) + (n-2) + ... + 2 + 1$ násobení a n sčítání, tj. celkem $\left(n^2 + 3n\right)/2$ operací a výpočet by byl pro větší n podstatně pomalejší, nehledě na to, že se mužeme snadno dostat z rozsahu zobrazovaných čísel (odst. 2.2).

	a_0	a_1	a_2	$\dots$	a_{n-1}	a_n
α		αb_0	αb_1		αb_{n-2}	αb_{n-1}
	b_0	b_1	b_2	$\dots$	b_{n-1}	$b_n = p(\alpha)$

Při "ručním" počítání postupně zaplňujeme tuto tabulku.

K zápisu algoritmů lze, kromě slovního vyjádření, užívat především dvou schematických způsobů. Ukážeme si to na Hornerově algoritmu.

Vstup: $n, \alpha, a_0, a_1, a_2, ..., a_n.$

$b_0 = a_0.$

Pro $i = 1, 2, ..., n :$

$\lfloor b_i = \alpha b_{i-1} + a_i.$

Výstup: $p(\alpha) = b_n.$

Znakem $\lfloor$ budeme znázorňovat, které výpočty se mají opakovaně provádět pro postupně se měnící index. Tento způsob zápisu algoritmu je pouze přehledněji zapsaná matematická formule doplněná vstupními a výstupními daty.

1.4.2 Příklad.

1.4.3 Syntetické dělení (dělení lineárním polynomem).

1.4.4 Opakovaný Hornerův algoritmus.

1.4.5 Příklad.

1.4.6 Iterační procesy.

Algoritmy v předcházejících odstavcích jsme získali tak, že jsme určili (jednoduché) rekurentní vztahy mezi jednotlivými mezivýsledky. Často však potřebujeme stanovit limitní hodnotu vhodně zvolené posloupnosti mezivýsledků. V tomto případě uvažovaný rekurentní vztah nazýváme *iterační formulí* a výpočet samotný *iteračním procesem*[4]).

Například pro $a > 0$ bude posloupnost určená rekurentním vztahem (tzv. *Heronův vzorec*)

$$x_n = \frac{1}{2}\left(x_{n-1} + \frac{a}{x_{n-1}}\right), \qquad n = 1, 2, 3, \dots \quad , \tag{1.4.6}$$

konvergovat k číslu $\sqrt{a}$ (prověřte!). Současně vidíme, že tato limita je kořenem rovnice $x^2 - a = 0$.

[4] Z latinského slova iteratio - opakování

Protože žádný algoritmus nemůže obsahovat nekonečně mnoho kroků, musíme proces určený formulí (1.4.6) pro nějaké $n = N$ ukončit. Potom ovšem x_N bude pouze přibližnou hodnotou (aproximací) čísla $\sqrt{a}$ (tj. kořene zmíněné rovnice). Číslo N nevybíráme většinou předem, ale určujeme jej implicitně *zastavovací podmínkou*. Tato podmínka obvykle zní: Pokračuj ve výpočtu podle formule (1.4.6), pokud bude $|x_k - x_{k-1}| \geqq \delta$, kde δ je nějaké předem zvolené (malé) číslo. Zahájení výpočtu provádíme vhodnou volbou x_0, např. $x_0 = a$.

Algoritmus výpočtu $\sqrt{a}$, vycházející z formule (1.4.6), zapíšeme vývojovým diagramem:

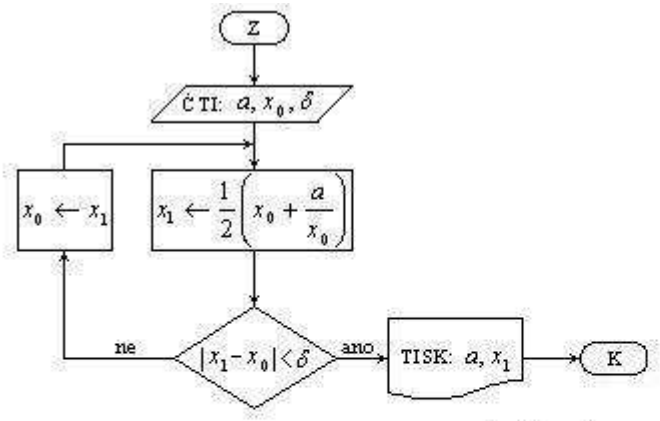


Schéma 2.

Počítáme-li $\sqrt{2}$ podle tohoto algoritmu, pokud nebude $|x_k - x_{k-1}| < 10^{-5}$, postupně dostáváme ($\sqrt{2} = 1,414\,213\,562...$)

$$
\begin{aligned}
x_0 &= 2 \qquad \text{(volba)},\\
x_1 &= \frac{1}{2}\left(2 + \frac{2}{2}\right) = 1,500\,00,\\
x_2 &= 1,416\,667,\\
x_3 &= 1,414\,215,\\
x_4 &= 1,414\,213\,.
\end{aligned}
$$

Proces výpočtu jsme zastavili v okamžiku, kdy se po sobě jdoucí členy posloupnosti shodovaly na pěti desetínných místech. Později (v kap. III) se ještě uvedeme, jak určit odhad čísla $\left|\sqrt{2} - x_4\right|$ pomocí δ.

Jestlíže iterační proces lze zapsat formulí typu

$$x_n = F\left(x_{n-1}\right), \qquad n = 1, 2, 3, ... \quad , \tag{1.4.7}$$

Hovoríme o *jednokrokovém iteračním procesu* [např. právě vztah (1.4.6)]. K zahájení výpočtu potřebujeme znát pouze jednu hodnotu - např. x_0 - a

postupně vypočítáme

$$x_1 = F(x_0), \quad x_2 = F(x_1), \quad x_3 = F(x_2), \ ..., \quad x_k = F(x_{k-1}), \ ... \quad .$$

Členy posloupnosti $\{x_0, x_1, x_2, ...\}$ nazýváme *iteracemi* nebo *postupnými aproximacemi*. Konverguje-li tato posloupnost k limitě $\alpha = \lim_{n\to\infty} x_n = \lim_{n\to\infty} F(x_{n-1})$, potom s rostoucím n se stále více "přibližujeme" k této limitě α. Jak jsme uvedli, než nějaké předem zadané číslo δ.

Jestlíže iterační proces lza zapsat formulí typu

$$x_{n+p} = F(x_n, x_{n+1}, ..., x_{n+p-1}), \qquad n = 0, 1, 2, ... \quad , \tag{1.4.8}$$

hovoříme o *p-krokovém iteračním procesu*. K zahájení výpočtu potřebujeme znát p hodnot, např. $x_0, x_1, ..., x_{p-1}$.

1.5 Rekurence.

Protože rekurentní formule (viz odst. 1.4) jsou součástí celé řady algoritmů, bude užitečné všimnout si jich podrobněji.

1.5.1 Numerické aspekty rekurentních formulí.

Uvažujme např. rekurentní vztah

$$y_{n+2} = \frac{10}{3} y_{n+1} - y_n. \tag{1.5.1}$$

Známé-li hodnoty y_0, y_1, můžeme podle (1.5.1) postupně určit y_2, y_3, y_4, ... atd. Zvolíme-li $y_0 = 1$, $y_1 = \frac{1}{3}$, dostáváme $y_2 = 1/3^2$, $y_3 = 1/3^3$, ..., $y_n = 1/3^n$ atd. Zvolíme-li však $y_0 = 1$, $y_1 = 0.333$, vypočítáme postupně $y_2 = 0.110$; $y_3 = 0.033\,6$; $y_4 = 0.002\,00$; $y_5 = -0.026\,9$; $y_6 = -0.091\,6$, ..., $y_{10} = -5.65$ atd. Vidíme, že malá změna ve vstupních datech vyvolává velkou změnu ve výsledcích. Nebude tedy vhodné takové rekurence k výpočtum užívat. Ale jak poznáme, která rekurence je vhodná a která ne?

Anychom byli schopni posoudit algoritmy reprezentované rekurentními formulemi, uvedeme si o nich některé teoretické výsledky.

1.5.2 Diferenční rovnice.

Rekurentní formuli typu

$$y_{n+p} = F(y_n, y_{n+1}, ..., y_{n+p-1}, n) \tag{1.5.2}$$

budeme nazývat *diferenční rovnicí řádu* p (*p-kroková rekurence*). Funkci (posloupnost) $y_n = y(n)$ definovanou na množině celých čísel n budeme nazývat *řešením diferenční rovnice*, jestliže splňuje rovnici (1.5.2) pro každé n. Toto řešení je určeno jednoznačně, když je dáno p počatečních hodnot, např. y_0, y_1, y_2,..., y_{p-1}.

Rovnici tvaru

$$y_{n+p} = a_1 y_{n+p-1} + ... + a_p y_n \tag{1.5.3}$$

nazýváme *homogenní lineární diferenční rovnicí řádu p s konstantními koeficienty*. Posloupnost $\{y_n\}$, kde $y_n = Cr^n$ ($C \neq 0, r \neq 0$) je řešením rovnice (1.5.3), právě když číslo r je řešením algebrické rovnice

$$r^p = a_1 r^{p-1} + a_2 r^{p-2} + ... + a_p . \tag{1.5.4}$$

Rovnici (1.5.4) nazýváme *charakteristickou rovnicí* rovnice (1.5.3).

Má-li charakteristická rovnice p různých kořenů $r_1\, r_2, ..., r_p$, potom funkce

$$y_n = C_1 r_1^n + C_2 r_2^n + ... + C_p r_p^n \tag{1.5.5}$$

je tzv. *obecným řešením rovnice* (1.5.3). *Partikulární řešení* s počátečnimi hodnotami $y_0, y_1, ..., y_{p-1}$ najdeme tak, že konstanty $C_1, C_2, ..., C_p$ určíme ze soustavy

$$\begin{pmatrix} 1 & 1 & \dots & 1 \\ r_1 & r_2 & \dots & r_p \\ \vdots & & & \vdots \\ r_1^{p-1} & r_2^{p-1} & \dots & r_p^{p-1} \end{pmatrix} \begin{pmatrix} C_1 \\ C_2 \\ \vdots \\ C_p \end{pmatrix} = \begin{pmatrix} y_0 \\ y_1 \\ \vdots \\ y_{p-1} \end{pmatrix}$$

Je-li r_i m-násobný kořen ($m \geqq 1$) charakteristické rovnice, potom každá funkce tvaru

$$y_n = P(n) r_i^n ; \tag{1.5.6}$$

kde P je libovolný polynom stupně $m - 1$, je řešením rovnice (1.5.3).

Obecné řešení homogenní diferenční rovnice je potom lineární kombinací řešení tvaru (1.5.6) příslušných ke všem různým kořenům charakteristické rovnice [i zde je p konstant; jsou to koeficienty všech polynomů z (1.5.6)].

1.5.3 Příklad.

Necht' $a_n = a(n)$, $f_n = f(n)$ jsou dané funkce argumentu n. Stanovme obecné řešení *(nehomegenní) diferenční rovnice 1. řádu*

$$y_{n+1} = a_n y_n + f_n, \qquad a_n = 0. \tag{1.5.7}$$

Obecné řešení musí obsahovat jednu libovolnou konstantu. Pro $n = 0, 1, 2, ...$ postupně dostáváme

$$\begin{aligned} y_1 &= a_0 y_0 + f_0, \\ y_2 &= a_1 a_0 y_0 + a_1 f_0 + f_1, \\ &\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ y_n &= A_{n0} y_0 + \sum_{j=1}^{n} A_{nj} f_{j-1}, \end{aligned} \tag{1.5.8}$$

kde

$$A_{kj} = a_{k-1} a_{k-2} ... a_j; \qquad A_{kk} = 1, \qquad k = 1, 2, ..., n.$$

Pokud $a_k = a$ (rovnice s konstantním koeficientem), pak $A_{kj} = a^{k-j}$, a tedy

$$y_n = a^n y_0 + \sum_{j=1}^{n} a^{n-j} f_{j-1}. \tag{1.5.9}$$

1.5.4 Příklad.

Stanovme řešení nehomogenní diferenční rovnice 1.řádu

$$y_{n+1} = 2y_n + b^n, \qquad b \neq 0.$$

s počáteční podmínkou $y_0 = 1$

1. metoda: Podle (1.5.9) (tj. postupným dosazováním do rovnice) dostaneme:

$$y_n = 2^n + \sum_{j=1}^{n} 2^{n-j} b^{j-1}$$

2. metoda: Hledáme řešení ve tvaru $y_n = Cb^n$ (tvar pravé strany). Dosazením do rovnice dostaneme:

$$Cb^{n+1} = 2Cb^n + b^n, \qquad tj. \quad C = \frac{1}{b-2}, \qquad b \neq 2.$$

Protože obecné řešení příslušné homogenní rovnice $u_{n+1} = 2u_n$ je $u_n = K \cdot 2^n$, bude obecné řešení dané nehomogenní rovnice mít tvar (analogie s diferenciálnimi rovnicemi!)

$$y_n = K \cdot 2^n + \frac{1}{b-2} b^n.$$

Konstantu K určíme pomocí počáteční podmínky. Pro $n = 0$ je

$$1 = K + \frac{1}{b-2},$$

a tedy

$$y_n = 2^n + \frac{b^n - 2^n}{b-2}, \qquad b \neq 2.$$

1.5.5 Příklad.

Řešme diferenční rovnici $T_{n+1}(x) - 2xT_n(x) + T_{n-1}(x) = 0$, $n \geqq 1$, $-1 < x < 1$, $T_0(x) = 1$, $T_1(x) = x$.

Charakteristická rovnice $r^2 - 2xr + 1 = 0$ má kořeny $r_{1,2} = x \pm \mathrm{i}\sqrt{(1-x^2)}$. Položme $x = \cos t$; potom $r_{1,2} = \cos t \pm \mathrm{i} \sin t = e^{\pm it}$. Obecné řešení má tedy tvar

$$T_n(x) = C_1 e^{int} + C_2 e^{-int}.$$

Z počátečních podmínek dostaneme $C_1 = C_2 = \frac{1}{2}$, a tedy

$$T_n(x) = \frac{1}{2}\left(e^{int} + e^{-int}\right) = \cos n(\arccos x).$$

Funkcím $T_n(x)$ se říká *Čebyševovy polynomy* a jsou důležité v teorii aproximací.

Uved'me si několik prvních Čebyševových polynomů:

$$\begin{aligned}
T_0(x) &= 1, \\
T_1(x) &= x, \\
T_2(x) &= 2x^2 - 1, \\
T_3(x) &= 4x^3 - 3x, \\
T_4(x) &= 8x^4 - 8x^2 + 1 \quad \text{atd.}
\end{aligned}$$

1.5.6 Soustava lineárních diferenčních rovnic 1. řádu.

Uvažujme soustavu k diferenčních rovnic (zapsaných maticově)

$$\mathbf{y}_{n+1} = \mathbf{A}_n\mathbf{y}_n + \mathbf{f}_n, \tag{1.5.10}$$

kde

$$\mathbf{y}_n = \begin{pmatrix} y_{1,n} \\ y_{2,n} \\ \vdots \\ y_{k,n} \end{pmatrix}; \qquad \mathbf{A}_n = \begin{pmatrix} a_{11}^{(n)} & \dots & a_{1k}^{(n)} \\ a_{21}^{(n)} & \dots & a_{2k}^{(n)} \\ \vdots & & \vdots \\ a_{k1}^{(n)} & \dots & a_{kk}^{(n)} \end{pmatrix}; \qquad \mathbf{f}_n = \begin{pmatrix} f_{1,n} \\ f_{2,n} \\ \vdots \\ f_{k,n} \end{pmatrix}.$$

Analogický jako v příkl. 1.5.3 postupným dosazováním dostaneme

$$\mathbf{y}_n = \mathbf{B}_{n0}\mathbf{y}_0 + \sum_{j=1}^{n} \mathbf{B}_{nj}\mathbf{f}_{j-1}, \tag{1.5.11}$$

kde

$$\mathbf{B}_{kj} = \mathbf{A}_{k-1}\mathbf{A}_{k-2}...\mathbf{A}_j, \qquad \mathbf{B}_{kk} = \mathbf{I}, \quad k = 1, 2, ..., n,$$

$\mathbf{I}$ je identická matice. Když

$$\mathbf{A}_k = \mathbf{A}, \qquad \text{potom} \qquad \mathbf{B}_{kj} = \mathbf{A}^{k-j},$$

a tedy

$$\mathbf{y}_n = \mathbf{A}^n\mathbf{y}_0 + \sum_{j=1}^{n} \mathbf{A}^{n-j}\mathbf{f}_{j-1}. \tag{1.5.12}$$

1.5.7 Diferenční rovnice 2. řádu s okrajovými podmínkami.

Uvažujme diferenční rovnici 2. řádu

$$a_n y_{n+1} + b_n y_n + c_n y_{n-1} = f_n \tag{1.5.13}$$

a předpokládejme že jsou známy (kromě a_n, b_n, c_n, f_n) hodnoty y_0, y_N, $N > 1$. Potom k jednoznačnému určení řešení je třeba řešit soustavu $N-1$ rovnic pro neznámé $y_1, y_2, ..., y_{N-1}$,

$$\begin{aligned} b_1y_1 + a_1y_2 &= f_1 - c_1y_0, \\ c_2y_1 + b_2y_2 + a_2y_3 &= f_2, \\ \dots\dots\dots\dots\dots\dots & \quad \dots\dots\dots\dots\dots \\ c_{N-1}y_{N-2} + b_{N-1}y_{N-1} &= f_{N-1} - a_Ny_N. \end{aligned}$$

1.5.8 Numerické aspekty.

Vrátíme-li se k rovnici (1.5.1), zjistíme, že obecným řešením je funkce $\left(r_1 = 3, r_2 = \frac{1}{3}\right)$

$$y_n = C_1 \cdot 3^n + C_2 \cdot 3^{-n},$$

a tedy pro počáteční hodnoty y_0, y_1 máme

$$y_n = \frac{3y_1 - y_0}{8} \cdot 3^n + \frac{3\,(3y_0 - y_1)}{8} \cdot 3^{-n}.$$

Pokud $3y_1 \neq y_0$ bude $\lim\limits_{n\to\infty} |y_n| = \infty$. Tato skutečnost se nám nepříjemně projevila právě u rovnice (1.5.1), pokud jsme zvolili $y_0 = 1\,,\ y_1 = 0,333\,.$

Bude-li aspoň jeden kořen charakteristické rovnice splňovat podmínku

$$|r_i| > 1, \tag{1.5.14}$$

bude v obecném řešení rostoucí člen, který se může díky zaokrouhlovacím chybám nepříjemně projevit na výsledku, i když je tento člen počátečními podmínkami teoreticky anulován. Podmínka (1.5.14) nám vždy signalizuje potíže, které mohou nastat při realizaci algoritmu vycházejícího z rekurentní formule typu diferenční rovnice. Analogický příklad najde čtenář v odst. 4.4.1.

U diferenčních rovnic s okrajovými podmínkami je situace poněkud složitější. Zde jde o to, aby pro velká N členy $y_1, y_2, ..., y_N$ zůstávaly v nějakých rozumných mezích. Podrobnější informace najde čtenář např. v kníze Godunov, S. K. - Rjabeňkij, V. S.: Raznostnyje schemy. Moskva 1973.

1.6 Cvičení.

1.6.1

Nakreslete vývojový diagram opakovaného Hornerova schématu.

1.6.2

Nakreslete vývojový diagram výpočtu řetězového zlomku.

1.6.3

Posud'te, zda iterační formule $x_{n+2} = x_n + x_{n+1}$ reprezentuje konvergentní iterační proces.

[Diverguje: rekurentní zápis tzv. *Fibonacciovy posloupnosti.*]

1.6.4

Řešte diferenční rovnici $y_{n+2} - 6y_{n+1} + 9y_n = 0\,,\ y_0 = 1\,,\ y_2 = 3\,.$

[$y_n = 3^n - n \cdot 3^{n-1}$; užijte vztahu (1.5.6) a analogie s diferenciálnimi rovnicemi.]

1.6.5

Řešte diferenční rovnici $y_{n+1} = y_n + e^n\,,\ y_0 = 1\,.$

[$y_n = 1 + (e^n - 1)\,/\,(e - 1)$. Návod. Hledejte řešení ve tvaru $y_n = u_n + v_n\,,$ kde u_n je obecné řešení homogenní rovnice, a řešení nehomogenní rovnice předpokládejte ve tvaru $v_n = Ae^n\,,$ A je konstanta.]

1.6.6

Řešte diferenční rovnici $y_{n+2} - y_{n+1} - 2y_n = 2n^2 + 2\,,\ y_0 = 0\,,\ y_1 = 1\,.$

[$y_n = 3 \cdot 2^n - n^2 - n - 3$. Návod. $y_n = u_n + v_n\,,\ v_n = An^2 + Bn + C\,;$ dále viz cvič. 1.6.5.]

1.6.7

Řešte diferenční rovnici $y_{n+1} - 2y_n + 2y_{n-1} = 0\,,\ y_0 = 1\,,\ y_1 = 2\,.$

[$y_n = \left(\sqrt{2}\right)^{n+1} \sin\left[(n+1)\,\pi/4\right]$. Návod. Využijte skutečnosti, že $\sqrt{2}\exp\left(\pm \mathrm{i}\pi/4\right) = 1 + \mathrm{i}\,.$]

1.6.8

Besselovy funkce jsou vázány rekurentním vztahem

$$\mathbf{J}_{p+1}(x) = \frac{2p}{x}\mathbf{J}_p(x) - \mathbf{J}_{p-1}(x)$$

Je-li $\mathbf{J}_0(1) \approx 0,765\,2\,,\ \mathbf{J}_1(1) \approx 0,440\,1\,,$ vypočtěte $\mathbf{J}_2(1)\,,\ \mathbf{J}_3(1)\,,\ \mathbf{J}_4(1)\,,$ $\mathbf{J}_5(1)$ atd. Jak jsou výsledky v souladu s faktem, že $\lim\limits_{n\to\infty} \mathbf{J}_p(1) = 0$? Vysvětlete.

1.6.9

Jsou dány koeficienty dvou polynomů

$$p(x) = \sum_{i=1}^{m} a_i x^{i-1}, \qquad q(x) = \sum_{j=1}^{m} b_j x^{j-1}.$$

Odvod'te algoritmus pro výpočet koeficientů součinu $p(x)$, $q(x)$.

2 Zobrazení čísel v počítači

2.1 Množina počítačových čísel.

Každý počítač provádějící vědeckotechnické výpočty může pracovat s čísly, která se dají vyjádřit v *semilogaritmickém tvaru* s *normalizovanou mantisou.*

$$\alpha = \operatorname{sgn}\alpha \left(\frac{\alpha_1}{q} + \frac{\alpha_2}{q^2} + ... + \frac{\alpha_l}{q^l} \right) q^b, \qquad \alpha_1 \neq 0, \tag{2.1.1}$$

kde $q > 1$ je *základ*, $\alpha_i = \{0, 1, 2, ..., q-1\}$ jsou *číslice mantisy* a b je exponent. Přirozené číslo t určuje počet číslic mantisy a je dáno konstrukcí počítače, stejně jako přirozené číslo q. Konstrukcí počítače jsou též dány hranice m_1, m_2 celého čísla b.

Množina čísel α není nekonečná; má přesně $2(q-1)q^{l-1}(m_2 - m_1 + 1) + +1$ prvků (viz [7a]) a budeme ji označovat symbolem

$$M(q, t, m_1, m_2), \qquad \text{nebo zkráceně} \qquad M(q, t).$$

Počítači, který pracuje s čísly z $M(q,t)$, někdy říkáme *q-aditický t-místný počítač.*

Například číslo $\alpha = 13,25$ lze zobrazit bez vstupní chyby do množiny:

$M(10,4)$, nebot' $\alpha = 0,132\,5 \cdot 10^2$,
$M(2,6)$, nebot' $\alpha = 0,110\,101 \cdot 2^4$, $(13,25 = 1\cdot 2^3 + 1\cdot 2^2 + 0\cdot 2^1 + +1\cdot 2^0 + 0\cdot 2^{-1} + 1\cdot 2^{-2})$,
$M(8,3)$, nebot' $\alpha = 0,152 \cdot 8^2$, $(13,25 = 1\cdot 8^1 + 5\cdot 8^0 + 2\cdot 8^{-1})$,
$M(16,2)$, nebot' $\alpha = 0,D4 \cdot 16^1$, $[13,25 = (13/16 + 4/16^2)\cdot 16^1$

a pro cifry 10, 11, 12, 13, 14, 15 užíváme v hexadecimálním (šestnáckovém) systému znaků [A, B, C, D, E, F].

Pro informace si uved'me množiny $M(q,t,m_1,m_2)$, se kterými pracují následující počítače. (q^{l-1} je charakteristika relativní přesnosti aritmetiky.) (Tab. 1.)

2.1.1 Poznámka.

Systém popsaný v odst. 2.1 se nazývá systém s *pohyblivou řádovou čárkou* (pracuje se s konstantním počtem číslic mantisy). Pro některé výpočty je výhodnější pracovat s čísly v tzv. *pevné řádové čárce* (pracuje se s konstantním počtem desetinných míst; sem patří také čísla typu "integer" ve Fortranu či Algolu).

Tab. 1.

počítač	q	t	m_1	m_2	q^{1-t}
PDP-11	2	24	-128	127	$1,19 \cdot 10^{-7}$
Hewlett Packard HP-45	10	10	-98	100	$1,00 \cdot 10^{-9}$
Texas Instruments SR-5x	10	12	-98	100	$1,00 \cdot 10^{-11}$
IBM 360 a 370	16	6	-64	63	$9,54 \cdot 10^{-7}$
	16	14	-64	63	$2,22 \cdot 10^{-16}$ (DP)
ICL 4-72 Tesla 200 EC 1030 a 1040	jako IBM 360				

Poznamenejme ještě, že některé počítače mohou pracovat kromě systému $M(q,t)$ také se systémem $M(q,\tau)$, přičemž obvykle $\tau \geqq 2t$ a poněkud nelogicky se pak hovoří o *dvojnásobné aritmetice* a *přesnosti*; srov. IBM 360/370 v tab. 1.

2.2 Zobrazení reálných čísel do množiny $M(q,t)$.

Při zavádění vstupních dat do počítače musí počítač reálnému číslu $x \in \in \mathbf{R}$ přiřadit číslo $\alpha \in M(q,t)$. Tato přiřazení se realizuje dvěma způsoby - *řezáním* nebo *zaokrouhlováním*:

Je-li

$$x = aq^b = \operatorname{sgn} x \left(\sum_{k=1}^{\infty} x_k q^{-k} \right) q^b \in \mathbf{R},$$

potom obraz čísla $x \in \mathbf{R}$ označíme

$$\alpha = \widetilde{a} q^b = \operatorname{sgn} \alpha \left(\sum_{k=1}^{t} \alpha_k q^{-k} \right) q^b \in M,$$

Provádíme-li řezání, klademe

$$\alpha_k = x_k \qquad \text{pro} \qquad k = 1, 2, ..., t.$$

a provádíme-li zaokrouhlování [5]), klademe

$$\alpha_k = x_k \qquad \text{pro} \qquad k = 1, 2, ..., t-1,$$

$$\alpha_i = \begin{cases} x_i + 1 & \text{pro} \quad x_{i+1} \geqq \frac{q}{2}, \\ x_i & \text{pro} \quad x_i \quad < \frac{q}{2}. \end{cases}$$

Označíme $\gamma : \mathbf{R} \to M$ takto definované zobrazení množiny reálných čísel do množiny M, tj. $\alpha = \gamma(x)$. Protože $a \geqq q^{-1}$, pak pro relativní chybu platí

$$\left| \frac{x - \gamma(x)}{x} \right| \leqq \kappa q^{1-t}, \tag{2.2.1}$$

kde $\kappa = 1$ při řezání a $\kappa = \frac{1}{2}$ při zaokrouhlování, nebot'

$$\frac{x - a}{x} = \frac{q^b (a - \widetilde{a})}{q^b a} \qquad \text{a} \qquad |a - \widetilde{a}| \leqq \kappa q^{-1}.$$

Číslu κq^{1-t} se také často říká *strojové epsilon.*

Např. obrazem čísla $x = 3,141\,592$ v množině $M(10,5)$ při řezání ($\kappa = 1$) bude číslo $\alpha = \gamma(x) = 0,314\,15 \cdot 10^1$, nebot' $|0,314\,159\,2 - 0,314\,15| = 9,26 \cdot 10^{-6} \leqq 1 \cdot 10^{-5}$. při zaokrouhlování bude obrazem téhož čísla x číslo $\alpha = \gamma(x) = 0,314\,16 \cdot 10^1$, nebot' $|0,314\,159\,2 - 0,314\,16| = 0,08 \cdot 10^{-5} \leqq 0,5 \cdot 10^{-5}$.

[5]) *Zaokrouhlené číslo* $\alpha \in M(q,t)$ čísla $x \in \mathbf{R}$ se obecně definuje podmínkou, že musí být $|x - \alpha| = \min_{\beta \in M} |x - \beta|$. *Uříznuté číslo* $\alpha \in M(q,t)$ čísla $x \in \mathbf{R}$ se obecně definuje podmínkou, že musí být

$$\alpha = \begin{cases} \max \beta \in M \,|\, \beta \leq x & \text{pro} \quad x > 0, \\ \min \beta \in M \,|\, \beta \geq x & \text{pro} \quad x < 0. \end{cases}$$

2.3 Aritmické operace v množině $M(q,t)$.

Množina $M(q,t)$ není z hlediska základních aritmetických operací uzavřená, tzn. výsledek nějaké operace s čísly z M nemusí být v M. Např. součin čísel s t-místnou mantisou má obecně $2t$ nebo $2t-1$ číslic.

Pro ilustraci aritmetických operací v typické aritmetické jednotce počítače zvolíme systém IBM 370, proto předpokládáme, že pro sčítání (odčítání) má sčítací registr jedno rezervní místo (níže označené zatržením), o které se v procesu sčítání rozšíří mantisy sčítanců. Toto místo není uživateli dostupné. Sčítání dvou čísel v systému IBM 370 spočívá v porovnání exponentů a sečtení mantis. Mantisa čísla s menším exponentem se posune doprava tak, aby se exponenty vyrovnaly (denormalizace). Pak se mantisy sečtou; vznikne-li při sčítání mantisa $\geqq 1$, provede se posun výsledné mantisy o jedno místo doprava a exponent se zvýší o jednotku. V případě, že výsledná mantisa není normalizovaná, posune se podle potřeby doleva a exponent se odpovidajícím způsobem sníží. Chybějící číslice vpravo se doplní nulami.

Sčítání v $M(10,6)$ se tedy podle tohoto systému provádí takto:

$$0,256\,845\cdot 10^1 + 0,327\,917\cdot 10^{-2} \to 0,256\,845\lfloor 0\cdot 10^1 + 0,000\,327\lfloor 9\cdot 10^1 \to$$

$$\to 0,257\,172\lfloor 9\cdot 10^1 \to 0,257\,172\cdot 10^1$$

(řezání - v uvedeném systému není možnost volby zaokrouhlování)

Timto způsobem je tedy přiřazeno součtu dvou čísel z M opět číslo v M.

Je-li tedy $\gamma : \mathbf{R} \to M$ a znamená-li symbol $\circledast$ aritmetickou operací prováděnou na počítači a symbol $*$ tutéž operaci nad tělesem reálných čísel, dá se ověřit, že naprosté většině počítačů platí vztah

$$x \circledast y = \gamma\,(x * y)\,, \qquad x, y \in M,$$

a

$$\left|\frac{x \circledast y - x * y}{x * y}\right| \leqq \kappa q^{1-t},$$

neboli $x \circledast y = (x * y)(1+\varepsilon)$, kde $\varepsilon \leqq \kappa q^{1-t}$ pro všechna $x, y \in M$.

Aritemtickou operaci na počítači můžeme tedy napodobit tím, že provedeme obvyklou operaci a její výsledek zobrazíme na příslušné strojové číslo z M.

Dá se ukázat (viz např. [8]), že pro aritmetické operace v M neplatí obecně asociativní a distributivní zákon. Jedním z důsledku neplatnosti axiómů aritmeticky reálných čísel při realizaci na počítači dávat rozdílné výsledky.

Jestliže se tedy algoritmus (1.3.2) realizuje na počítači, potom vstupní a výstupní data $\tilde{\mathbf{x}}, \tilde{\mathbf{y}}$ strojového algoritmu (1.3.3) jsou dána čísly z $M(q,t)$.

2.3.1 Příklad.

Předpokládejme, že je naší úlohou stanovit součet

$$\lim_{n=1}^{10\,000} \frac{1}{n^2} \quad (\approx 1,644\,834)$$

v systemu $M(16,6)$. Při sčítání v přírozeném pořadí $\left[\left(1+\frac{1}{4}\right)+\frac{1}{9}\right]+...$ dostaneme celkový součet s chybou $1\,317 \cdot 10^{-6}$. Sčítáme-li však v opačném pořadí.

$$\left[\left(\frac{1}{10\,000^2}+\frac{1}{9\,999^2}\right)+\frac{1}{9\,998^2}\right]+...$$

je chyba výsledku $2 \cdot 10^{-6}$. V prvním případě se totiž od určitého n počínaje součet již nemění.

Obecně pro sčítání podobných součtů, kde se absolutní hodnoty sčítanců liší o několik řádů, platí, že pro dosažení vyšší přesnosti je třeba sčítat od členů s nejmenší absolutní hodnotou k členům s největší absolutní hodnotou. V praxi však tento efekt nemusí mít vždy rozhodující význam.

2.4 Cvičení.

2.4.1

Vypočtěte Hornerovým algoritmem $P(1,12)$, kde $P(x) = 2x^4 + 16x^3 + x^2 - 74x + 56$: a) v $M(10,4)$; b) v $M(10,6)$; c) v $M(10,10)$.

a) $-0,110\,0 \cdot 10^{-1}$; b) $0,300\,000 \cdot 10^{-3}$; c) $0,286\,720\,000\,0 \cdot 10^{-3}$. Návod. Modelujte práci v jednotlivých množinách $M(10,t)$ např. na kapesním kalkulátoru tak, že budete v každém výpočetním kroku uřezávat nebo zaokrouhlovat mantisy na příslušný počet číslic.

2.4.2

Stanovte obrazy čísel π, e, $\sqrt{2}$, $\sqrt{3}$ v množínách a) $M(10,2)$; b) $M(10,4)$; c) $M(10,6)$. Užijte zaokrouhlování i řezání.

3 Problém přesnosti výsledků

3.1 Zdroje chyb.

Při vytváření matematického modelu reálného problému vždy provádíme jisté idealizace. Rozdíl řešení idealizovaného problému a řešení reálného problému nazýváme *chybou matematického modelu*. Z velikosti této chyby posuzujeme zpětně vhodnost či nevhodnost zvoleného matematického modelu (matematické úlohy).

Jestliže k řešení matematické úlohy použijeme metodu, která nám neposkytne přesné (teoretické) řešení dané úlohy, pak chybu, které se dopustíme, nazýváme *chybou metody*. Typickým příkladem je chyba, které se dopustíme, když za limitu nekonečné posloupnosti vezmeme některý její člen s dostatečně velkým indexem.

Často řešíme matematickou úlohu tím, že pomocí jistých metod ji nahradíme (aproximujeme) úlohou jednodušší - obvykle již úlohou numerickou - a rozdíl řešení těchto dvou úloh nazýváme *chybou aproximace*. Jde také o případ chyby metody.

K posouzení přesnosti výsledku musíme ještě vzít v úvahu *chyby ve vstupních datech* dané jednak chybami měření, jednak způsobené zobrazením vstupních dat do množiny $M(q,t)$ počítače.

Nakonec jsou to *chyby zaokrouhlovací* kam zahrnujeme všechny nepřesnosti způsobené realizací algoritmu v počítači včetně nepřesného provádění aritmetických operací.

3.1.1 Příklad.

Máme-li popsat pohyb kyvadla (závislost dráhy s na čase), vyjdeme z Newtonova zákona

$$m\frac{d^2s}{dt^2} = -F$$

a dostaneme diferenciální rovnici $(s = L\varphi)$

$$\frac{d^2\varphi}{dt^2} = -\frac{g}{L}\sin\varphi$$

pro neznámou (úhlovou) výchylku $\varphi = \varphi(t)$ (g je gravitační zrychlení, L je délka kyvadla). Rozdíl funkce φ a skutečné (naměřené) závislosti je chybou modelu (idealizace), nebot' při aplikaci Newtonova zákona neuvažujeme např. odpor prostředí a jiné vlivy.

Odvozenou nelineární rovnici obvykle nahrazujeme rovnici lineární (protože pro malé výchylky φ je přibližně $\sin\varphi \approx \varphi$)

$$\frac{d^2\varphi}{dt^2} = -\frac{g}{L}\varphi$$

a rozdíl řešení těchto dvou rovnic je chybou metody linearizace[6]). Kteroukoliv z těchto rovnic ovšem můžeme řešit tak, že ji nahradíme úlohou numerickou, např. diferenční rovnicí. Pak se dopustíme chyby numerické metody. Při konkrétních výpočtech pochopitelně nemůžeme vyloučit ani zaokrouhlovací chyby.

Podotkněme, že vstupními údaji uvažovaných úloh jsou konstanty g, L a čísla $\varphi(0)$, $\varphi'(0)$ - tzv. počáteční podmínky. Přejdeme-li k numerické úloze, pak máme situaci obdobnou té, která byla popsána v příkl. 1.2.2.

3.2 Aproximace čísel.

3.2.1 Užitečná označení.

V numerické praxi používáme některá označení, která nejsou zcela přesná v matematickém smyslu:

$a \ll b$ (nebo $b \gg a$); čteme : "a je mnohem menší než b"

nebo "b je mnohem menší než a"

Z kontextu musí být patrné, zda máme na mysli, že např. $a < b/2$ nebo, že $a < b/100$.

$a \approx b$; čteme : "a je přibližně rovno b"

a znamená totéž co nerovnost $|a - b| \ll c$, kde velikost c je zřejmá z kontextu, nebot' obecně nemůžeme např. říci, že $10^{-6} \approx 0$.

$a \lesssim b$ (nebo $b \lesssim a$); čteme : "a je menší nebo přibližně rovno b"

a znamená totéž co " $a < b$ nebo $a \approx b$ "

Příležitostně budeme také užívat následujících symbolů, které již maji matematicky zcela přesný význam:

Symbol

$$f(x) = O(g(x)), \quad \text{když} \quad x \to a$$

[6]) Má-li být linearizovaný model přesný např. s chybou $0,5 \cdot 10^{-4}$ (na 4 desetinná místa), musíme se omezít na taková φ, pro něž $|\sin\varphi - \varphi| \leqq 0,5 \cdot 10^{-4}$, tj. pro $|\varphi| \leqq \leqq 3,8^\circ \approx 0,067$ rad.

znamená, že funkce $|f(x)/g(x)|$ je omezená, když $x \to a$ (místo čísla a můžeme brát i $+\infty$ nebo $-\infty$):

symbol

$$f(x) = o(g(x)), \quad \text{když} \quad x \to a$$

znamená, že $\lim\limits_{x\to a}[f(x)/g(x)] = 0$.

Bude-li např. $\lim\limits_{h\to 0}[F(x)/h^n] = A \neq 0$, pak řekneme, že *funkce* $F(h)$ *je typu* $O(h^n)$ nebo *řádu* h^n, a píšeme $F(h) = O(h^n)$.

3.2.2 Absolutní a relativní chyba.

Ve výpočtech jsme často nuceni přibližně nahradit číslo x číslem $\tilde{x}$. Číslo $\tilde{x}$ potom nazýváme *aproximací čísla* x. Rozdíl $|x - \tilde{x}| = \triangle x$ nazýváme *absolutní chybou aproximace* $\tilde{x}$.

Číslo $\varepsilon(\tilde{x}) \geqq 0$, pro které platí

$$|x - \tilde{x}| \leqq \varepsilon(\tilde{x}), \tag{3.2.1}$$

nazýváme *odhadem absolutní chyby*.

Číslo

$$\frac{\triangle x}{x} = \frac{x - \tilde{x}}{x}, \quad x \neq 0.$$

nazýváme *relativní chybou aproximace* $\tilde{x}$ [7]). Relativní chyba se často uvádí v procentech.

Číslo $\delta(\tilde{x})$, pro které platí

$$\left|\frac{x - \tilde{x}}{x}\right| \leqq \frac{\varepsilon(\tilde{x})}{|x|} = \delta(\tilde{x}), \tag{3.2.2}$$

nazýváme *odhadem relativní chyby*.

Například pro $x = \pi = 3,141\,59...$, $\tilde{x} = 3,14$ je $\triangle x = +0,001\,59...$, $\varepsilon(\tilde{x}) = 2 \cdot 10^{-3}$ (lze ovšem vzít i $1,6 \cdot 10^{-3}$; $1,7 \cdot 10^{-3}$ atd.). Dále pak

$$\frac{\triangle x}{x} \approx \frac{\triangle x}{\tilde{x}} \approx +\frac{0,001\,59}{3,14} \approx +0,000\,506 = 0,050\,6\%$$

[7]) Někdy se z praktických důvodu relativní chybou rozumí číslo $\triangle x/\tilde{x}$ (viz [8],[1]).

3.2.3

Nerovnost (3.2.2) neříká nic jiného, než že

$$\tilde{x} = x(1+\delta), \tag{3.2.3}$$

kde $|\delta| \leqq \delta(\tilde{x})$.

Analogicky nerovnost (3.2.1) neříká nic jiného, než že

$$x \in \langle \tilde{x} - \varepsilon(\tilde{x}), \tilde{x} + \varepsilon(\tilde{x}) \rangle ,$$

což můžeme zapsat symbolicky

$$x = \tilde{x} \pm \varepsilon(\tilde{x}).$$

Např. zápis $x = 0,587\,6 \pm 0,001\,4$ znamená pouze, že $0,586\,2 \leqq x \leqq 0,589\,0$.

V odstavci 2.2 jsme si ukázali, že odhad relativní chyby aproximace $\tilde{x} = \gamma(x)$ čísla $x \in \mathbf{R}$ dané zobrazením do množiny $M(q,t)$ je dán vztahem (2.2.1) (viz poslední sloupec tab. 1).

3.2.4 Šíření chyb aritmetických operacích.

Budeme předpokládat, že provádíme přesné aritmetické operace s nepřesnými čísly, tj. s aproximacemi a budeme předpokládat, že známe chyby (resp. odhady chyb) těchto aproximací. Zajímá nás, s jakou přesností lze stanovit výsledek, tj. jaká je chyba (resp. její odhad) výsledku.

Necht'

$$x_i = \tilde{x_i} + \triangle x_i, \quad |\triangle x_i| \leqq \varepsilon_i, \quad \left| \frac{\triangle x_i}{x_i} \right| \leqq \delta_i, \quad i = 1,2,$$

kde ε_i, resp. δ_i je odhad absolutní, resp. relativní chyby aproximace $\tilde{x_i}$.

(a) Je-li $\tilde{u} = \tilde{x_1} + \tilde{x_2}$ *aproximace součtu* $u = x_1 + x_2$, potom

$$u = \tilde{x_1} + \triangle x_1 + \tilde{x_2} + \triangle x_2 = \tilde{u} + \triangle u,$$

kde

$$\triangle u = \triangle x_1 + \triangle x_2,$$

a platí

$$|\triangle u| \leqq |\triangle x_1| + |\triangle x_2| \leqq \varepsilon_1 + \varepsilon_2.$$

(b) Je-li $\tilde{v} = \tilde{x_1} - \tilde{x_2}$ *aproximace rozdílu* $v = x_1 - x_2$, potom

$$\triangle v = \triangle x_1 - \triangle x_2,$$

a platí

$$|\triangle v| \leqq |\triangle x_1| + |\triangle x_2| \leqq \varepsilon_1 + \varepsilon_2.$$

(c) Je-li $\tilde{w} = \tilde{x_1}\tilde{x_2}$ *aproximace součinu* $w = x_1 x_2$, potom

$$w = (\tilde{x_1} + \triangle x_1)(\tilde{x_2} + \triangle x_2) = \tilde{x_1}\tilde{x_2} + \tilde{x_1} \triangle x_2 + \tilde{x_2} \triangle x_1 + \triangle x_1 \triangle x_2 = \tilde{w} + \triangle w;$$

klademe

$$\triangle w \approx \tilde{x_1} \triangle x_2 + \tilde{x_2} \triangle x_1 \quad \text{(člen } \triangle x_1 \triangle x_2 \text{ nebereme v úvahu)}$$

a platí

$$|\triangle w| \lesssim |\tilde{x_1}|\, \varepsilon_2 + |\tilde{x_2}|\, \varepsilon_1.$$

(d) Je-li $\tilde{z} = \tilde{x_1}/\tilde{x_2}$ *aproximace podílu* $z = x_1/x_2$, potom

$$z = \frac{\tilde{x_1} + \triangle x_1}{\tilde{x_2} + \triangle x_2} = \tilde{z} + \triangle z$$

kde

$$\triangle z = \frac{\tilde{x_2} \triangle x_1 - \tilde{x_1} \triangle x_2}{\tilde{x_2}(\tilde{x_2} + \triangle x_2)} \approx \frac{\tilde{x_2} \triangle x_1 - \tilde{x_1} \triangle x_2}{{\tilde{x_2}}^2},$$

a platí

$$\triangle z \lesssim \frac{|\tilde{x_2}|\, \varepsilon_1 + |\tilde{x_1}|\, \varepsilon_2}{|\tilde{x_2}|^2}.$$

Pro relativní chyby můžeme z pravidel (a), (b), (c), (d) odvodit

(A)
$$\frac{\triangle u}{u} \approx \frac{\tilde{x_1}}{\tilde{x_1} + \tilde{x_2}} \frac{\triangle x_1}{x_1} + \frac{\tilde{x_2}}{\tilde{x_1} + \tilde{x_2}} \frac{\triangle x_2}{x_2},$$
$$\left|\frac{\triangle u}{u}\right| \lesssim \frac{1}{|\tilde{x_1} + \tilde{x_2}|} \left(|\tilde{x_1}|\, \delta_1 + |\tilde{x_2}|\, \delta_2\right).$$

(B)
$$\frac{\triangle v}{v} \approx \frac{\tilde{x_1}}{\tilde{x_1} - \tilde{x_2}} \frac{\triangle x_1}{x_1} - \frac{\tilde{x_2}}{\tilde{x_1} - \tilde{x_2}} \frac{\triangle x_2}{x_2},$$
$$\left|\frac{\triangle v}{v}\right| \lesssim \frac{1}{|\tilde{x_1} - \tilde{x_2}|} \left(|\tilde{x_1}|\, \delta_1 + |\tilde{x_2}|\, \delta_2\right).$$

Zde si všimneme, že při odčítání blízkých čísel má na velikost relativní chyby rozdílu rozhodující význam číslo $1/(\tilde{x_1} - \tilde{x_2})$.

(C)
$$\frac{\triangle w}{w} \approx \frac{\tilde{x_1} \triangle x_2 + \tilde{x_2} \triangle x_1}{\tilde{x_1}\tilde{x_2}} = \frac{\triangle x_2}{\tilde{x_2}} + \frac{\triangle x_1}{\tilde{x_1}},$$
$$\left|\frac{\triangle w}{w}\right| \lesssim \delta_1 + \delta_2.$$

(D)
$$\frac{\triangle z}{z} \approx \frac{\triangle x_1}{\tilde{x_1}} - \frac{\triangle x_2}{\tilde{x_2}},$$
$$\left|\frac{\triangle z}{z}\right| \lesssim \delta_1 + \delta_2.$$

3.2.5 Příklad.

Je-li $x_1 = \tilde{x_1} + \triangle x_1 = 758\,320$, $\tilde{x_1} = 758\,330$; $x_2 = \tilde{x_2} + \triangle x_2 = 757\,940$, $\tilde{x_2} = 757\,930$ [v $M(10,6)$], potom

$$\triangle x_1 = -10, \quad \triangle x_2 = 10, \quad \left|\frac{\triangle x_1}{x_1}\right| \leqq 1,32 \cdot 10^{-5}, \quad \left|\frac{\triangle x_2}{x_2}\right| \leqq 1,32 \cdot 10^{-5}$$

a dále pak

$$v = x_1 - x_2 = 380, \quad \tilde{v} = \tilde{x_1} - \tilde{x_2} = 400,$$

$$\triangle v = \triangle x_1 - \triangle x_2 = -20, \quad \triangle v \leqq 20,$$

$$\left|\frac{\triangle v}{v}\right| < \frac{20}{380} \approx 5,2 \cdot 10^{-2} \approx 2\,000 \left(\left|\frac{\triangle x_1}{x_1}\right| + \left|\frac{\triangle x_2}{x_2}\right|\right)$$

Pro porovnání $|\triangle u/u| \leqq 1,33 \cdot 10^{-5}$ [podle (A)], $u = x_1 + x_2$.

3.2.6 Platné dekadické číslice.

Uvažujme aproximaci $\tilde{x} = \tilde{a} \cdot 10^b$ čísla $x = a \cdot 10^b$ s normalizovaným tvarem mantisy (viz odst. 2.2).

Řekneme, že j-tá *dekadická číslice* aproximace $\tilde{x}$ (mantisy $\tilde{a}$) je *platná*, platí-li

$$|x - \tilde{x}| \leqq 0,5 \cdot 10^{b-j}, \tag{3.2.4}$$

resp.

$$|a - \tilde{a}| \leqq 0,5 \cdot 10^{-j}. \tag{3.2.5}$$

Platí-li nerovnost (3.2.4) pro nějaké $j = s$ (tedy i pro $j < s$), ale pro $j = s + 1$ už nikoliv, říkáme, že $\tilde{x}$ *má s platných číslic.*

U čísel tvaru $\tilde{x} = \tilde{a} \cdot 10^b$, kde $\tilde{a}$ není normalizováno, určíme počet platných číslic tak, že $\tilde{a}$ nejprve normalizujeme.

V jiných číselných soustavách lze pojem platné číslice zavést analogicky.

U čísel s nenormalizovanou mantisou často užíváme termínu *platné desetinné místo*. Rozumíme tím ta místa za desetinnou čárkou, která jsou obsazena platnými číslicemi ve smyslu (3.2.4), resp. (3.2.5) a nulami před první platnou číslicí.

3.2.7 Příklady.

a) Je-li $x = \pi$, $\tilde{x} = 0,314\,16 \cdot 10^1$, pak

$$|a - \tilde{a}| \leqq 8 \cdot 10^{-7} < 0,5 \cdot 10^{-5},$$

$$\left|\frac{x - \tilde{x}}{x}\right| \leqq 3 \cdot 10^{-6} < 0,5 \cdot 10^{-5}, \quad |x - \tilde{x}| \leqq 8 \cdot 10^{-6} < 0,5 \cdot 10^{-4}.$$

Podle (3.2.4) má $\tilde{x}$ pět platných číslic a aproximace $3,141\,6$ má čtýři platná desetinná místa.

b) Je-li Je-li $x = \pi$, $\tilde{x} = 0,314\,15 \cdot 10^1$, pak

$$|a - \tilde{a}| \leqq 10^{-5} \leqq 0,5 \cdot 10^{-4},$$

$$\left|\frac{x - \tilde{x}}{x}\right| \leqq 3 \cdot 10^{-5} \leqq 0,5 \cdot 10^{-4},$$

$$|x - \tilde{x}| \leqq 10^{-4} \leqq 0,5 \cdot 10^{-3};$$

v tomto případě má $\tilde{x}$ pouze čtyři platné číslice a aproximace $3,141\,5$ má tři platná desetinná místa.

Čtenář necht' si povšimne souvislosti počtu platných číslic s exponentem u nejlepšího odhadu relativní chyby ve tvaru $0,5 \cdot 10^{-p}$ a souvislosti počtu platných desetinných míst s exponentem u nejlepšího odhadu absolutní chyby opět ve tvaru $0,5 \cdot 10^{-p}$ a necht' si na základě tohoto poznatku odvodí praktické pravidlo pro určování počtu platných číslic, resp. počtu platných desetinných míst[8]).

c) Řekneme-li naopak, že např. aproximace $\tilde{x} = 0,001\,478$ nějakého čísla x má tři platné číslice (tj. pět platných desetinných míst - počítáme i nuly za desetinnou čárkou), znamená to, že odhad chyby této aproximace je $0,5 \cdot \cdot 10^{-5}$.

3.2.8 Ztráta platných číslic.

Ze zmíněné souvislosti mezi počtem platných číslic a relativní chybou aproximace vyplývá, že zvětšování relativní chyby se projeví ztrátou platných

[8]) Naznačené pravidlo není vždy v souladu s přesnou definicí (3.2.4). Např. pro $x = 2,555$, $\tilde{x} = 2,55$ je $|\triangle x/x| \leqq 2 \cdot 10^{-3} < 0,5 \cdot 10^{-2}$, a podle praktického pravidla bychom řekli, že aproximace $\tilde{x}$ má dvě platné číslice. Přesná definice nám však řekne, že $|a - \tilde{a}| = |0,255\,5 - 0,255| \leqq 0,5 \cdot 10^{-3}$, a tedy aproximace $\tilde{x}$ má ve skutečnosti tři platné číslice.

číslic a obráceně. Proto se ve výpočtech vyhýbáme rozdílům blízkých čísel, při nichž právě k této ztrátě dochází.

Máme-li např.

$$x_1 = 0,501\,027\,8, \quad \tilde{x_1} = 0,501\,0,$$

$$x_2 = 0,500\,781\,2, \quad \tilde{x_2} = 0,500\,8,$$

potom

$$\left.\begin{aligned} |x_1 - \tilde{x_1}| = |\triangle x_1| = 0,278 \cdot 10^{-4} \leqq 0,5 \cdot 10^{-4}, \\ \left|\frac{\triangle x_1}{x_1}\right| \approx 0,554\,8 \cdot 10^{-3} \leqq 0,5 \cdot 10^{-3}, \\ |x_2 - \tilde{x_2}| = |\triangle x_2| = 0,188 \cdot 10^{-4} \leqq 0,5 \cdot 10 - 4, \\ \left|\frac{\triangle x_2}{x_2}\right| \approx 0,375\,4\,8 \cdot 10^{-4} \leqq 0,5 \cdot 10^{-4}. \end{aligned}\right\} \quad \text{čtyři platné číslice.}$$

Vypočteme rozdíl uvedených čísel:

$$v = x_1 - x_2 = 0,246\,6 \cdot 10^{-3}, \quad \tilde{v} = \tilde{x_1} - \tilde{x_2} = 0,2 \cdot 10^{-3},$$

$$|\triangle v| = |v - \tilde{v}| = 0,466 \cdot 10^{-4} \leqq 0,5 \cdot 10^{-4}$$

a pro mantisy platí

$$|a - \tilde{a}| = |0,246\,6 - 0.2| = 0,466 \cdot 10^{-1} \leqq 0,5 \cdot 10^{-1}.$$

Tedy $\tilde{v}$ má pouze jednu platnou číslici.

Vidíme také, že

$$\left|\frac{\triangle v}{v}\right| \approx 1,9 \cdot 10^{-1}, \quad \text{tj.} \left|\frac{\triangle v}{v}\right| \approx 3\,500 \cdot \left|\frac{\triangle x_1}{x_1}\right|, \quad \left|\frac{\triangle v}{v}\right| \approx 5\,035 \cdot \left|\frac{\triangle x_2}{x_2}\right|,$$

došlo tedy k velkému zesílení výstupní relativní chyby ve srovnání s relativními chybami vstupních dat.

Velmi poučný je následující příklad (viz též odst. 4.3): Kvadratická rovnice $x^2 - 56x + 1 = 0$ má kořeny $x_1 = 28 + \sqrt{783}$, $x_1 = 28 - \sqrt{783}$.

Vezmeme-li $\sqrt{783}$ na pět platných číslic (tj. s chybou menší než $0,5 \cdot \cdot 10^{-3}$) ($\sqrt{783} \doteq 27,982$) dostaneme [v $M(10,5)$]:

$$x_1 = 55,982, \quad x_2 = 0,018.$$

Absolutní chyba kořenů není větší než $0,000\,5$. Avšak pro relativní chyby platí

$$\left|\frac{\triangle x_1}{x_1}\right| \leqq 0,5 \cdot 10^{-5}; \quad \left|\frac{\triangle x_2}{x_2}\right| \leqq 0,8 \cdot 10^{-2}.$$

Kořen x_1 (resp. jeho aproximace) je tedy určen na pět platných číslic, kdežto kořen x_2 (jeho aproximace) pouze na dvě (dá se to přesně zjistit podle (3.2.4)].

Chceme-li přesněji vypočítat i aproximaci kořene x_2, máme v zásadě dvě možnosti:

(i) vzít $\sqrt{783}$ např. na deset platných číslic ($= 27,982\,137\,15$) - u počítače to znamená použít dvojnasobné aritmetiky (větší pracnost a spotřeba paměti!); potom

$$x_2 = 0,178\,63 \cdot 10^{-1}$$

má pět platných číslic (relativní chyba $\approx 10^{-5}$).

(ii) Kořen x_2 počítat jinou metodou, resp. algoritmem (větší chytrost!). Např. algoritmem A1 z odst. 1.3.1, tj.

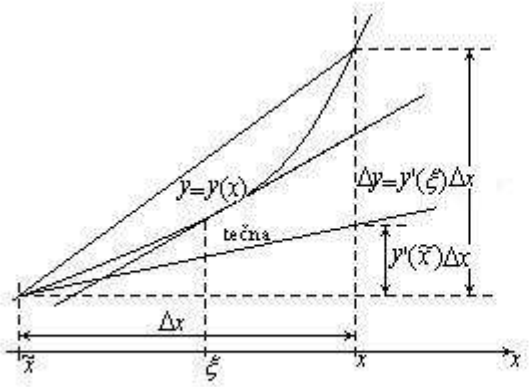
$$x_2 = \frac{1}{x_1} = \frac{1}{55,982} = 0,017\,862\,88\ldots .$$

Pět platných číslic výsledku máme zaručeno, pokud správně zaokrouhlíme.

3.2.9 Chyba funkční hodnoty.

Uvažujme funkci $y = y(x)$ jedné proměnné a nechť $\tilde{x}$ je aproximací x. Chceme odhadnout velikost chyby

$$\triangle y = y(x) - y(\tilde{x}) = y(\tilde{x} + \triangle x) - y(\tilde{x})$$



Obr. 1.

v závislosti na chybě argumentu $\triangle x = x - \tilde{x}$. Obvykle chybu $\triangle y$ nahrazujeme diferenciálem funkce $y = y(x)$ (viz obr. 1). To znamená, že lokálně nahradíme funkci y lineární funkcí argumentu $\triangle x$:

$$y(x) = y(\tilde{x}) + y'(\xi) \triangle x.$$

Funkce $|y'|$ může být interpretována jako míra citlivosti funkce y na poruchách v argumentu. Protože

$$|\triangle y| = |y'(\xi)|\,|\triangle x| \leqq M\,\varepsilon(\tilde{x}), \quad M = \sup|y'(z)|\,, \tag{3.2.6}$$

kde suprémum (maximum) hledáme v okolí bodu $\tilde{x}$ obsahujícím x. Pro relativní chybu funkční hodnoty máme odhad

$$\left|\frac{\triangle y}{y}\right| \leqq \frac{M\,\varepsilon(\tilde{x})}{|y|} \lesssim \frac{|\tilde{x}|\,M\,\delta(\tilde{x})}{|y|}. \tag{3.2.7}$$

Je-li $x = (\tilde{x}_1, \tilde{x}_2, ..., \tilde{x}_n)$ aproximace hodnoty $x = (x_1, x_2, ..., x_n)$ a je-li $y(x) = y\,(x_1, x_2, ..., x_n)$ funkce n proměnných, stanovíme odhad chyby funkční hodnoty analogicky, když označíme

$$\triangle y = y(x) - y(\tilde{x}) = y\,(x_1, x_2, ..., x_n) - y\,(\tilde{x}_1, \tilde{x}_2, ..., \tilde{x}_n) =$$

$$= y\,(\tilde{x}_1 + \triangle x_1, \tilde{x}_2 + \triangle x_2, ..., \tilde{x}_n + \triangle x_n) - y\,(\tilde{x}_1, \tilde{x}_2, ..., \tilde{x}_n)\,.$$

Podle věty o střední hodnotě máme

$$\triangle y = \sum_{k=1}^{n} \frac{\partial y\,(x_1, ..., x_{k-1}, \xi_k, x_{k+1}, ..., x_n)}{\partial x_k} \triangle x_k, \quad \triangle x_k = x_k - \tilde{x}_k,$$

$$\xi_k \text{ leží mezi } \tilde{x}_k \text{ a } x_k.$$

Proto

$$\triangle y \approx \sum_{k=1}^{n} \frac{\partial y\,(x_1, ..., x_{k-1}, \tilde{x}_k, x_{k+1}, ..., x_n)}{\partial x_k} \triangle x_k, \tag{3.2.8}$$

a tudíž

$$|\triangle y| \lesssim \sum_{k=1}^{n} \left|\frac{\partial y\,(x_1, ..., x_{k-1}, \tilde{x}_k, x_{k+1}, ..., x_n)}{\partial x_k}\right| |\triangle x_k|\,. \tag{3.2.9}$$

Pokud se derivace funkce y v okolí bodu $\tilde{x}$ dosti mění, užijeme místo (3.2.9) přesného odhadu (vyplývajícího z věty o střední hodnotě).

$$|\triangle y| \leqq \sum_{k=1}^{n} M_k\,|\triangle x_k|\,, \quad M_k = \sup\left|\frac{\partial y(x)}{\partial x_k}\right|, \tag{3.2.10}$$

kde suprémum (maximum) hledáme v okolí bodu $\tilde{x}$ obsahujícim bod x. Pro odhad relativní chyby dostáváme

$$\left|\frac{\triangle y}{y}\right| \leqq \sum_{k=1}^{n} \frac{M_k\,|x_k|}{y} \left|\frac{\triangle x_k}{x_k}\right|. \tag{3.2.11}$$

3.2.10 Příklady

3.3 Aproximace v normovaném prostoru

Jestliže aproximujeme nějaký prvek $x \in B_1$, kde B_1 je obecně nějaký lineární normovaný prostor, prvkem $\tilde{x} \in B_2$, kde $B_2 \subset B_1$, je *chyba* této *aproximace* opět prvkem prostoru B_1, tj. $x - \tilde{x} \in B_1$.Velikost této chyby pak posuzujeme pomocí normy v prostoru B_1.

Odhadem chyby rozumíme číslo $\varepsilon(\tilde{x})$, pro něž platí

$$\|x - \tilde{x}\| \leqq \varepsilon(\tilde{x}). \tag{3.3.1}$$

Relativní chybu aproximace $\tilde{x}$ definujeme pomocí norem jako číslo $\|x - \tilde{x}\| \,/\, \|x\|$.

Základní úlohou teorie aproximace je úloha najít k danému prvku $x \in B_1$ takový prvek $\tilde{x} \in B_2 \subset B_1$, aby chyba byla co nejmenší. Prostor B_2 se obvykle volí konečně dimenzionální.

V kapitole II budou B_1, B_2 vektorovými prostory konečné dimenze.

Pojmu aproximace ve funkcionálních prostorech (tj. aproximace funkcí) je věnována větší pozornost v publikaci [11]. Kromě toho řadu konkrétních poznatků najde čtenář také ve skriptech [8].

3.4 Cvičení

3.4.1

Určete aproximace čísel $\sqrt{2}$, $\sqrt{200}$ na 3, 4, 5, 6, 7 platných číslic.

3.4.2

Obsah trojúhelníka se vypočte podle vzorce $P = \frac{1}{2}ab\sin\gamma$, kde $\gamma \in \in (0, \pi/2)$ je úhel sevřený stranami a, b. Jestliže $\delta(\tilde{a})$, $\delta(\tilde{b})$, $\delta(\tilde{\gamma})$ jsou

odhady relativních chyb aproximací daných hodnot a, b, γ, ukažte, že platí odhad

$$\delta(\tilde{P}) \leqq \delta(\tilde{a}) + \delta(\tilde{b}) + \delta(\tilde{\gamma}).$$

3.4.3

Odhadněte absolutní chybu aproximace čísla u: a) $u = 3x + y - z$; b) $u = \frac{1}{2}xy$; c) $u = x\sin(y/40)$, když $x = 2,00 \pm 0,005$; $y = 3,00 \pm 0,005$; $z = 4,00 \pm 0,005$.

3.4.4

Určete $u = x^y$, když $x = 8,47 \pm 0,5 \cdot 10^{-2}$, $y = 0,643 \pm 0,5 \cdot 10^{-3}$.

3.4.5

Číslo $z = xy$ určete na tři platná desetinná místa, když: a) $x = 1,313\,4 \pm$ $\pm 0,5 \cdot 10^{-4}$, $y = \pi$; b) $x = \pi$, $y = e$.

[a) $z = 4,126$; b) $z = 8,540$. Návod. Užijte vztahů pro odhad relativních chyb.]

3.4.6

Určete a) $(0,643\,1)^4$; b) $\frac{2,489\,7 \cdot 1,980}{16,387}$. Uvažovaná čísla mají všechny napsané číslice platné.

[a) $0,171\,05 \pm 6 \cdot 10^{-5}$; b) $0,300\,8 \pm 10^{-4}$. Návod. Užijte funkcí x^4, xy/z.]

4 Podmíněnost úloh a algoritmů

4.1 Korektní a nekorektní úlohy.

Mezi matematickými úlohami, včetně úloh numerických, se setkáváme s úlohami, jejichž řešení jsou velmi citlivá na změny ve vstupních datech. Míníme tím ten fakt, že malé změny vstupních dat se projeví velkým "rozptylem" řešení. Proto je nutné v takových situacích podrobit hlubší analýze

závislost vstupních a výstupních dat. Rovněž algoritmy mohou být citlivé na změny ve vstupních datech a je proto nutné této problematice také věnovat pozornost i při volbě algoritmu pro řešení dané numerické úlohy.

Předpokládejme, že B_1 (množina vstupních dat) a B_2 (množina výstupních dat) jsou Banachovy prostory. Řekneme, že úloha

$$y = U(x), \quad x \in B_1, \quad y \in B_2,$$

je *korektní* na dvojici prostorů (B_1, B_2), když

1. ke každému $x \in B_1$ existuje jediné řešení $y \in B_2$,

2. toto řešení spojitě závisí na vstupních datech, tj. když $x_n \longrightarrow x$, $U(x_n) = y_n$, potom $U(x_n) \longrightarrow U(x) = y$.

Protože B_1, B_2 jsou Banachovy prostory, potom spojitá závislost řešení na vstupních datech se dá zaručit podmínkou

$$\|y_n - y\|_{B_2} \leqq \|x_n - x\|_{B_1}, \quad L \text{ je konstanta.} \tag{4.1.1}$$

Velkou třídu nekorektních úloh tvoří nejednoznačně řešitelné úlohy. Např. je to úloha určit některé nebo všechny prvky matice $\mathbf{A}$ v rovnici $\mathbf{Ax} = \mathbf{b}$, jsou-li dány vektory $\mathbf{x}$ a $\mathbf{b}$. Úlohu určit chybu vstupních dat, aby výstupní data v dané toleranci, lze také zařadit mezi nekorektní úlohy, nebot' má více než jedno řešení.

Někdy je nekorektnost úlohy (ve smyslu naši definice) "zaviněná" pouze nevhodnou - nekorektní formulací. Např. úloha určit všechny kořeny polynomu. Takové úlohy v praxi většinou za nekorektní nepovažujeme. Stačí totiž doplnit formulaci úlohy tak, aby byla splněna i podmínka jednoznačnosti. Např. určit největší reálný kořen polynomu apod.

4.2 Podmíněnost úloh

Budeme říkat, že korektní úloha je *dobře podmíněná*, jestliže malá změna ve vstupních datech vyvolá malou změnu řešení. Je-li $y + \triangle y$, resp. y řešení úlohy odpovídající vstupním datům $x + \triangle x$, resp. x, potom číslo

$$C_p = \frac{\frac{\|\triangle y\|}{\|y\|}}{\frac{\|\triangle x\|}{\|x\|}} = \frac{\text{relativní chyba výstupu}}{\text{relativní chyba vstupu}} \tag{4.2.1}$$

nazýváme *číslem podmíněnosti úlohy* $y = U(x)$. Protože většinou umíme stanovit pouze odhady relativních chyb, stanovíme číslo podmíněnosti přibližně

$$C_p \approx \frac{\delta(y)}{\delta(x)}.$$

Když $C_p \approx 1$, je úloha (velmi) *dobře podmíněná*. Pro velká C_p jde o úlohu *špatně podmíněnou*. Je patrné, že jsou to pojmy relativní. V praxi obvykle hovoříme o špatně podmíněné úloze, je-li $C_p \gtrsim 100$.

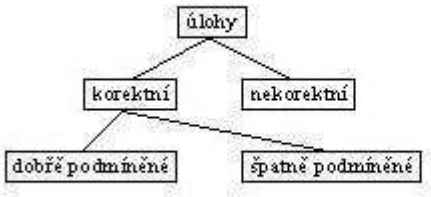


Schéma 3.

Uvažujeme např. polynom $p(x) = x^2 + x - 1\,150$. Snadno zjistíme, že $p(33) = = -28$ a $p(100/3) = -50/9 \approx -5,6$. Tedy změně $\triangle x = \frac{1}{3}$ na vstupu a odpovídána změna $\triangle y = 22,4$. Zde $C_p \approx 80$ a také $|\triangle y| < 70\,|\triangle x|$. Úloha je špatně podmíněná. Čtenář necht' si všimne, že jde o výpočet hodnoty polynomu v okolí kořene (33,415 335 76).

4.2.1 Příklad

Vypočtěme číslo podmíněnosti úlohy: stanovit funkční hodnotu (diferencovatelné) funkce $y = f(x)$, $x \in J$. Z věty o střední hodnotě dostaneme

$$|\triangle y| \approx |f'(x)|\,|\triangle x|\,.$$

Odtud

$$\left|\frac{\triangle y}{y}\right| \approx \left|\frac{x f'(x)}{f(x)}\right| \left|\frac{\triangle x}{x}\right|,$$

a tedy

$$C_p \approx \left|\frac{x f'(x)}{f(x)}\right|. \tag{4.2.2}$$

Pro funkci $u = f(x) = f(x_1, x_2, ..., x_n)$ proměnných máme analogicky

$$|\triangle u| \lesssim \sum_{i=1}^{n} \left|\frac{\partial f(x)}{\partial x_i}\right| |\triangle x_i|\,,$$

a tedy

$$\left|\frac{\triangle u}{u}\right| \lesssim \begin{cases} C_1 \sum_{i=1}^{n} \left|\frac{\triangle x_i}{x_i}\right|, & \text{kde} \quad C_1 = \max_i \left|\frac{x_i}{f(x)}\frac{\partial f(x)}{\partial x_i}\right|, \\ C_2 \max_i \left|\frac{\triangle x_i}{x_i}\right|, & \text{kde} \quad C_2 = \sum_{i=1}^{n} \left|\frac{x_i}{f(x)}\frac{\partial f(x)}{\partial x_i}\right|, \\ C_3 \sqrt{\sum_{i=1}^{n} \left|\frac{\triangle x_i}{x_i}\right|^2}, & \text{kde} \quad C_3 = \sqrt{\sum_{i=1}^{n} \left|\frac{x_i}{f(x)}\frac{\partial f(x)}{\partial x_i}\right|^2}. \end{cases} \tag{4.2.3}$$

Zde je vidět, že číslo podmíněnosti závisí na vybrané normě vektoru o složkách

$$\frac{x_i}{f(x)}\frac{\partial f(x)}{\partial x_i}.$$

Tato skutečnost nikterak nevadí, protože $|\triangle u/u|$ bude velké pro špatně podmíněnou úlohu bez ohledu na normu (zmíněná složka vektoru nemůže být v jedné normě velká a v jiné malá).

4.2.2 Příklad

Prověříme, že úloha stanovit $\sin x$ v okolí bodu 0 je dobře podmíněná a v okolí bodu π špatně podmíněná.

Volíme $x = 3,14$, $\triangle x = 0,01$. Potom [počítáme v $M(10,7)$]

$$\sin x = \sin 3,14 = 1,592\,599 \cdot 10^{-3},$$

$$\sin(x + \triangle x) = \sin 3,15 = -8,407\,366 \cdot 10^{-3},$$

$$\triangle y = \sin(x + \triangle x) - \sin x = -9,999\,965 \cdot 10^{-3},$$

$$\left|\frac{\triangle y}{y}\right| = 6,279\,022,$$

$$\left|\frac{\triangle x}{x}\right| = 3,184\,713 \cdot 10^{-3},$$

$$C_p \approx 1\,971,613 \approx 2\,000.$$

Velmi špatně podmíněná úloha!

Volme $x = -1,592\,653 \cdot 10^{-3}$, $\triangle x = 0,01$ (volíme zde x "stejně daleko" od 0, jako bylo daleko od π v předchozí volbě). Potom

$$\sin x = 1,592\,652 \cdot 10^{-3},$$

$$\sin(x+\triangle x) = \sin(8,407\,347\cdot 10^{-3}) = 8,407\,247\cdot 10^{-3},$$
$$\triangle y = -9,999\,899\cdot 10^{-3},$$
$$\left|\frac{\triangle y}{y}\right| = 6,278\,772,$$
$$\left|\frac{\triangle x}{x}\right| = 6,278\,831,$$
$$C_p \approx 0,999\,990\,5 \approx 1.$$

Velmi dobře podmíněná úloha!

Protože podle vzorce (4.2.2) je pro naši funkci

$$C_p \approx \left|\frac{x\cos x}{\sin x}\right|,$$

vidíme, že daná úloha bude špatně podmíněná v okolí těch bodů x_0, v nichž

$$\lim_{x\to x_0} \operatorname{cotg} x = \pm\infty.$$

4.2.3 Příklad

Vyšetřme, jaká bude změna řešení soustavy

$$x_1 + 0,99x_2 = 1,99,$$
$$o,99x_1 + 0,98x_2 = 1,97$$

při malé změně pravé strany, tj. jaká bude citlivost řešení k těmto změnám.

Řešení této soustavy je

$$\mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \end{pmatrix}.$$

Řešení soustavy s toutéž maticí, ale s pravou stranou

$$\tilde{\mathbf{b}} = \begin{pmatrix} 1,989\,903 \\ 1,970\,106 \end{pmatrix} \quad \text{je} \quad \tilde{\mathbf{x}} = \begin{pmatrix} \tilde{x}_1 \\ \tilde{x}_2 \end{pmatrix} = \begin{pmatrix} 3 \\ -1,021 \end{pmatrix}.$$

Chyba v řešení: $\|\mathbf{x}-\tilde{\mathbf{x}}\| = |x_1-\tilde{x}_1| + |x_2-\tilde{x}_2| = 4,021$; chyba v pravých stranách $||\mathbf{b}-\tilde{\mathbf{b}}|| = |b_1-\tilde{b_1}| + |b_2-\tilde{b_2}| = 2,03\cdot 10^{-4}$. Protože $\|\mathbf{x}\| = = |x_1| + |x_2| = 2,,\ ||\mathbf{b}|| = |b_1| + |b_2| = 3,96$, potom

$$C_p = \frac{\frac{\|\mathbf{x}-\tilde{\mathbf{x}}\|}{\|x\|}}{\frac{\|\mathbf{b}-\tilde{\mathbf{b}}\|}{\|\mathbf{b}\|}} \approx 39\,220$$

a daná numerická úloha

$$\mathbf{x} = \mathbf{U}(\mathbf{b}), \quad \text{kde} \quad \mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}, \mathbf{b} = \begin{pmatrix} b_1 \\ b_2 \end{pmatrix}$$

je velmi špatně podmíněná. Pozornému čtenáři jistě neuniklo, že špatná podmíněnost této úlohy (hovoříme též o špatně podmíněnosti soustavy rovnic) je "zaviněna" skutečnosti, že determinant soustavy je velmi malé číslo (-10^{-4}).

Poněkud podrobněji si takových soustav všimneme v kap. II této publikace (čl. 9).

4.3 Stabilita algoritmů.

Řekli jsme si v odst. 2.4, že při realizaci konkrétního numerického algoritmu na počítači se dopouštíme chyb v aritmetických operacích, nebot' počítač pracuje s čísly z množiny $M(q,t)$, i kdyby vstupní data byla dána přesně. Vzniká tedy opět problém, jak posoudit citlivost každého konkréthího algoritmu na zaokrouhlovací chyby včetně chyb ve vstupních datech.

Chceme-li se při výpočtech vyvarovat nesmyslných výsledků, musíme si vybírat algoritmy málo citlivé ke všem zmíněným vlivům. Takové algoritmy budeme nazývat *stabilní*. Aby algoritmus byl stabilní, musí být:

1. dobře podmíněný, tj. málo citlivý na poruchy v datech,

2. *numerický stabilní* (přesněji - numerická realizace algoritmu musí být numericky stabilní), tj. málo citlivý na vliv zaokrouhlovacích chyb během výpočtu na počítači.

Určité představy o stabilitě algoritmu nám poskytne metoda počítačových experimentů.

Uvažujme algoritmy výpočtu reálných kořenů kvadratické rovnice.

$$x^2 - 2bx + c = 0$$

podle vzorců

$$x_1 = b + \sqrt{b^2 - c}, \quad x_2 = b - \sqrt{b^2 - c}, \quad b^2 - c > 0.$$

Tyto algoritmy jsou uvedený v odst. 1.3.1; označili jsme je A1, A2.

(i) $\quad 2b = 56, \quad c = 1; \quad M(10,5): \quad x_1 = 55,982,$

$$x_2 = 0,018\,000, \qquad (A2)$$

$$x_3 = 0,017\,862, \qquad (A1)$$

$$M(10,8): \quad x_1 = 55,982\,137,$$
$$x_2 = 0,017\,863\,000, \qquad (A2)$$
$$x_3 = 0,017\,862\,800, \qquad (A1)$$

(ii) $2b = 10^5$, $c = 1$; $M(10,8): \quad x_1 = 100\,000,00,$
$$x_2 = 0, \qquad (A2)$$
$$x_3 = 0,000\,010\,000\,000, \qquad (A1)$$
$$M(10,11): \quad x_1 = 99\,999,999\,990,$$
$$x_2 = 0,000\,100\,000\,000\,01.$$

Vidíme, že algortmus A2 dává horší výsledky (sledujeme např. splnění rovnosti $x_1 x_2 = 1$).

Uvažujme jiný případ: Kořeny kvadratické rovnice

$$x^2 - 4x + 4 = 0$$

jsou $x_1 = x_2 = 2$. Kořeny rovnice s poněkud "porušeným" koeficientem

$$x^2 - 4x + 3,999\,9 = 0$$

jsou

$$x_1 = 2,01; \quad x_2 = 1,99.$$

Porucha $|c - \tilde{c}| = 10^{-4}$ v koeficientech (vstupních datech) vyvolala stokrát větší poruchu $|x_i - \tilde{x_i}| = 10^{-2}$ v kořenech (na výstupu). Tuto skutečnost neovlivníme výběrem algoritmu - jde o špatně podmíněnou úlohu. Porovnánim relativních chyb zjistíme, že číslo podmíněnosti této úlohy je $C_p = 200$.

Tento příklad nám naznačuje metodu, jak posuzovat podmíněnost algoritmu: Výsledek uvažovaného algoritmu můžeme v mnoha případech interpretovat jako přesné řešení stejné úlohy s poněkud změněnými vstupními daty. Chceme-li získat představu o stabilitě algoritmu (a přesnosti získaných výsledků), řešíme několik stejných úloh s poněkud pozměněnými daty a sledujeme, jak se poruchy ve vstupních datech projeví ve výsledku (na výstupu). Tomuto postupu říkáme *metoda experimentálních perturbací.*

Uvažujme algoritmus $\mathbf{y} = r(\mathbf{x})$. Je-li $p(\mathbf{x})$ porucha ve vstupních datech a $p(\mathbf{y})$ odpovídající porucha ve výsledku, potom číslo podmíněnosti algoritmu definujeme vztahem

$$\frac{\|p(\mathbf{y})\|}{\|y\|} = C_A \frac{\|p(\mathbf{x})\|}{\|\mathbf{x}\|}, \qquad (4.3.1)$$

kde C_A je právě tím *číslem podmíněnosti algoritmu.* Zde $\|.\|$ značí nějakou normu vstupních a výstupních dat.

4.3.1 Příklad

Úloha 1: Máme stanovit x ze soustavy

$$x + \alpha y = 1,$$
$$\alpha x + y = 0,$$

kde $\alpha \neq \pm 1$ je vstupní parametr.

Protože $x = 1/(1-\alpha^2)$, bude číslo podmíněnosti úlohy [podle (4.2.2)]

$$C_p \approx \left| \frac{\alpha x'(\alpha)}{x(\alpha)} \right| = \frac{2\alpha^2}{|1-\alpha^2|}.$$

Úloha bude špatně podmíněná pro $\alpha^2 \approx 1$ a bude velmi dobře podmíněná, pokud se α^2 bude dosti lišit od 1.

Úloha 2: Stanovit $z = x + y$, kde x, y jsou řešením výše uvedené soustavy. Vstupním parametrem je opět α.

Algoritmus 1: Sečtením obou rovnic dostaneme

$$z = \frac{1}{1+\alpha}.$$

Algoritmus 2: Vypočteme z rovnic nejdříve

$$x = \frac{1}{1-\alpha^2} \quad \text{a} \quad y = \frac{\alpha}{1-\alpha^2}$$

a potom sečteme

$$z = x + y = \frac{1}{1-\alpha^2} - \frac{\alpha}{1-\alpha^2} = \frac{1}{1+\alpha}.$$

Úloha 2 je pro $\alpha \approx 1$ dobře podmíněná, ovšem algoritmus 2 je špatně podmíněný pro tato α. Např. pro $\alpha = 0,999\,9$ [v $M(10,4)$] je $z = 1,000$ (podle A2) a $z = 0,500\,0$ (podle A1). Bude-li $\triangle\alpha = 0,000\,1$, dostaneme $\triangle z = 0,5$. Proto číslo podmíněnosti algoritmu A2 je $C_A \approx 10\,000$.

4.3.2 Příklad

Chceme-li vypočítat integrál

$$J_n = \int_0^1 \frac{x^n}{x+5} dx, \quad n = 0, 1, 2, \ldots .$$

Protože platí

$$\int_0^1 x^{n-1}dx = \int_0^1 \frac{x^{n-1}(x+5)}{x+5}dx = \int_0^1 \frac{x^n}{x+5}dx + 5\int_0^1 \frac{x^{n-1}}{x+5}dx,$$

dostáváme rekurentní vztah

$$\frac{1}{n} = J_n + 5J_{n-1}, \quad \text{resp.} \quad J_n = -5J_{n-1} + \frac{1}{n}.$$

Budeme integrály J_n počítat podle tohoto vztahu. Tedy

$$\begin{aligned}
J_0 &= \int_0^1 \frac{dx}{x+5} = \ln\frac{6}{5} && (\approx 0,182\,32...),\\
J_1 &= -5\ln\frac{6}{5} + 1 && (\approx 0,088\,392\,2...),\\
J_2 &= -5J_1 + \frac{1}{2} = 25\ln\frac{6}{5} - \frac{9}{2} && (\approx 0,058\,037\,5...),\\
J_3 &= -5J_2 + \frac{1}{3} = -125\ln\frac{6}{5} + \frac{137}{6} && (\approx 0,043\,146...),\\
J_4 &= -5J_3 + \frac{1}{4} = 625\ln\frac{6}{5} - \frac{1367}{12} && (\approx ,034\,27...)
\end{aligned}$$

Počítáme-li nyni v $M(10,3)$, pak

$$\begin{aligned}
&J_0 \approx 0,182,\\
&J_1 \approx -5\cdot 0,182 + 1 = -0,910 + 1,00 = 0,090\,0,\\
&J_2 \approx -5\cdot 0,090\,0 + \frac{1}{2} = 0,050\,0,\\
&J_3 \approx 0,083\,0,\\
&J_4 \approx -0,165, \qquad \text{(je absurní, že } J_4 < 0!)\\
&J_5 \approx 1,02, \qquad \text{zcela nesmyslné výsledky!}\\
&J_6 \approx -4,93,\\
&\quad \text{atd.}
\end{aligned}$$

Co je příčinou tak velkého růstu chyby? Při výpočtu J_0 jsme se dopustili chyby $\varepsilon_0 \approx 3,2\cdot 10^{-4}$ a každým krokem se nám tato chyba zvětšovala pětkrát, nehledě na další chyby vlivem zaokrouhlování. Takže už $\varepsilon_4 \approx 625\varepsilon_0 \approx 2\cdot 10^{-1}$ a v hodnotě $J_4 \approx -0,165$ už není ani jedna číslice platná.

Algoritmus, který vycházel z uvedené rekurentní formule, byl nestabilní. Vzniká otázka, zda např. můžeme vypočítat J_4 pomocí uvedené rekurentní

formule, aby všechny číslice výsledku byly platné. Možné to je, ale musíme užít jiného algoritmu. Přepíšeme-li rekurentní formuli na tvar

$$J_{n-1} = \frac{-J_n}{5} + \frac{1}{5n},$$

bude se chyba (nepřesnost) vstupující do každého kroku dělit číslem 5, pokud počítáme ve smyslu klesajicího n. Z odhadu

$$|J_n| \leqq \int_0^1 \left| \frac{x_n}{x+5} \right| dx \leqq \frac{1}{5} \int_0^1 x_n dx = \frac{1}{5(n+1)}$$

vyplývá, že $J_n \to 0$, pro $n \to \infty$. Položíme např. $J_{10} = 0$, potom [opět v $M(10,3)$]:

$$\begin{aligned} J_9 &= -\frac{J_{10}}{5} + \frac{1}{50} = 0,020\,0, \\ J_8 &= -\frac{J_9}{5} + \frac{1}{45} = 0,018\,2, \\ J_7 &= 0,021\,3, \\ J_6 &= 0,024\,2, \\ J_5 &= 0,028\,4, \\ J_4 &= 0,034\,3. \end{aligned}$$

Tento výsledek má všechna desetinná místa platná.

4.3.3 Příklad

Vezměme si příklad jiného druhu. Chceme užít řadu

$$e^x = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + ...$$

k výpočtu $e^{-5,5}$. Pro $x = -5,5$ dostáváme alternující číselnou řadu a sečteme-li prvních 25 členů, bude zbytek řady (tj. chyba aproximace) menší než $|5,5^{25}/26!| \approx 2,1 \cdot 10^{-7}$. Tedy [v $M(10,5)$ - simulováno na T1 59].

$$\begin{aligned} e^{-5,5} = {} & 1,000\,0 - 5,499\,9 + 15,124 - 27,726 + 38,122 - 41,933 + \\ & + 38,438 - 30,199 + 20,761 - 12,686 + 6,977\,1 - 3,498\,5 + \\ & + 1,598\,8 - 0,676\,40 + ... = 0,007\,518\,1. \end{aligned}$$

Protože přesný výsledek na pět platných číslic je $e^{-5,5} = 0,004\,086\,7$, nemá předchozí výsledek ani jednu platnou číslici.

Opět máme příklad nevhodného algoritmu. Zde je příčina v tom, že zaokrouhlovací chyby některých členů, např. $38,122$, jsou stejného řádu jako konečný výsledek. Tuto nepříjemnou vlastnost užitého algoritmu bychom zeslabili kdybychom počítali na větší počet platných číslic.

Stačí však použít algoritmu vycházejícího z formule

$$e^{-5,5} = \frac{1}{e^{5,5}} = \frac{1}{1 + 5,5 + 15,12 + 27,7 + ...}$$

a dostaneme výsledek $0,004\,088\,9$ v aritmetice $M(10,5)$ na tři platné číslice.

4.4 Automatická kontrola přesnosti.

V předchozích odstavcích jsme si ukázali že snaha získat věrohodné výsledky při výpočtech vede k dosti obtížné práci analyzovat výpočetní procesy z hlediska kontroly přesnosti.

Jedním z prostředků, jak tuto kontrolu provádět, je *intervalová analýza*. Její výhodou je, že ztrátu přesnosti vlivem šíření chyby může registrovat samotný počítač, nevýhodou, že to vede ke zpomalení výpočtů a že dostáváme maximální odhady chyb.

Hlavní myšlenka spočívá v následujícím: Každé číslo, které vstupuje v libovolné fázi do algoritmu, nahradíme intervalem strojových čísel, tj. čísel z $M(q,t)$. který s jistotou obsahuje toto číslo. Celý algoritmus se potom provádí s těmito intervaly a výsledek celého výpočtu dostaneme opět jako interval.

Přistupme k přesné definici.

Necht' $\mathscr{M} : \mathbf{R} \to M \times M$ je zobrazení, které každému číslu $z \in \mathbf{R}$ přiřazuje dvojici čísel $\zeta_1, \zeta_2 \in M(q,t)$ takových, že platí

$$z \in \langle \zeta_1, \zeta_2 \rangle, \quad \text{tj.} \quad \zeta_1 \leqq z \leqq \zeta_2. \tag{4.4.1}$$

Tímto zobrazením je tedy k číslu $z \in \mathbf{R}$ určen interval $Z = \langle \zeta_1, \zeta_2 \rangle$. Říkáme také, že interval $Z = \langle \zeta_1, \zeta_2 \rangle$ je obrazem čísla $z \in \mathbf{R}$ v zobrazení $\mathscr{M}$ a píšeme $\mathscr{M}(z) = Z$.

Jestliže zkratkou op označíme kteroukoliv z operací $+$, -, $\cdot$, : s reálnými čísly, pak $\bigcirc$ bude značit odpovídající operaci s obrazy těchto čísel v zobrazení $\mathscr{M}$.

Necht'

$$\mathscr{M}(x) = X \equiv \langle \zeta_1, \zeta_2 \rangle, \quad x \in \mathbf{R}, \quad \zeta_1, \zeta_2 \in M,$$
$$\mathscr{M}(y) = Y \equiv \langle \eta_1, \eta_2 \rangle, \quad y \in \mathbf{R}, \quad \eta_1, \eta_2 \in M.$$

Je-li $z = x \,\mathrm{op}\, y$, potom definujeme

$$\mathscr{M}(z) = Z = X \bigcirc Y = \bigcup_{\zeta \in X, \eta \in Y} \mathscr{M}(\zeta \,\mathrm{op}\, \eta). \tag{4.4.2}$$

Z této definice vyplývá, že interval $Z = X \bigcirc Y$ je nejmenším strojovým číselným intervalem, který obsahuje všechny možné výsledky exaktních operaci s čísly $\zeta \in X$, $\eta \in Y$. Pochopitelně pro dělení musí být $0 \notin Y$. Je-li dána spojitá funkce $y = f(x)$, $x \in X$, potom definujeme

$$Y = f(x) = \bigcup_{\zeta \in X} \mathscr{M}(f(\zeta)) = \left\langle \min_{x \in X} f(x), \max_{x \in X} f(x) \right\rangle. \tag{4.4.3}$$

4.4.1 Příklad

4.4.2 Příklad

4.4.3 Příklad

4.5 Cvičení

4.5.1

4.5.2

4.5.3

4.5.4

4.5.5

II. Metody lineární algebry

5 Základní pojmy lineární algebry

5.1 Matice a vektory.

Matici $\mathbf{A}$ *typu* (m,n) (m, n jsou přirozená čísla) rozumíme množinu mn reálných nebo komplexních čísel seřazených do schématu

$$\mathbf{A} = \begin{pmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \dots & \dots & \dots & \dots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{pmatrix}$$

a stručně zapisujeme $\mathbf{A} = (a_{ij})$, $i = 1,2,...,m$; $j = 1,2,...,n$. Říkáme také, že v *pozici* (i,j) je prvek a_{ij}.

Dvě matice $\mathbf{A} = (a_{ij})$ a $\mathbf{B} = (b_{ij})$ téhož typu jsou si *rovny*, když $a_{ij} = b_{ij}$ pro všechna i a j. Píšeme $\mathbf{A} = \mathbf{B}$.

Transponovaná matice $\mathbf{A}^T$ k matici $\mathbf{A}$ má v pozici (i,j) prvek a_{ji} matice $\mathbf{A}$.

Hermitovsky transponovaná matice $\mathbf{A}^H$ k matici $\mathbf{A}$ má v pozici (i,j) prvek $\bar{a}_{ji}$ což je číslo komplexně sdružené k číslu a_{ji}. Pro reálnou matici (a_{ij} jsou reálná čísla) je $\mathbf{A}^H = \mathbf{A}^T$. Aby nedošlo k nedorozumění, budeme horní index T užívat pouze pro transpozici reálné matice.

Matici typu $(m,1)$ nazýváme *sloupcovým vektorem* a značíme

$$\mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_m \end{pmatrix} = (x_1, x_2, ..., x_m)^T .$$

Matici typu $(1,n)$ nazýváme *řádkovým vektorem* a značíme

$$\mathbf{y}^T = (y_1, y_2, ..., y_n) .$$

Řádky a sloupce matice $\mathbf{A}$ označujeme

$$\mathbf{r}_i(\mathbf{A}) = (a_{i1}, a_{i2}, ..., a_{in}), \quad i = 1,2,...,m,$$
$$\mathbf{s}_j(\mathbf{A}) = (a_{1j}, a_{2j}, ..., a_{mj})^T, \quad j = 1,2,...,n.$$

Vztah transpozice lze tedy stručně zapsat jako

$$\mathbf{r}_i(\mathbf{A}^H) = \bar{\mathbf{s}}_i(\mathbf{A}), \quad \text{resp.} \quad \mathbf{s}_i(\mathbf{A}^H) = \bar{\mathbf{r}}_i(\mathbf{A}),$$

Násobek matice $\mathbf{A}$ *číslem* α je matice s prvky αa_{ij} a značíme ji $\alpha\mathbf{A}$.

Součet dvou matic (téhož typu) $\mathbf{A}$ a $\mathbf{B}$ je matice s prvky $a_{ij} + b_{ij}$ a značíme ji $\mathbf{A} + \mathbf{B}$.

Součin matice $\mathbf{A}$ typu (m, r) *s maticí* $\mathbf{B}$ typu (r, n) je matice $\mathbf{C} = \mathbf{AB}$ typu (m, n) s prvky

$$c_{ij} = \sum_{k=1}^{r} a_{ik} b_{kj}.$$

Říkáme, že $\mathbf{A}$ násobí $\mathbf{B}$ zleva nebo $\mathbf{B}$ násobí $\mathbf{A}$ zprava.

Pro operace s maticemi platí (nebereme-li v úvahu zaokrouhlovací chyby):

$$\mathbf{A} + \mathbf{B} = \mathbf{B} + \mathbf{A},$$

$$\mathbf{A} + (\mathbf{B} + \mathbf{C}) = (\mathbf{A} + \mathbf{B}) + \mathbf{C},$$

$$\mathbf{A}(\mathbf{BC}) = (\mathbf{AB})\mathbf{C},$$

$$(\mathbf{A} + \mathbf{B})\mathbf{C} = \mathbf{AC} + \mathbf{BC},$$

$$\mathbf{A}(\mathbf{B} + \mathbf{C}) = \mathbf{AB} + \mathbf{AC},$$

$$(\mathbf{AB})^H = \mathbf{B}^H \mathbf{A}^H,$$

pokud jsou ovšem uvedené operace definovány. Násobeni není obecně komutativní, tj. existují matice $\mathbf{A}$, $\mathbf{B}$, pro něž $\mathbf{AB} \neq \mathbf{BA}$.

Poznamenejme, že u uvedených maticových rovností nemusí být počet aritmetických operací (s čísly) potřebných k realizaci výrazu na levé straně roven počtu aritmetických operací potřebných na straně pravé.

Čtvercová matice řádu n je matice typu (n, n). Maticím typu (m, n), kde $m \neq n$, se proto říká *obdélníkové*.

Diagonální matice je taková čtvercová matice, pro níž $a_{ij} = 0$, když $i \neq j$. Diagonální matice budeme obvykle značit

$$\mathbf{D} = \operatorname{diag}(a_{11}, a_{22}, ..., a_{nn}) \quad \text{nebo} \quad \mathbf{D} = \operatorname{diag}(a_{ii}).$$

Jednotkovou matici definujeme jako diagonální matici s jedničkami na diagonále, tj.

$$\mathbf{I} = \operatorname{diag}(1, 1, ..., 1), \quad \text{resp.} \quad \mathbf{I} = \operatorname{diag}(1).$$

Čtenář si lehce dokáže, že pro libovolnou čtvercovou matici $\mathbf{A}$ platí

$$\mathbf{AI} = \mathbf{IA} = \mathbf{A}.$$

Nulová matice [obecně typu (m,n)] je taková matice, jejíž všechny prvky jsou nulové. Z definice součtu a součinu matic přímo plynou rovnosti

$$\mathbf{A}+\mathbf{0}=\mathbf{A};\quad \mathbf{A0}=\mathbf{0};\quad \mathbf{0A}=\mathbf{0}.$$

[Uvědomte si však, že v těchto rovnostech jsou nulové matice různých typů, pokud matice $\mathbf{A}$ je typu (m,n) a $m\neq n$].

5.1.1 Regulární a singulární matice.

V lineární algebře (viz např. [9], [15]) se čtenář seznámil s pojmem *determinantu čtvercové mative* $\mathbf{A}$. Je to číslo, které je jednoznačným způsobem přiřazeno matici - označujeme jej $\det\mathbf{A}$. Lze jej definovat induktivním způsobem:

$$\det(a_{11})=a_{11};\quad \det\begin{pmatrix}a_{11}&a_{12}\\a_{21}&a_{22}\end{pmatrix}=a_{11}a_{22}-a_{21}a_{12};$$

pro maticí řádu n (tzv. rozvoj determinantu podle prvků i-tého řádku, resp. j-tého sloupce):

$$\det\mathbf{A}=\sum_{i=1}^{n}(-1)^{i+j}a_{ij}\det\mathbf{S}_{ij},$$

resp.

$$\det\mathbf{A}=\sum_{i=1}^{n}(-1)^{i+j}a_{ij}\det\mathbf{S}_{ij},$$

kde $\mathbf{S}_{ij}$ je matice řádu $n-1$, kterou dostaneme z matice $\mathbf{A}$ vynecháním i-tého řádku a j-tého sloupce.

Z uvedené definice bezprostředně vyplývají následující tvrzení:

$$\begin{aligned}&\det\mathbf{0}=0,\\&\det\mathbf{I}=1,\\&\det\mathbf{D}=a_{11}a_{22}...a_{nn}\quad\text{pro}\quad\mathbf{D}=\operatorname{diag}(a_{ii}),\\&\det\mathbf{A}^T=\det\mathbf{A},\quad\det\mathbf{A}^H=\overline{\det\mathbf{A}},\\&\det\mathbf{AB}=\det\mathbf{A}\det\mathbf{B}.\end{aligned}$$

Čtvercová matice $\mathbf{A}$ je *singulární*, je-li $\det\mathbf{A}=0$, a je *regulární*, je-li $\det\mathbf{A}\neq 0$.

K regulární matici $\mathbf{A}$ existuje jediná matice $\mathbf{X}$ stejného řádu, pro kterou platí

$$\mathbf{AX}=\mathbf{XA}=\mathbf{I}.$$

Takové matici říkáme *matice inverzní* k matici $\mathbf{A}$ a značíme ji $\mathbf{A}^{-1}$.

Pro regulární matice $\mathbf{A}$, $\mathbf{B}$ platí

$$(\mathbf{AB})^{-1} = \mathbf{B}^{-1}\mathbf{A}^{-1},$$

$$\det \mathbf{A}^{-1} = (\det \mathbf{A})^{-1}.$$

5.1.2 Ortogonalita vektorů a matic

Skalárním součinem vektorů $\mathbf{x}$, $\mathbf{y}$ rozumíme číslo

$$x_1\bar{y}_1 + x_2\bar{y}_2 + ... + x_n\bar{y}_n.$$

Číslo

$$\bar{x}_1 y_1 + \bar{x}_2 y_2 + ... + \bar{x}_n y_n.$$

je skalárím součinem vektorů $\mathbf{y}$, $\mathbf{x}$ (v tomto pořadí).

Skalární součin vektorů budeme zapisovat (ve smyslu násobení matic)

$$\mathbf{x}^T\mathbf{y} = \mathbf{y}^T\mathbf{x} = \sum_{k=1}^{n} x_k y_k \quad \text{pro reálné vektory,}$$

$$\left.\begin{aligned} \mathbf{y}^H\mathbf{x} &= \sum_{k=1}^{n} x_k\bar{y}_k, \\ \mathbf{x}^H\mathbf{y} &= \sum_{k=1}^{n} \bar{x}_k y_k \end{aligned}\right\} \quad \text{pro komplexní vektory.}$$

[Pozor, $\mathbf{x}\mathbf{y}^T$ je matice typu (n, n) !]

Platí-li $\mathbf{x}^T\mathbf{y} = 0$, říkáme, že *vektory* $\mathbf{x}$, $\mathbf{y}$ jsou *ortogonální*.

Reálná *matice* $\mathbf{B}$ se nazývá *ortogonální*, jsou-li její sloupce *ortonormální* vektory, tj.

$$\mathbf{s}_i^T(\mathbf{B})\mathbf{s}_j(\mathbf{B}) = \begin{cases} 1 & \text{pro} \quad i = j, \\ 0 & \text{pro} \quad i \neq j, \end{cases}$$

což lze zapsat podmínkou

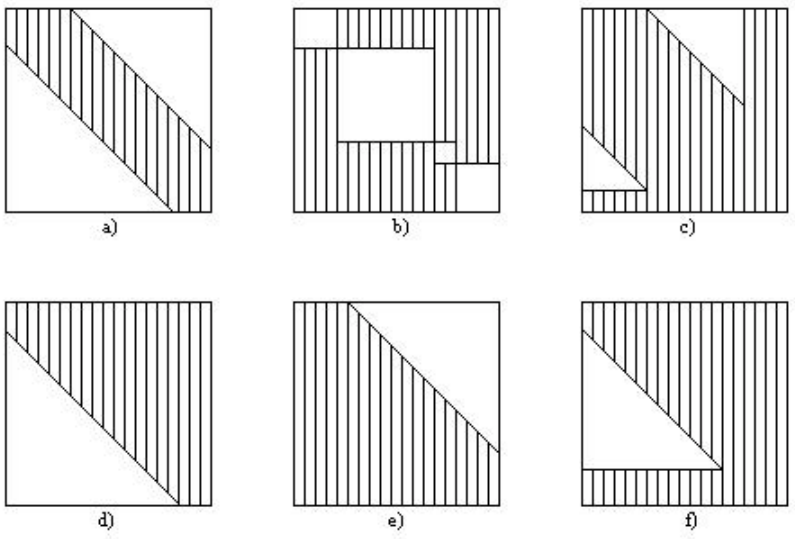
$$\mathbf{B}^T\mathbf{B} = \mathbf{I}, \quad \text{resp.} \quad \mathbf{B}^{-1} = \mathbf{B}^T.$$

Komplexní matice $\mathbf{B}$, pro niž $\mathbf{B}^{-1} = \mathbf{B}^H$, se nazývá *unitární*.

5.1.3 Řídké matice

Při výpočtech se často vyplatí brát v úvahu (z hlediska paměti počítače nebo z hlediska počtu operací) strukturu rozložení nulových (resp. nenulových) prvků. Maticím, které mají větší počet nulových prvků, říkáme *řídké*. V opačném případě hovoříme o *plných* nebo *obecných maticích*. Speciálním případem řídkých matic jsou *matice pásové* (viz odst. 6.6.1).

Na obr. 2 máme nakreslené některé typy řídkých matic. Šrafováním vyznačeny oblasti výskytu nenulových prvků.



Obr. 2

5.2 Normy matic a vektorů

Čtvercové matici $\mathbf{A}$ přiřadíme číslo $\|A\|$, které bude v jistém smyslu mírou její velikosti. Tomuto číslu říkáme *norma matice* $\mathbf{A}$.

Existuje řada možností, jak normu matice definovat. V dalším textu budeme užívat následující konkrétní normy:

$$\|\mathbf{A}\|_{\mathbf{R}} = \max_i \sum_j |a_{ij}| \qquad \text{(maximální řádkovy součet)}, \tag{5.2.1}$$

$$\|\mathbf{A}\|_{\mathbf{S}} = \max_j \sum_i |a_{ij}| \qquad \text{(maximální sloupcový součet)}, \tag{5.2.2}$$

$$\|\mathbf{A}\|_{\mathbf{E}} = \sqrt{\sum_i \sum_j |a_{ij}|^2} \qquad \text{(euklidovská norma)}. \tag{5.2.3}$$

Takto definované normy splňují následující podmínky (pokud píšeme znak normy bez indexu, máme na mysli kteroukoliv z definovaných norem):

(5.2.4) $$\|\mathbf{A}\| \geqq 0, \quad \|\mathbf{A}\| = 0, \quad \text{právě když } \mathbf{A} \text{ je nulová matice.}$$

$$\|\alpha\mathbf{A}\| = |\alpha| \, \|\mathbf{A}\|, \quad \alpha \text{ je libovolné číslo,}$$

$$\|\mathbf{A} + \mathbf{B}\| \leqq \|\mathbf{A}\| + \|\mathbf{B}\| \quad \text{(trojuhelníková nerovnost),}$$

$$\|\mathbf{AB}\| \leqq \|\mathbf{A}\| \, \|\mathbf{B}\| .$$

Protože vektor $\mathbf{x}$ chápme jako matici typu $(n, 1)$ dostáváme z definic (5.2.1) až (5.2.3) odpovídající normy vektorů

(5.2.5) $$\|\mathbf{x}\|_R = \max_i |x_i| \qquad \text{(řádková norma),}$$

(5.2.6) $$\|\mathbf{x}\|_S = \sum_i |x_i| \qquad \text{(sloupcová norma),}$$

(5.2.7) $$\|\mathbf{x}\|_E = \sqrt{\sum_i |x_i|^2} \qquad \text{(euklidovská norma).}$$

Pro odpovídající si normy vektorů a matic platí

(5.2.8) $$\|\mathbf{Ax}\| \leqq \|\mathbf{A}\| \, \|\mathbf{x}\| \quad .$$

5.2.1 Příklad

Stanovme normu matice

$$\mathbf{A} = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}$$

a vektorů

$$\mathbf{x} = (1, -2, 3)^T; \qquad \mathbf{y} = (0, 2, 3)^T.$$

$\|\mathbf{A}\|_{\mathbf{R}} = \max(|1| + |2| + |3|, |4| + |5| + |6|, |7| + |8| + |9|) = \max(6, 15, 24) = 24;$
$\|\mathbf{A}\|_{\mathbf{S}} = \max(|1| + |4| + |7|, |2| + |5| + |8|, |3| + |6| + |9|) = \max(12, 15, 18) = 18;$
$\|\mathbf{A}\|_{\mathbf{E}} = (1^2 + 2^2 + 3^2 + 4^2 + 5^2 + 6^2 + 7^2 + 8^2 + 9^2)^{1/2} = \sqrt{285} \doteq 16,88.$

$$\|\mathbf{x}\|_{\mathbf{R}} = \max(|1|, |-2|, |3|) = 3 \quad , \quad \|\mathbf{y}\|_{\mathbf{R}} = 3;$$
$$\|\mathbf{x}\|_{\mathbf{S}} = |1| + |-2| + |3| = 6, \qquad \|\mathbf{y}\|_{\mathbf{S}} = 5;$$
$$\|\mathbf{x}\|_{\mathbf{E}} = \sqrt{1^2 + (-2)^2 + 3^2} = \sqrt{14}, \quad \|\mathbf{y}\|_{\mathbf{E}} = \sqrt{13};$$

Vidíme, že ze skutečnosti $\|\mathbf{x}\|_{\mathbf{R}} = \|\mathbf{y}\|_{\mathbf{R}}$ neplyne $\mathbf{x} = \mathbf{y}$!

5.2.2 Konvergence posloupnosti vektorů

Říkáme, že posloupnost vektorů $\left\{\mathbf{x}^{\{k\}}\right\}_{k=1}^{\infty}$, $\mathbf{x}^{(k)} = (x_1^{(k)}, x_2^{(k)}, ..., x_n^{(k)})^T$, *konverguje* k vektoru $\mathbf{x} = (x_1, x_2, ..., x_k)^T$, tj. $\lim\limits_{k\to\infty} \mathbf{x}^{(k)} = \mathbf{x}$, jestliže

$$\lim_{k\to\infty} x_i^{(k)} = x_i, \quad i = 1, 2, ..., n. \tag{5.2.9}$$

Dá se dokázat (ne zcela triviálním způsobem), že podmínka (5.2.9) ke ekvivalentní podmínce

$$\lim_{k\to\infty} \left\|\mathbf{x}^{(k)} - \mathbf{x}\right\| = 0 \tag{5.2.10}$$

pro kteroukoliv z uvedených norem.

Poznamenejme, že podmínka $\lim\limits_{k\to\infty} \left\|\mathbf{x}^{(k)}\right\| = \|\mathbf{x}\|$ nezaručuje konvergenci $\mathbf{x}^{(k)} \to \mathbf{x}$. Např. posloupnost vektorů $(1,-1)^T$, $(1,1)^T$, $(1,-1)^T$, $(1,1)^T$, $(1,-1)^T$ atd. je divergentní (posloupnst složek je divergentní), avšak normy těchto vektorů konvergují k normě vektoru $(1,1)^T$.

5.2.3 Příklad

Prověřme platnost vztahu (5.2.8) pro definované (konkrétní) typy norem. Volme

$$\mathbf{A} = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix}, \qquad \mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}.$$

Potom

$$\mathbf{Ax} = \begin{pmatrix} a_{11}x_1 + a_{12}x_2 \\ a_{21}x_1 + a_{22}x_2 \end{pmatrix} = \mathbf{y}.$$

Označíme $\mathbf{y} = (y_1, y_2)^T$. Podle (5.2.5) je

$$\|\mathbf{y}\|_{\mathbf{R}} = \|\mathbf{Ax}\|_{\mathbf{R}} = \max(|a_{11}x_1 + a_{12}x_2|, |a_{21}x_1 + a_{22}x_2|),$$

$$\|\mathbf{x}\|_{\mathbf{R}} = \max(|x_1|, |x_2|).$$

Užitím vlastností absolutní hodnoty čísla dostaneme odhady:

$$|y_1| = |a_{11}x_1 + a_{12}x_2| \leqq |a_{11}||x_1| + |a_{12}||x_2| \leqq \max(|x_1|, |x_2|)(|a_{11}| + |a_{12}|),$$
$$|y_2| = |a_{21}x_1 + a_{22}x_2| \leqq |a_{21}||x_1| + |a_{22}||x_2| \leqq \max(|x_1|, |x_2|)(|a_{21}| + |a_{22}|).$$

Proto

$$\|\mathbf{y}\|_{\mathbf{R}} = \max(|y_1|, |y_2|) \leqq \max(|x_1|, |x_2|) \max(|a_{11}| + |a_{12}|, |a_{21}| + |a_{22}|),$$

tj. [podle (5.2.1)]

$$\|\mathbf{y}\|_{\mathbf{R}} \leqq \|\mathbf{x}\|_{\mathbf{R}} \|\mathbf{A}\|_{\mathbf{R}} .$$

Podle (5.2.6) je

$$\|\mathbf{y}\|_{\mathbf{S}} = \|\mathbf{A}\mathbf{x}\|_{\mathbf{S}} = |y_1| + |y_2| = |a_{11}x_1 + a_{12}x_2| + |a_{21}x_1 + a_{22}x_2|,$$
$$\|\mathbf{x}\|_{\mathbf{S}} = |x_1| + |x_2|.$$

Proto platí [s použitím (5.2.2)]

$$\begin{aligned} |y_1| + |y_2| &\leqq |a_{11}||x_1| + |a_{12}||x_2| + |a_{21}||x_1| + |a_{22}||x_2| = \\ &= (|a_{11}| + |a_{21}|)|x_1| + (|a_{12}| + |a_{22}|)|x_2| \leqq \\ &\leqq \max(|a_{11}| + |a_{21}|, |a_{12}| + |a_{22}|)(|x_1| + |x_2|) = \|\mathbf{A}\|_{\mathbf{S}} \|\mathbf{x}\|_{\mathbf{S}} . \end{aligned}$$

Podle (5.2.7)

$$\|\mathbf{y}\|_{\mathbf{E}} = \|\mathbf{A}\mathbf{x}\|_{\mathbf{E}} = \sqrt{|y_1|^2 + |y_2|^2} = (|a_{11}x_1 + a_{12}x_2|^2 + |a_{21}x_1 + a_{22}x_2|^2)^{1/2},$$
$$\|\mathbf{x}\|_{\mathbf{E}} = \sqrt{|x_1|^2 + |x_2|^2}.$$

Z nerovnosti

$$\begin{aligned} |y_1|^2 + |y_2|^2 \leqq{}& |a_{11}|^2|x_1|^2 + 2|a_{11}||a_{12}||x_1||x_2| + |a_{12}|^2|x_2|^2 + |a_{21}|^2|x_1|^2 + \\ &+ 2|a_{21}||a_{22}||x_1||x_2| + |a_{22}|^2|x_2|^2 \end{aligned}$$

použitím zřejmých nerovností[9]) dostáváme

$$\begin{aligned} |y_1|^2 + |y_2|^2 \leqq{}& |x_1|^2(|a_{11}|^2 + |a_{12}|^2 + |a_{21}|^2 + |a_{22}|^2) + \\ &+ |x_2|^2(|a_{11}|^2 + |a_{12}|^2 + |a_{21}|^2 + |a_{22}|^2). \end{aligned}$$

Odtud

$$|y_1|^2 + |y_2|^2 \leqq \|\mathbf{A}\|_{\mathbf{E}}^2 (|x_1|^2 + |x_2|^2), \quad \text{tj.} \quad \|\mathbf{y}\|_{\mathbf{E}}^2 \leqq \|\mathbf{A}\|_{\mathbf{E}}^2 \|\mathbf{x}\|_{\mathbf{E}}^2$$

nebot'

$$\|\mathbf{A}\|_{\mathbf{E}}^2 = (|a_{11}|^2 + |a_{12}|^2 + |a_{21}|^2 + |a_{22}|^2).$$

5.2.4 Poznámka.

V předcházejících odstavcích jsme hovořili o nulových maticích a vektorech, o singulárních maticích, o normách rovných nule atd. Z numerického hlediska "nulou" musíme rozumět množinu všech čísel, která se zobrazí do množiny $M(q, t, m_1, m_2)$ jako "strojová nula", např. čísla q^{-r}, kde $r < m_1$. Proto např. některé teoreticky regulární matice jsou z hlediska počítače neodlišitelné od matic singulárních.

---

[9]) $2|a_{11}||a_{12}||x_1||x_2| \leqq |a_{11}|^2|x_1|^2 + |a_{11}|^2|x_2|^2$,
$2|a_{21}||a_{22}||x_1||x_2| \leqq |a_{22}|^2|x_1|^2 + |a_{21}|^2|x_2|^2$,

5.3 Základní numerické úlohy lineární algebry

5.3.1 Řešení soustavy lineárních rovnic.

Chceme stanovit n-tici čísel $x_1, x_2, ..., x_n$ tak, aby

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + ... + a_{1n}x_n &= a_{1,n+1}, \\ a_{21}x_1 + a_{22}x_2 + ... + a_{2n}x_n &= a_{2,n+1}, \\ &\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \\ a_{n1}x_1 + a_{n2}x_2 + ... + a_{nn}x_n &= a_{n,n+1}. \end{aligned} \tag{5.3.1}$$

kde a_{ij}, $i = 1, 2, ..., n$; $j = 1, 2, ..., n+1$, jsou daná čísla. Čtvercovou matici $\mathbf{A} = (a_{ij})$, $i, j = 1, 2, ..., n$, nazýváme *maticí soustavy* a sloupcový vektor $\mathbf{b} = (a_{1,n+1}, a_{n,n+1}, ..., a_{n,n+1})^T$ *pravou stranou*. Jsou to vstupní data dané úlohy.

Maticově vektorovou symbolikou můžeme danou soustavu psát ve tvaru

$$\mathbf{Ax} = \mathbf{b}. \tag{5.3.2}$$

Je-li matice $\mathbf{A}$ regulární, existuje k danému $\mathbf{b}$ jediné řešení

$$\mathbf{x}_t = \mathbf{a}^{-1}\mathbf{b}. \tag{5.3.3}$$

Tomuto řešení budeme říkat *teoretické*, neboli *přesné řešení*. V dalších odstavcích se seznámíme s řadou metod a algoritmů, umožňujících stanovit řešení soustavy (5.3.1). Při konkrétním výpočtu na počítači nikdy nedostaneme $\mathbf{x}_t$ (s výjimkou zcela triviálních případů), ale pouze *výpočtené řešení* $\mathbf{x}_c$, které se vždy bude od $\mathbf{x}_t$ více či méně lišit.

V praxi se často vyskytují úlohy vedoucí na soustavy lineárních rovnic, které mají řádově stovky neznámých. Je zřejmé, že řešení takových úloh bude klást velké požadavky na rychlost a kapacitu paměti počítače. Proto se efektivnost metody či algoritmu posuzuje podle tří hlavních kritérií:

(i) jak je algoritmus rychlý, tj. kolik vyžaduje aritmetických operací,

(ii) jaké nároky klade algoritmus na pamět' počítače,

(iii) jaká je přesnost vypočteného řešení.

Z hlediska rychlosti výpočtu je např. tzv. Cramerovo pravidlo neefiktivní metodou. Pokud bychom navíc chtěli detreminanty počítat metodou rozvoje podle prvků jedné řady, pak pro větší n je tato metoda na současných počítačích nerealizovatelná. Pro výpočet n- řadového determinantu tímto způsobem je totiž zapotřebí $(n-1)n!$ násobení a zhruba $n!$ sčítání. K realizaci Cramerova pravidla musíme vypočítat $n+1$ determinantů, což představuje

celkem $(n+1)n!$ operací. Máme-li řešit např. soustavu o 30 rovnicích pro 30 neznámých na počítači, který vykoná 10^6 operací za sekundu, potom k vykonání $31 \cdot 30 \cdot 30! \approx 2,5 \cdot 10^{35}$ operací potřebuje $2,5 \cdot 10^{29}$ sekund ($\approx 7 \cdot 10^{25}$ hodin $\approx 2,9 \cdot 10^{24}$ dní $\approx 8 \cdot 10^{21}$ let).

Optimální metoda (nejrychlejší, nejpřesnější a s nejmenšími nároky na pamět' počítače) pro řešení všech možných soustav lineárních rovnic neexistuje. Proto si uvedeme celou řadu metod a jejich modifikací a současně si popíšeme jejich přednosti (či nevýhody) pro konkrétnější třídy lineárních soustav.

5.3.2 Řešení maticové rovnice.

Jsou dány matice $\mathbf{A}$ typu (m,n), $\mathbf{B}$ typu (m,r). Má se určit matice $\mathbf{X}$ typu (n,r) taková, aby platilo

$$\mathbf{AX} = \mathbf{B} \tag{5.3.4}$$

Rovnici $\mathbf{YA} = \mathbf{B}$ lze převést na (5.3.4) transportováním (odst. 5.1), tj.

$$\mathbf{A}^T\mathbf{Y}^T = \mathbf{B}^T.$$

Ve složkách lze rovnice (5.3.4) psát takto:

$$\begin{pmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \dots & \dots & \dots & \dots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{pmatrix} \begin{pmatrix} x_{11} & x_{12} & \dots & x_{1r} \\ x_{21} & x_{22} & \dots & x_{2r} \\ \dots & \dots & \dots & \dots \\ x_{n1} & x_{n2} & \dots & x_{nr} \end{pmatrix} = \begin{pmatrix} b_{11} & b_{12} & \dots & b_{1r} \\ b_{21} & b_{22} & \dots & b_{2r} \\ \dots & \dots & \dots & \dots \\ b_{m1} & b_{m2} & \dots & b_{mr} \end{pmatrix}.$$

Rovnice (5.3.4) je ekvivalentní r soustavám lineárních rovnic

$$\mathbf{A}\mathbf{s}_k(\mathbf{X}) = \mathbf{s}_k(\mathbf{B}), \quad k = 1,2,...,r, \quad r \geqq 1,$$

kde

$$\mathbf{s}_k(\mathbf{X}) = (x_{1k}, x_{2k}, ..., x_{nk})^T, \quad \mathbf{s}_k(\mathbf{B}) = (b_{1k}, b_{2k}, ..., b_{mk})^T,$$

jsou k-té sloupce matic $\mathbf{X}$ a $\mathbf{B}$.

Do kategorie úloh reprezentovaných rovnicí (5.3.4) patří úloha určit inverzní matici $\mathbf{A}^{-1}$ k dané čtvercové regulární matici $\mathbf{A}$. Jde v tomto případě o řešení maticové rovnice

$$\mathbf{AX} = \mathbf{I}, \quad \mathbf{X} = \mathbf{A}^{-1}. \tag{5.3.5}$$

5.3.3 Úloha na vlastní čísla.

Je dána čtvercová matice $\mathbf{A}$ řádu n. Chceme stanovit takové číslo $\lambda = \lambda(\mathbf{A})$, pro které má (homogenní) rovnice

$$\mathbf{A}\mathbf{v} = \lambda\mathbf{v}, \quad \text{resp.} \quad (\mathbf{A}-\lambda\mathbf{I})\mathbf{v} = 0 \tag{5.3.6}$$

nenulové řešení $\mathbf{v}$ (nulové řešení nás nezajímá). Takovým číslům λ říkáme *vlastní čísla matice* $\mathbf{A}$ a odpovídajícím nenulovým řešením $\mathbf{v}$ *vlastní vektory matice* $\mathbf{A}$.

5.3.4 Zobecněná řešení soustav lineárních rovnic.

Počet lineárně nezávislých řádků matice $\mathbf{A}$ nazýváme *hodností* (řádkovou hodností) matice $\mathbf{A}$ a značíme $h(\mathbf{A})$. Symbolem $h(\mathbf{A}, \mathbf{b})$ budeme značit *hodnost matice rozšířené*, odpovídající soustavě $\mathbf{A}\mathbf{x} = \mathbf{b}$.

Z lineární algebry víme (viz např. [15], [9]), že soustava $\mathbf{A}\mathbf{x} = \mathbf{b}$ [$\mathbf{A}$ je typu (m, n)] má řešení, právě když $h(\mathbf{A}) = h(\mathbf{A}, \mathbf{b})$. Když $h(\mathbf{A}) = h(\mathbf{A}, \mathbf{b}) = n$, má tato soustava jediné řešení (pro čtvercovou matici řádu n viz odst.5.3.7) a když $h(\mathbf{A}) = h(\mathbf{A}, \mathbf{b}) < n$, má zmíněná soustava nekonečně mnoho řešení tvaru (tzv. *obecné řešení*) $\mathbf{x} = \mathbf{x}_0 + \sum_{i=1}^{s} \alpha_i \mathbf{u}_i$, kde $\mathbf{u}_i$ jsou lineárně nezávislé vektory, α_i jsou libovolná čísla a platí $\mathbf{A}\mathbf{x}_0 = \mathbf{b}$, $\mathbf{A}\mathbf{u}_i = \mathbf{0}$, $i = 1, 2, ..., s$, $s = n - h(\mathbf{A})$. Stručná zmínka o této situaci je v příkl. 6.3.3.

V praxi ovšem nejsou řídké případy soustav, ve kterých $m > n$, a je tedy obtížné zjistit, zda je soustava vůbec řešitelná. Je proto užitečné mít k dispozici nějaký numericky efektivní algoritmus výpočtu hodnosti matice a současně návod jak takové soustavy s obdélníkovými maticemi řešit, aniž bychom museli předem pracně zjišťovat, které rovnice jsou v soustavě "nadbytečné" nebo které si dokonce odporují.

Pro obecnou matici $\mathbf{A}$ a libovolný vektor $\mathbf{b}$ nás může zajímat takový vektor $\mathbf{x}_p$, pro který $\mathbf{A}\mathbf{x}_p - \mathbf{b}$ je v nějaké normě co nejmenší.

O zmíněných úlohách nalezne čtenář stručnou informaci v čl. 11.

6 Přímé metody soustavy lineárních rovnic

Chceme řešit soustavu $\mathbf{A}\mathbf{x} = \mathbf{b}$ s regulární (plnou) maticí soustavy řádu n. Podrobně je tato úloha formulována v odst. 5.3.1.

Metodu řešení soustavy $\mathbf{Ax} = \mathbf{b}$, která vede k přesnému řešení (neberemeli v úvahu vliv zaokrouhlovacích chyb) po konečném počtu kroků, nazýváme *přímou metodou*. Základním principem přímých metod je *eliminace* neznámých. Pro plné matice jsou přímé metody většinou nejefektivnější. pro velká n je však jejich použití limitováno kapacitou paměti počítače.

Iterační metody bývají efektivnější pro některé typy řídkých matic. Pro velké soustavy bývají iterační metody většinou nepostradatelné. Těmto metodám je věnován čl. 7.

Všechny typy eliminačních metod vycházejí z faktu, že soustavy

$$\mathbf{Ax} = \mathbf{b}, \qquad \mathbf{TAx} = \mathbf{Tb},$$

kde $\mathbf{T}$ je libovolná regulární matice řádu n, mají totéž řešení - říkáme, že jsou *ekvivalentní*.

6.1 Řešení trojúhelníkových soustav

6.1.1 Příklad.

Stanovme řešení soustavy $\mathbf{Ux} = \mathbf{y}$:

$$\begin{aligned} x_1 + 2x_2 + 3x_3 + 4x_4 &= 2, \\ 2x_2 + 6x_3 + 12x_4 &= 8, \\ 6x_3 + 24x_4 &= 18, \\ 24x_4 &= 24. \end{aligned}$$

Soustava má jediné řešení, nebot' $\det\mathbf{U} = 1\cdot 2\cdot 6\cdot 24 \neq 0$ (součin diagonálních prvků).

Z poslední rovnice vypočteme $x_4 = 1$; tuto hodnotu dosadíme do předposlední rovnice a vypočteme

$$x_3 = \frac{1}{6}(18 - 24\cdot 1) = -1.$$

Dále pak

$$\begin{aligned} x_2 &= \frac{1}{2}(8 - 6x_3 - 12x_4) = 1, \\ x_1 &= (2 - 2x_2 - 3x_3 - 4x_4) = -1. \end{aligned}$$

Postup, který jsme si na tomto příkladu ukázali, je v té či oné modifikaci součástí většiny algoritmů eliminačních metod. Říkáme mu *zpětná substituce.*

6.1.2 Obecný případ.

Mějme soustavu lineárních rovnic.

$$\begin{aligned} u_{11}x_1 + u_{12}x_2 + ... + u_{1n}x_n &= y_1, \\ u_{22}x_2 + ... + u_{2n}x_n &= y_2, \\ &\dots\dots\dots\dots\dots, \\ u_{nn}x_n &= y_n, \end{aligned} \tag{6.1.1}$$

maticově zapsanou ve tvaru

$$\mathbf{U}\mathbf{x} = \mathbf{y}. \tag{6.1.2}$$

V matici $\mathbf{U} = (u_{ij})$ jsou prvky pod hlavní diagonálou nulové ($i > j$). Takovým maticím říkáme (*horní*) *trojúhelníkové*. Předpokládáme, že $u_{ii} \neq 0$, $i = 1, 2, ..., n$, a vypočítáváme neznámé v pořadí $x_n, x_{n-1}, ..., x_2, x_1$ zpětnou substitucí ze vztahů

$$\begin{aligned} x_n &= \frac{y_n}{u_{nn}}, \\ x_{n-1} &= \frac{1}{u_{n-1,n-1}}(y_{n-1} - u_{n-1,n}x_n), \\ &\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots, \\ x_1 &= \frac{1}{u_{11}}(y_1 - u_{12}x_2 - u_{13}x_3 - ... - u_{1n}x_n). \end{aligned}$$

6.1.3 Algoritmus zpětné substituce.

Neznámou x_i soustavy (6.1.1) počítáme z formulace

$$x_i = \frac{1}{u_{ii}} \left(y_i - \sum_{k=i+1}^{n} u_{ik}x_k \right).$$

Zapíšeme tedy algoritmus ve tvaru:

$$\begin{aligned} &\text{Vstup :} \quad n, \mathbf{y} = (y_1, y_2, ..., y_n)^T, \quad \mathbf{U} = (u_{ij}). \\ &\qquad \text{Pro} \quad i = n, n-1, ..., 2, 1 : \\ &\qquad\quad x_i = \frac{1}{u_{ii}} \left(y_i - \sum_{j=i+1}^{n} u_{ij}x_j \right). \\ &\text{Výstup :} \quad \mathbf{x} = (x_1, x_2, ..., x_n)^T. \end{aligned} \tag{6.1.3}$$

(Klademe $\sum_{j=k}^{n} c_j = 0$, když $k > n$.) K výpočtu neznámé x_j potřebujeme jedno dělení, $n-j$ násobení a $n-j$ sčítání. Tedy celkem n^2 aritmetických operací [$n + \sum_{j=1}^{n}(n-j) = \frac{n(n+1)}{2}$ násobení a dělení[10]), $\sum_{j=1}^{n}(n-j) = \frac{(n-1)n}{2}$ sčítání]. Při realizaci algoritmu na počítači vzniknou potíže, pokud čísla u_{jj} budou malá. Potom matice $\mathbf{U}$ je skoro singulární $\det\mathbf{U} \approx 0$. O takových soustavách se zmíníme v č. 10.

6.1.4 Příklad.

Chceme stanovit řešení soustavy $\mathbf{Ly} = \mathbf{b}$, kde $\mathbf{L} = (l_{ij})$ je *dolní trojúhelníková matice*, tj. taková, v níž $l_{ij} = 0$, když $i < j$:

$$\begin{aligned} l_{11}y_1 &= b_1, \\ l_{21}y_1 + l_{22}y_2 &= b_2, \\ &\dots\dots\dots\dots\dots\dots\dots\dots, \\ l_{n1}y_1 + l_{n2}y_2 + ... + l_{nn}y_n &= b_n. \end{aligned}$$

Zde použijeme algoritmu přímé substituce, tj. vypočítáváme neznámé v pořadí $y_1, y_2, ..., y_n$.

$$\begin{aligned} &\text{Vstup}: \quad n, \mathbf{b} = (b_1, b_2, ..., b_n)^T, \quad \mathbf{L} = (l_{ij}). \\ &\qquad \text{Pro} \quad i = 1, 2, ..., n: \\ &\qquad\qquad y_i = \frac{1}{l_{ii}}\left(b_i - \sum_{k=1}^{i-1} l_{ik}y_k\right). \\ &\text{Výstup}: \quad \mathbf{y} = (y_1, y_2, ..., y_n)^T. \end{aligned} \tag{6.1.4}$$

6.2 Gaussova eliminační metoda - GEM.

Je to jedna z nejstarších numerických metod. Pozornost věnovaná této metodě především v létech 1955-1965 se zaměřila na dva aspekty eliminační metody: výběr hlavních prvků (pivotů) a problematiku efektů způsobených zaokrouhlováním.

[10]) Zde i v dalším užíváme vzorců

$$\sum_{k=1}^{m} k = \frac{m(m+1)}{2}; \quad \sum_{k=1}^{m} k^2 = \frac{1}{6}m(m+1)(2m+1).$$

Předpokládáme, že čtenář se již seznámil s Gaussovou eliminační metodou při studiu lineární algebry (např. v publikacích [9], [15]). Na jednoduchém příkladě si připomeneme algoritmus GEM už v té podobě, ve které bude popsán i pro soustavu n lineárních rovnic.

6.2.1 Příklad.

Chceme řešit soustavu lineárních rovnic

$$\begin{pmatrix} 2 & -1 & 3 & -1 \\ 1 & -1 & 4 & -2 \\ 3 & 2 & 1 & 4 \\ 4 & -3 & 3 & -3 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} = \begin{pmatrix} 7 \\ 5 \\ 31 \\ -5 \end{pmatrix}.$$

Budeme provádět tzv. řádkové úpravy rozšířené matice soustavy tak, abychom dostali pod hlavní diagonálu matice soustavy nulové prvky.

$$\begin{matrix} \\ -\frac{1}{2} \\ -\frac{3}{2} \\ -2 \end{matrix} \left(\begin{array}{cccc|c} 2 & -1 & 3 & -1 & 7 \\ 1 & -1 & 4 & -2 & 5 \\ 3 & 2 & 1 & 4 & 31 \\ 4 & -3 & 3 & -3 & -5 \end{array}\right) \overset{\text{1. fáze}}{\longrightarrow} \begin{matrix} \\ 7 \\ \\ -2 \end{matrix} \left(\begin{array}{cccc|c} 2 & -1 & 3 & -1 & 7 \\ 0 & -\frac{1}{2} & \frac{5}{2} & -\frac{3}{2} & \frac{3}{2} \\ 0 & \frac{7}{2} & -\frac{7}{2} & \frac{11}{2} & \frac{41}{2} \\ 0 & -1 & -3 & -1 & -19 \end{array}\right) \overset{\text{2. fáze}}{\longrightarrow}$$

$$\overset{\text{2. fáze}}{\longrightarrow} \begin{matrix} \\ \\ \\ -\frac{4}{7} \end{matrix} \left(\begin{array}{cccc|c} 2 & -1 & 3 & -1 & 7 \\ 0 & -\frac{1}{2} & \frac{5}{2} & -\frac{3}{2} & \frac{3}{2} \\ 0 & 0 & 14 & -5 & 31 \\ 0 & 0 & -8 & 2 & -22 \end{array}\right) \overset{\text{3. fáze}}{\longrightarrow} \left(\begin{array}{cccc|c} 2 & -1 & 3 & -1 & 7 \\ 0 & -\frac{1}{2} & \frac{5}{2} & -\frac{3}{2} & \frac{3}{2} \\ 0 & 0 & 14 & -5 & 31 \\ 0 & 0 & 0 & -\frac{6}{7} & -\frac{30}{7} \end{array}\right).$$

K realizaci 1. fáze eliminace nejdříve stanovíme multiplikátory 1. fáze. Jsou to záporně vzaté podíly čísel v pozicích $(2,1)$, $(3,1)$, $(4,1)$ a čísla v pozici $(1,1)$ (tzv. hlavní prvek 1. fáze eliminace). Tyto multiplikátory jsou zapsány vlevo od rozšířené matice soustavy. Multiplikátorem 2. řádku (číslo $-\frac{1}{2}$) vynásobíme 1. řádek a tento součin přičteme k 2. řádku. Dostaneme tak 2. řádek redukované matice po 1. fáze. Analogicky postupujeme u zbývajících řádků. Řádek, jehož násobky přičítáme k ostatním řádkům (neupravoval se), se nazývá hlavním řádkem 1. fáze. Zde to byl 1. řádek.

Úpravy 2. fáze jsou zcela analogické; bereme v úvahu pouze 2., 3. a 4. řádek - k 3. a 4. řádku přičítáme násobky 2. řádku (hlavní řádek 2. fáze). Úpravy v 3. fázi eliminace se týkají pouze 3. a 4. řádku.

Soustavu s redukovanou (trojúhelníkovou) maticí soustavy řešíme zpětnou substitucí (odst. 6.1).

Dostaneme

$$x_4 = 5, \quad x_3 = 4, \quad x_2 = 2, \quad x_1 = 1,$$

tj. řešením (přesným) dané soustavy je vektor

$$\mathbf{x}_t = (1, 2, 4, 5)^T.$$

Označíme

$$\mathbf{L} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ \frac{1}{2} & 1 & 0 & 0 \\ \frac{3}{2} & -7 & 1 & 0 \\ 2 & 2 & -\frac{4}{7} & 1 \end{pmatrix}, \quad \mathbf{U} = \begin{pmatrix} 2 & -1 & 3 & -1 \\ 0 & -\frac{1}{2} & \frac{5}{2} & -\frac{3}{2} \\ 0 & 0 & 14 & -5 \\ 0 & 0 & 0 & -\frac{6}{7} \end{pmatrix}$$

kde matice $\mathbf{L}$ je sestavena pomocí multiplikátorů jednotlivých fází eliminace a $\mathbf{U}$ je výsledná (horní trojúhelníková) matice získaná příslušnými řádkovými úpravami.

Čtenář necht' se přesvědčí o tom, že platí $\mathbf{LU} = \mathbf{A}$.

6.2.2 Výklad Gaussovy eliminace.

Uvažujme soustavu n rovnic pro n neznámých ve tvaru

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + ... + a_{1n}x_n &= a_{1,n+1}, \\ a_{21}x_1 + a_{22}x_2 + ... + a_{2n}x_n &= a_{2,n+1}, \\ &\cdots\cdots\cdots\cdots\cdots\cdots\cdots \\ a_{n1}x_1 + a_{n2}x_2 + ... + a_{nn}x_n &= a_{n,n+1}. \end{aligned} \tag{6.2.1}$$

Předpokládáme, že matice $\mathbf{A} = (a_{ij})$ je regulární. Necht' $a_{11} \neq 0$. Přičtením vhodných násobků 1. rovnice ke zbývajícím $n-1$ rovnicím vyloučíme neznámou x_1 z těchto rovnic. Těmito vhodnými násobky jsou čísla

$$m_{i1} = \frac{a_{i1}}{a_{11}}, \quad i = 2, 3, ..., n,$$

a říkáme jim *multiplikátory* 1. fáze eliminace. Dostaneme první *redukovanou soustavu* (1. rovnice se nemění)

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + ... + a_{1n}x_n &= a_{1,n+1}, \\ \left.\begin{aligned} a^{(1)}_{22}x_2 + ... + a^{(1)}_{2n}x_n &= a^{(1)}_{2,n+1}, \\ &\cdots\cdots\cdots\cdots\cdots \\ a^{(1)}_{n2}x_2 + ... + a^{(1)}_{nn}x_n &= a^{(1)}_{n,n+1}, \end{aligned}\right\} & \text{ soustava } n-1 \text{ rovnic} \\ & \text{pro } n-1 \text{ neznámých} \end{aligned}$$

kde

$$a^{(1)}_{ij} = a_{ij} + m_{i1}a_{1j}; \quad i = 2, 3, ..., n; \; j = 2, 3, ..., n, n+1.$$

V druhé fázi eliminace předpokládáme, že $a_{22}^{(1)} \neq 0$ a vyloučíme neznámou x_2 ze zbývajících $n-2$ rovnic redukované soustavy tím, že 2. rovnici redukované soustavy vynásobíme multiplikátory

$$m_{i2} = -\frac{a_{i2}^{(1)}}{a_{22}^{(1)}}, \quad i = 3, 4, ..., n,$$

a postupně přičteme tento násobek 2. rovnice ke zbývajícím $n-3$ rovnicím. Dostaneme tak druhou redukovanou soustavu

$$\left.\begin{aligned} a_{11}x_1 + a_{12}x_2 + a_{13}x_3 + ... + a_{1n}x_n &= a_{1,n+1}, \\ a_{22}^{(1)}x_2 + a_{23}^{(1)}x_3 + ... + a_{2n}^{(1)}x_n &= a_{2,n+1}^{(1)}, \\ a_{33}^{(2)}x_3 + ... + a_{3n}^{(2)}x_n &= a_{3,n+1}^{(2)}, \\ ..... & \\ a_{n3}^{(2)}x_2 + ... + a_{nn}^{(2)}x_n &= a_{n,n+1}^{(2)}, \end{aligned}\right\} \begin{array}{l} \text{soustava } n-2 \text{ rovnic} \\ \text{pro } n-2 \text{ neznámých} \end{array}$$

kde

$$a_{ij}^{(2)} = a_{ij}^{(1)} + m_{i2}a_{2j}^{(1)}; \quad i = 3, 4, ..., n; \; j = 3, 4, ..., n, n+1.$$

Dále pokračujeme analogickým způsobem, až po $n-1$ krocích dostaneme konečnou redukovanou soustavu s trojúhelníkovou maticí ve tvaru

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + a_{13}x_3 + ... + a_{1n}x_n &= a_{1,n+1}, \\ a_{22}^{(1)}x_2 + a_{23}^{(1)}x_3 + ... + a_{2n}^{(1)}x_n &= a_{2,n+1}^{(1)}, \\ a_{33}^{(2)}x_3 + ... + a_{3n}^{(2)}x_n &= a_{3,n+1}^{(2)}, \\ ..... & \\ a_{nn}^{(n-1)}x_n &= a_{n,n+1}^{(n-1)}, \end{aligned}$$

Tím je původní soustava převedena na trojúhelníkový tvar a k výpočtu řešení můžeme užit algoritmu zpětné substituce (6.1.3).

6.2.3 Algoritmus redukce matice A na trojúhelníkový tvar.

Označíme $\mathbf{A} = (a_{ij}) = \mathbf{A}^{(0)} = (a_{ij}^{(0)})$. Řádkové úpravy z předchozího odstavce zapíšeme ve tvaru

$$\left.\begin{aligned} &\mathbf{r}_i(\mathbf{A}^{(k)}) = \mathbf{r}_i(\mathbf{A}^{(k-1)}), \qquad \mathbf{r}_i(\mathbf{b}^{(k)}) = \mathbf{r}_i(\mathbf{b}^{(k-1)}), \\ &\mathbf{r}_i(\mathbf{A}^{(k)}) = \mathbf{r}_i(\mathbf{A}^{(k-1)}) + m_{ik}\mathbf{r}_k(\mathbf{A}^{(k-1)}) \\ &\mathbf{r}_i(\mathbf{b}^{(k)}) = \mathbf{r}_i(\mathbf{b}^{(k-1)}) + m_{ik}\mathbf{r}_k(\mathbf{b}^{(k-1)}) \end{aligned}\right\} \begin{array}{ll} \text{pro} & i \leqq k, \\ & \\ \text{pro} & i > k, \end{array}$$

kde

$$m_{ik} = -\frac{a_{ik}^{(k-1)}}{a_{kk}^{(k-1)}} \quad \text{(za předpokladu} \quad a_{kk}^{(k-1)} \neq 0).$$

Pak již můžeme psát v rozepsané podobě (stačí zapisovat pouze případ $i > k$):

$$\begin{aligned}
&\text{Vstup}: \quad n,\ \mathbf{A} = (a_{ij}^{(0)}), \quad \mathbf{b} = (a_{1,n+1}^{(0)}, a_{2,n+1}^{(0)}, ..., a_{n,n+1}^{(0)})^T.\\
&\qquad \text{Pro} \quad k = 1, 2, ..., n-1:\\
&\qquad\qquad \text{Pro} \quad i = k+1, k+2, ..., n;\\
&\qquad\qquad\qquad m_{ik} = -\frac{a_{ik}^{(k-1)}}{a_{kk}^{(k-1)}}.\\
&\qquad\qquad\qquad \text{Pro} \quad j = k+1, k+2, ..., n, n+1:\\
&\qquad\qquad\qquad\qquad a_{ij}^{(k)} = a_{ij}^{(k-1)} + m_{ik} a_{kj}^{(k-1)}.\\
&\text{Výstup}: \quad \mathbf{U} = (a_{ij}^{(i-1)}), \quad \mathbf{y} = (a_{1,n+1}^{(0)}, a_{2,n+1}^{(1)}, ..., a_{n,n+1}^{(n-1)})^T.
\end{aligned} \tag{6.2.2}$$

K výpočtu $n-k$ multiplikátorů k-té fáze eliminace potřebujeme $n-k$ dělení; celkem pro $n-1$ fází

$$\sum_{k=1}^{n-1}(n-k) = \frac{n(n-1)}{1}$$

dělení. Každým multiplikátorem vynásobíme $n-k+1$ koeficientů rozšířené matice, tj. $(n-k)(n-k+1)$ násobení v k-té fázi, což dává celkem

$$\sum_{k=1}^{n-1}(n-k)(n-k+1) = \sum_{k=1}^{n-1}[(n^2+n) - k(2n+1) + k^2] = \frac{n^3-n}{3}$$

(viz str. **??**, p. čarou [10])) násobení; stejný bude počet sčítání.

K realizaci algoritmu (6.2.2) tedy potřebujeme

$$\begin{aligned}
&\frac{n^3-n}{3} + \frac{n(n-1)}{2} = \frac{1}{3}n^3 + \frac{n^2}{2} - \frac{5}{6}n && \text{násobení a dělení,}\\
&\frac{n^3-n}{3} && \text{sčítání;}
\end{aligned}$$

pro algoritmus Gaussovy eliminace [algoritmy (6.2.3) a (6.1.3) potřebujeme:

$$\begin{aligned}
&\frac{1}{3}n^3 + n^2 - \frac{1}{3}n = \frac{1}{3}n^3 + O(n^2) && \text{násobení a dělení,}\\
&\frac{1}{3}n^3 + \frac{n^2}{2} - \frac{5}{6}n = \frac{1}{3}n^3 + O(n^2) && \text{sčítání.}
\end{aligned}$$

.

6.2.4 Analýza eliminační metody

Z úvah odst. 6.2.2 vyplývá, že řádkové úpravy prováděné při Gaussově eliminaci lze interpretovat jako postupné násobení matice $\mathbf{A}$ *transformačními maticemi* $\mathbf{T}_1, \mathbf{T}_2, \mathbf{T}_3, ..., \mathbf{T}_{n-1}$, kde

$$\mathbf{T}_k = \begin{pmatrix} 1 & 0 & \cdots & 0 & \cdots & \cdots & \cdots & 0 \\ 0 & 1 & \ddots & \vdots & & & & \vdots \\ \vdots & \ddots & \ddots & 0 & & & & \vdots \\ 0 & \cdots & 0 & 1 & 0 & 0 & \cdots & 0 \\ 0 & \cdots & 0 & m_{k+1,k} & 1 & \ddots & & \vdots \\ \vdots & & \vdots & m_{k+2,k} & 0 & \ddots & \ddots & \vdots \\ \vdots & & \vdots & \vdots & \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & m_{n,k} & 0 & \cdots & 0 & 1 \end{pmatrix};$$

Označujeme

$$\mathbf{A} = \mathbf{A}^{(0)} = (a_{ii}^{(0)}),\ \mathbf{T}_1\mathbf{A}^{(0)} = \mathbf{A}^{(1)} = (a_{ij}^{(1)}), ...,$$
$$\mathbf{T}_k\mathbf{A}^{(k-1)} = \mathbf{A}^{(k)} = (a_{ij}^{(k)}), ...,\ \mathbf{T}_{n-1}\mathbf{A}^{(n-2)} = \mathbf{A}^{(n-1)} = \mathbf{U};$$
$$\mathbf{b} = \mathbf{b}^{(0)},\ \mathbf{b}^{(1)} = \mathbf{T}_1\mathbf{b}^{(0)}, ...,\ \mathbf{b}^{(k)} = \mathbf{T}_k\mathbf{b}^{(k-1)}, ...,\ \mathbf{b}^{(n-1)} = \mathbf{T}_{n-1}\mathbf{b}^{(n-2)}.$$

Po realitaci k-tého kroku eliminace dostaneme

$$\mathbf{A}^{(k)} = \mathbf{T}_k\mathbf{A}^{(k-1)} = \begin{pmatrix} a_{11}^{(0)} & a_{12}^{(0)} & \cdots & a_{1k}^{(0)} & \cdots & \cdots & a_{1n}^{(0)} \\ 0 & a_{22}^{(1)} & \cdots & a_{2k}^{(1)} & \cdots & \cdots & a_{2n}^{(1)} \\ \vdots & \ddots & \ddots & \vdots & & & \vdots \\ \vdots & & 0 & a_{k+1,k+1}^{(k)} & a_{k+1,k+2}^{(k)} & \cdots & a_{k+1,n}^{(k)} \\ \vdots & & 0 & a_{k+2,k+1}^{(k)} & a_{k+2,k+2}^{(k)} & \cdots & a_{k+2,n}^{(k)} \\ \vdots & & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & a_{n,k+1}^{(n)} & a_{n,k+2}^{(k)} & \cdots & a_{n,n}^{(k)} \end{pmatrix}$$

$$\mathbf{b}^{(k)} = \mathbf{T}_k\mathbf{b}^{(k-1)} = (a_{1,n+1}^{(0)}, a_{2,n+1}^{(1)}, a_{3,n+1}^{(2)}, ..., a_{k+1,n+1}^{(k)}, ..., a_{n,n+1}^{(k)})^T.$$

Po ukončení všech $n-1$ fází eliminace tedy dostáváme redukovanou matici soustavy a sloupec pravých stran:

$$\mathbf{U} = \mathbf{A}^{(n-1)} = \mathbf{T}\mathbf{A}^{(0)}, \qquad \mathbf{y} = \mathbf{b}^{(n-1)} = \mathbf{T}\mathbf{b}^{(0)},$$

kde

$$\mathbf{T} = \mathbf{T}_{n-1}\mathbf{T}_{n-2}...\mathbf{T}_2\mathbf{T}_1.$$

Matice $\mathbf{T}$ je regulární, a proto existuje inverzní matice $\mathbf{T}^{-1}$, kterou označíme $\mathbf{L}$. Čtenář se může přesvědčit, že

$$\mathbf{L} = \mathbf{T}^{-1} = \begin{pmatrix} 1 & 0 & \cdots & \cdots & 0 \\ -m_{21} & 1 & \ddots & & \vdots \\ -m_{31} & -m_{32} & \ddots & \ddots & \vdots \\ \vdots & \vdots & \ddots & \ddots & 0 \\ -m_{n1} & -m_{n2} & \cdots & -m_{n-1,n} & 1 \end{pmatrix}$$

Zjistili jsme, že matici $\mathbf{A}$ lze rozložit na součin dolní trojúhelníkové matice $\mathbf{L}$ a horní trojúhelníkové matice $\mathbf{U}$, tj. že platí

$$\mathbf{A} = \mathbf{LU}. \tag{6.2.3}$$

Stanovení matic $\mathbf{L}$ a $\mathbf{U}$ nazýváme *trojúhelníkovým rozkladem* (*LU-rozkládem*) matice $\mathbf{A}$. K dané matici $\mathbf{A}$ lze určit více trojúhelníkových rozkladů podle toho, jak volíme diagonální prvky matice $\mathbf{L}$. Je to patrné z následujícího příkladu:

$$\begin{pmatrix} 2 & 10 \\ 7 & 44 \end{pmatrix} = \begin{cases} \begin{pmatrix} 1 & 0 \\ \frac{7}{2} & 1 \end{pmatrix} \begin{pmatrix} 2 & 10 \\ 0 & 9 \end{pmatrix}, \\ \begin{pmatrix} 2 & 0 \\ 7 & 3 \end{pmatrix} \begin{pmatrix} 1 & 5 \\ 0 & 3 \end{pmatrix}. \end{cases}$$

Platí *věta o jednoznačnosti rozkladu: Trojúhelníkový rozklad regulární matice* $\mathbf{A}$ *je určen jednoznačně diagonálními prvky matice* $\mathbf{L}$ *a může být stanoven Gaussovou eliminací.* Důkaz najde čtenář např. v [4].

Otázka zda pro každou regulární matici $\mathbf{A}$ je realizovatelná Gaussova eliminace (popsaná v odst. 6.2.2), je ekvivalentní otázce, zda existuje trojúhelníkový rozklad čtvercové regulární matice $\mathbf{A}$.

Pro některé matice však není realizovatelná Gaussova eliminace, a proto neexistuje ani jejich trojúhelníkový rozklad. Například u matice $\begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix}$ není realizovatelný už první krok Gaussovy eliminace, a tedy ji nelze rozložit na součin dvou trojúhelníkových matic.

To je také jeden z důvodů, proč si v dalších odtsavcích uvedeme některé modifikace Gaussovy eliminační metody (tzv. metody s výběrem hlavního prvku).

Existují dvě třídy matic, pro které je algoritmus Gaussovy eliminace vždy proveditelný, a to matice symetrické pozitivně definitní nebo regulární diagonálně dominantní (viz odst. 6.6.1).

Uved'me si ještě *obecnou větu o existenci a jednoznačnosti rozkladu: Necht'* $\mathbf{A}$ *je čtvercová matice řádu* n *a necht'* $\mathbf{A}_k$ *je matice typu* (k,k) *sestavená ze společných prvků prvních* k *řádků a sloupců matice* $\mathbf{A}$ *a necht'* $\det \mathbf{A}_k \neq 0$, $k = 1, 2, ..., n-1$. *Potom existuje jediná dolní trojúhelníková matice* $\mathbf{L}$ *s jednotkami na diagonále a jediná horní trojúhelníková matice* $\mathbf{U}$ *takové, že* $\mathbf{LU} = \mathbf{A}$.

6.2.5 Numerické aspekty GEM.

Protože vždy počítáme v nějaké množině $M(q,t)$, nebudou jak multiplikátory, tak prvky redukované soustavy vypočítány přesně. Proto místo $\mathbf{L}$, $\mathbf{U}$ vlastně výpočteme $\tilde{\mathbf{L}}$, $\tilde{\mathbf{U}}$, přičemž $\tilde{\mathbf{L}}\tilde{\mathbf{U}} \neq A$. Označíme-li

$$\tilde{\mathbf{A}} = \tilde{\mathbf{L}}\tilde{\mathbf{U}},$$

bude nás zajímat rozdíl $\mathbf{A} - \tilde{\mathbf{A}}$. Jistě existují matice chyb $\mathbf{E}$, $\mathbf{F}$ takové, že

$$\tilde{\mathbf{L}} = \mathbf{L} + \mathbf{E}, \qquad \tilde{\mathbf{U}} = \mathbf{U} + \mathbf{F}.$$

Potom

$$\mathbf{A} - \tilde{\mathbf{A}} = \mathbf{LU} - \tilde{\mathbf{L}}\tilde{\mathbf{U}} = \mathbf{LU} - (\mathbf{L} + \mathbf{E})(\mathbf{U} + \mathbf{F}) = \mathbf{EU} + \mathbf{LF} + \mathbf{EF}.$$

Odtud plyne důležitý závěr: Pokud při realizaci Gaussovy eliminace vycházejí pro multiplikátory nebo prvky matice redukované soustavy velká čísla, jsou též prvky matice $\mathbf{L}$ velká čísla, a tedy rozdíl $\mathbf{A} - \tilde{\mathbf{A}}$ (zvláště pro velká n) bude řádově mnohonásobně větší než $\mathbf{E}$, resp. $\mathbf{F}$. V důsledku této skutečnosti může být rozdíl přesného a vypočteného řešení nepřijatelně velký.

Říkáme v takovém případě, že algoritmus Gaussovy eliminace je numerický nestabilní.

6.2.6 GEM pomocí kalkulátoru.

Často potřebujeme řešit lineární soustavy s poměrně malým $n\,(\leqq 10)$. Nebývá vždy optimální použít počítače. Vyjdeme z postupu evedeného v příkl. 6.2.1 a provedeme pouze jistou organizaci výpočtu. Poněkud jiný postup je popsán v odst. 6.5.4. Abychom vyloučili chyby lidského činitele, budeme v každé fázi výpočtu provádět kontrolu pomocí tzv. *sloupce kontrolních součtů* $\sum$.

Danou soustavu a všechny potřebné mezivýsledky budeme zapisovat do tabulky (v tab. 2 je $n = 4$):
Postup zaplňování tab. 2 (algoritmus):

1. Zapíšeme prvky matice $\mathbf{A}$ a složky vektoru $\mathbf{b}$ (první čtyři řídky tabulky).

2. Sečteme všechna čísla v jednotlivých řádcích a výsledky zapíšeme do sloupce $\sum$.

		A				b	$\sum$
1		$a_{11}^{(0)}$	$a_{12}^{(0)}$	$a_{13}^{(0)}$	$a_{14}^{(0)}$	$a_{15}^{(0)}$	$\sum_{j=1}^{5} a_{1j}^{(0)}$
2	m_{21}	$a_{21}^{(0)}$	$a_{22}^{(0)}$	$a_{23}^{(0)}$	$a_{24}^{(0)}$	$a_{25}^{(0)}$	$\sum_{j=1}^{5} a_{2j}^{(0)}$
3	m_{31}	$a_{31}^{(0)}$	$a_{32}^{(0)}$	$a_{33}^{(0)}$	$a_{34}^{(0)}$	$a_{35}^{(0)}$	$\sum_{j=1}^{5} a_{3j}^{(0)}$
4	m_{41}	$a_{41}^{(0)}$	$a_{42}^{(0)}$	$a_{43}^{(0)}$	$a_{44}^{(0)}$	$a_{45}^{(0)}$	$\sum_{j=1}^{5} a_{4j}^{(0)}$
5			$a_{22}^{(1)}$	$a_{23}^{(1)}$	$a_{24}^{(1)}$	$a_{25}^{(1)}$	$\sum_{j=2}^{5} a_{2j}^{(1)}$
6	m_{32}		$a_{32}^{(1)}$	$a_{33}^{(1)}$	$a_{34}^{(1)}$	$a_{35}^{(1)}$	$\sum_{j=2}^{5} a_{3j}^{(1)}$
7	m_{42}		$a_{42}^{(1)}$	$a_{43}^{(1)}$	$a_{44}^{(1)}$	$a_{45}^{(1)}$	$\sum_{j=2}^{5} a_{4j}^{(1)}$
8				$a_{33}^{(2)}$	$a_{34}^{(2)}$	$a_{35}^{(2)}$	$\sum_{j=3}^{5} a_{3j}^{(2)}$
9	m_{43}			$a_{43}^{(2)}$	$a_{44}^{(2)}$	$a_{45}^{(2)}$	$\sum_{j=3}^{5} a_{4j}^{(2)}$
10					$a_{44}^{(3)}$	$a_{45}^{(3)}$	$\sum_{j=4}^{5} a_{4j}^{(3)}$

3. Vypočteme multiplikátory $m_{i1} = -a_{i1}^{(0)}/a_{11}^{(0)}$, $i = 2, 3, 4$, a zapíšeme je do sloupce m.

4. i-tým multiplikátorem ($i = 2, 3, 4$) vynásobíme každé číslo 1. rádku, mezivýsledek přičteme k číslu i-tého řádku, včetně čísel sloupce $\sum$, a jednotlivé výsledky napíšeme postupně do příslušných 5., 6., 7. řádku. Tedy

$$\begin{aligned} m_{i1} a_{1j}^{(0)} + a_{ij}^{(0)} &\to a_{ij}^{(1)} \\ m_{i1} \sum_{j=1}^{5} a_{1j}^{(0)} + \sum_{j=1}^{5} a_{ij}^{(0)} &\to \sum_{j=1}^{5} a_{ij}^{(1)}, \qquad i = 2, 3, 4. \end{aligned}$$

5. Provedeme kontrolu: Součet čísel v 5., 6., 7. řádku musí být roven odpovídajícím číslům sloupce $\sum$. Pokud odchylky nelze vysvětlit zaokrouhlovacími chybami dopustili jsme se ve výpočtu chyby.

6. Vypočteme multiplikátory $m_{i2} = -a_{i2}^{(1)}/a_{22}^{(1)}$, $i = 3, 4$.

7. Při zaplňování 8., 9. řádku postupujeme analogicky jako v bodu 4.

8. Provedeme opět kontrolu ve smyslu bodu 5.

9. Stanovíme $m_{43} = -a_{43}^{(2)}/a_{33}^{(2)}$, vynásobíme jím čísla 8. řádku a výsledky napíšeme do 10. řádku.

10. Provedeme kontrolu.

Redukovaná matice je "uložena" v 1., 5., 8. a 10. řádku. Vypočteme x_4 z 10. řádku (nebot' $a_{44}^{(3)} x_4 = a_{45}^{(3)}$) a dosadíme do 8. řádku, odkud vypočteme x_3 , nebot' $a_{33}^{(2)} x_3 + a_{34}^{(2)} x_4 = a_{35}^{(2)}$. Z 5. řádku vypočteme analogicky x_2 a z 1. řádku nakonec x_1 .

Poznamenejme, že užitečnou informaci o přesnosti získaného řešení nám může (ale nemusí!) poskytnout reziduum $\mathbf{r} = \mathbf{b} - \mathbf{A}\mathbf{x_c}$, nebot' teoreticky má být $\mathbf{b} - \mathbf{A}\mathbf{x}_t = \mathbf{0}$. To znamená, že když se $\mathbf{r}$ dosti liší od "počítačové nuly" liší jen málo, není tím ještě zaručena přesnost výsledku, jak uvidíme později.

6.2.7 Příklad.

Na kalkulátoru, který pracuje s čísly z $M(10,5)$ (řezání), řešme Gaussovou eliminační metodou soustavu $\mathbf{A}\mathbf{x} = \mathbf{b}$. Rozšířená matice soustavy je "uložena" v silně orámované části tab. 3.

Tab. 3

	m	**A**			**b**	$\sum$
1		2,475 9	1,632 5	4,623 1	0,064 700	8,787 2
2	-0,594 73	1,472 5	0,959 80	- 1,325 3	1,047 5	2,154 5
3	-1,088 5	2,695 1	2,896 5	- 1,479 4	-0,678 90	3,433 3
4			-0,005 744 1	- 4,074 7	1,009 0	- 3,071 5
5	196,60		1,129 3	- 6,511 6	-0,749 32	- 6,131 5
6				-807,59	197,62	- 609,98

Zpětnou substitucí (z 6., 4., 1. řádku) dostáváme

$$x_3 = -\frac{197,62}{807,59} \approx -0,244\,70,$$
$$x_2 \approx -2,075\,3,$$
$$x_1 \approx \quad 1,817\,7.$$

Zde se projevil růst multiplikátorů a prvků matice redukované soustavy (poslední řádek), což signalizuje numerickou nestabilitu algoritmu. Nesouhlas je v sloupci $\sum$ označen zatržením.

Odchylky od výsledků téže soustavy řešené týmž algoritmem k knize [12], kde je $\mathbf{x_c} = (1,828\,6; -2,053\,2; -0,244\,3)^T$, jsou způsobeny tím, že zde pracujeme stále s čísly z $M(10,5)$ a nikoliv na pevný počet desetinných míst.

Ve [12] jsou uvedeny výsledky lepší, získané algoritmem s výběrem hlavního prvku (viz čl. 6.4) $\mathbf{x_c} = (1,840\,5; -2,071\,6; -0,244\,70)^T$. Doporučujeme čtenáři, aby si na tomto příkladě vyzkoušel všechny modifikace eliminační metody uvedené v dalších odstavcích včetně volby $M(10,t)$. Výpočty v $M(10,5)$ může do jisté míry modelovat na kapesním kalkulátoru tím, že všechny mezivýsledky bude zapisovat tak, aby odpovídaly pětimístné mantise.

6.3 Řešení maticové rovnice.

Budeme se poněkud podrobněji zabývat úlohou, která byla formulována v odst. 5.3.2, kde jsme si uvedli, že maticová tovnice $\mathbf{AX} = \mathbf{B}$ [$\mathbf{A}$ je typu (m,n), $\mathbf{B}$ typu (m,r), $\mathbf{X}$ typu (n,r)] je ekvivalentní r soustavám tvaru $\mathbf{Ax} = \mathbf{b}$, kde $\mathbf{x}$ je typu $(n,1)$ a $\mathbf{b}$ je typu $(m,1)$.

V dalším budeme předpokládat, že $m = n$ a že $\mathbf{A}$ je regulární (čtvercová) matice.

Každé metody řešení úlohy z odst. 5.3.1 lze tedy užít k řešení maticové rovnice. Pouze algoritmus musíme upravit tak, aby umožňoval pracovat s více pravými stranami.

Na příkladu maticové rovnice si však vyložíme jednoduchou modifikaci eliminační metody, které se říká *Gaussova-Jordanova metoda* (GJEM) a které lze užít všude tam, kde lze užít i GEM.

Základní myšlenkou GJEM jsou takové řádkové úpravy matice soustavy, při nichž redukovaná matice po všech $n-1$ fázích eliminace je diagonální nebo dokonce jednotková.

Postup je patrný z následujícího příkladu.

6.3.1 Příklad.

Řešme maticovou rovnici $\mathbf{AX} = \mathbf{B}$:

$$\begin{pmatrix} 2 & 1 & 0 \\ 1 & 1 & 2 \\ 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} x_{11} & x_{12} & x_{13} \\ x_{21} & x_{22} & x_{23} \\ x_{31} & x_{32} & x_{33} \end{pmatrix} = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 2 & -1 \\ 1 & 1 & 2 \end{pmatrix}.$$

Řádkové úpravy provádíme s dvojicí matic $(\mathbf{A}, \mathbf{B})$ tak, abychom anulovali všechny prvky každého sloupce matice $\mathbf{A}$ kromě diagonálního prvku. Jednotlivé fáze výpočtu zapisujeme analogicky jako v příkl. 6.2.1:

$$
\begin{matrix} \\ -\frac{1}{2} \\ -\frac{1}{2} \end{matrix}
\left(\begin{array}{ccc|ccc} 2 & 1 & 0 & 1 & 2 & 3 \\ 1 & 1 & 2 & 4 & 2 & -1 \\ 1 & 1 & 1 & 1 & 1 & 2 \end{array}\right)
\overset{\text{1. fáze}}{\longrightarrow}
\begin{matrix} -2 \\ \\ 1 \end{matrix}
\left(\begin{array}{ccc|ccc} 2 & 1 & 0 & 1 & 2 & 3 \\ 1 & \frac{1}{2} & 2 & \frac{7}{2} & 1 & -\frac{5}{2} \\ 0 & \frac{1}{2} & 1 & \frac{1}{2} & 0 & \frac{1}{2} \end{array}\right)
\overset{\text{2. fáze}}{\longrightarrow}
$$

$$
\overset{\text{2. fáze}}{\longrightarrow}
\begin{matrix} -4 \\ 2 \end{matrix}
\left(\begin{array}{ccc|ccc} 2 & 0 & -4 & -6 & 0 & 8 \\ 0 & \frac{1}{2} & 2 & \frac{7}{2} & 1 & -\frac{5}{2} \\ 0 & 0 & -1 & -3 & -1 & 3 \end{array}\right)
\overset{\text{3. fáze}}{\longrightarrow}
$$

$$
\overset{\text{3. fáze}}{\longrightarrow}
\left(\begin{array}{ccc|ccc} 2 & 0 & 0 & 6 & 4 & -4 \\ 0 & \frac{1}{2} & 0 & -\frac{5}{2} & -1 & \frac{7}{2} \\ 0 & 0 & -1 & -3 & -1 & 3 \end{array}\right)
\overset{\text{zpětná substituce}}{\longrightarrow}
$$

$$
\overset{\text{zpětná substituce}}{\longrightarrow}
\left(\begin{array}{ccc|ccc} 1 & 0 & 0 & 3 & 2 & -2 \\ 0 & 1 & 0 & -5 & -2 & 7 \\ 0 & 0 & 1 & 3 & 1 & -3 \end{array}\right).
$$

Přechody $\mathbf{AX} = \mathbf{B}$, $\mathbf{A}^{(1)}\mathbf{X} = \mathbf{B}^{(1)}$, $\mathbf{A}^{(2)}\mathbf{X} = \mathbf{B}^{(2)}$, $\mathbf{A}^{(3)}\mathbf{X} = \mathbf{B}^{(3)}$ lze zapsat rovnostmi:

$$
\left.\begin{aligned}
&\mathbf{r}_1(\mathbf{A}^{(1)}) = \mathbf{r}_1(\mathbf{A}); \quad \mathbf{r}_1(\mathbf{B}^{(1)}) = \mathbf{r}_1(\mathbf{B}); \\
&m_{i1} = -\frac{1}{2}, \quad i = 2, 3; \\
&\mathbf{r}_i(\mathbf{A}^{(1)}) = \mathbf{r}_i(\mathbf{A}) + m_{i1}\mathbf{r}_1(\mathbf{A}); \\
&\mathbf{r}_i(\mathbf{B}^{(1)}) = \mathbf{r}_i(\mathbf{B}) + m_{i1}\mathbf{r}_1(\mathbf{B});
\end{aligned}\right\}
$$

1. fáze (opisuje se 1. řádek a k druhým dvěma řádkům se přičítá příslušný m_{i1}-násobek 1. řádku).

$$
\left.\begin{aligned}
&\mathbf{r}_2(\mathbf{A}^{(2)}) = \mathbf{r}_2(\mathbf{A}^{(1)}); \quad \mathbf{r}_2(\mathbf{B}^{(2)}) = \mathbf{r}_2(\mathbf{B}^{(1)}); \\
&m_{12} = -2; \quad m_{32} = -1; \\
&\mathbf{r}_i(\mathbf{A}^{(2)}) = \mathbf{r}_i(\mathbf{A}^{(1)}) + m_{i2}\mathbf{r}_2(\mathbf{A}^{(1)}), \quad i = 1, 3; \\
&\mathbf{r}_i(\mathbf{B}^{(2)}) = \mathbf{r}_i(\mathbf{B}^{(1)}) + m_{i2}\mathbf{r}_2(\mathbf{A}^{(1)}), \quad i = 1, 3;
\end{aligned}\right\}
$$

2. fáze [opisuje se 2. řádek a ke zbývajícím (dvěma) řádkům se přičítá příslušný m_{i2}-násobek 2. řádku].

$$
\left.\begin{aligned}
&\mathbf{r}_3(\mathbf{A}^{(3)}) = \mathbf{r}_3(\mathbf{A}^{(2)}); \quad \mathbf{r}_3(\mathbf{B}^{(3)}) = \mathbf{r}_3(\mathbf{B}^{(2)}); \\
&m_{13} = -4; \quad m_{23} = 2; \\
&\mathbf{r}_i(\mathbf{A}^{(3)}) = \mathbf{r}_i(\mathbf{A}^{(2)}) + m_{i3}\mathbf{r}_3(\mathbf{A}^{(2)}), \quad i = 1, 2; \\
&\mathbf{r}_i(\mathbf{B}^{(3)}) = \mathbf{r}_i(\mathbf{B}^{(2)}) + m_{i3}\mathbf{r}_3(\mathbf{B}^{(2)}), \quad i = 1, 2;
\end{aligned}\right\}
$$

2. fáze [opisuje se 3. řádek a ke zbývajícím (dvěma) řádkům se přičítá příslušný m_{i3}-násobek 3. řádku].

Protože v rovnici $\mathbf{A}^{(3)}\mathbf{X} = \mathbf{B}^{(3)}$ je matice $\mathbf{A}^{(3)}$ diagonální, dostaneme $\mathbf{X}$ dělením řádků matice $\mathbf{B}^{(3)}$ příslušnými diagonálními prvky matice $\mathbf{A}^{(3)}$ (tato operace je vlastně zpětnou substitucí). Takže

$$
\mathbf{X} = \mathbf{IX} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \mathbf{X} = \begin{pmatrix} 3 & 2 & -2 \\ -5 & -2 & 7 \\ 3 & 1 & -3 \end{pmatrix} = \mathbf{A}^{-1}\mathbf{B}.
$$

6.3.2 Príklad.

Stanovme inverzní matici k matici

$$\mathbf{X} = \begin{pmatrix} 1 & 2 & 6 \\ 2 & 5 & 15 \\ 6 & 15 & 45 \end{pmatrix}.$$

V odstavci 5.3.2 jsme si řekli, že inverzní matice ke čtvercové regulární matici $\mathbf{A}$ je řešením maticové rovnice $\mathbf{AX} = \mathbf{I}$. Využijeme toho v této úloze. Zaznamenáváme pouze sled jednotlivích fází eliminace:

$$\begin{matrix} \\ -2 \\ 6 \end{matrix} \left(\begin{array}{ccc|ccc} 1 & -2 & 6 & 1 & 0 & 0 \\ 2 & 5 & 15 & 0 & 1 & 0 \\ 6 & 15 & 46 & 0 & 0 & 1 \end{array}\right) \rightarrow \begin{matrix} \\ -2 \\ -3 \end{matrix} \left(\begin{array}{ccc|ccc} 1 & -2 & 6 & 1 & 0 & 0 \\ 0 & 1 & 3 & -2 & 1 & 0 \\ 0 & 3 & 10 & -6 & 0 & 1 \end{array}\right) \rightarrow$$

$$\begin{matrix} 0 \\ -3 \\ \\ \end{matrix} \left(\begin{array}{ccc|ccc} 1 & 0 & 0 & 5 & -2 & 0 \\ 0 & 1 & 3 & -2 & 1 & 0 \\ 0 & 0 & 1 & 0 & -3 & 1 \end{array}\right) \rightarrow \left(\begin{array}{ccc|ccc} 1 & 0 & 0 & 5 & -2 & 0 \\ 0 & 1 & 0 & -2 & 10 & -3 \\ 0 & 0 & 1 & 0 & -3 & 1 \end{array}\right) \equiv (\mathbf{I}, \mathbf{A}^{-1}).$$

6.3.3 Príklad.

Chceme řešit "nedourčenou" soustavu (odst. 5.3.4)

$$\begin{aligned} x_1 - 4x_2 + 7x_3 &= 10, \\ x_2 - 2x_3 &= -4 \quad , \end{aligned}$$

maticově

$$\begin{pmatrix} 1 & 4 & 7 \\ 0 & 1 & -2 \\ 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 10 \\ -4 \\ 0 \end{pmatrix}.$$

Abychom mohli stanovit všechna řešení dané soustavy, sestavíme si maticovou rovnici

$$\begin{pmatrix} 1 & -4 & 7 \\ 0 & 1 & -2 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} u_1 & v_1 \\ u_2 & v_2 \\ u_3 & v_3 \end{pmatrix} = \begin{pmatrix} 10 & 0 \\ -4 & 0 \\ 0 & 1 \end{pmatrix}.$$

Nulu na diagonále matice soustavy odpovídající neznámé x_3 nahradíme jedničkou a k původní pravé straně připojíme vektor $(0, 0, 1)^T$. Řešením získané maticové rovnice je matice

$$\mathbf{X} = \begin{pmatrix} -6 & 1 \\ -4 & 2 \\ 0 & 1 \end{pmatrix},$$

jejíž sloupce určují všechna řešení původní nedourčené soustavy, tj.

$$\mathbf{x} = (-6, -4, 0)^T + \alpha(1, 2, 1)^T = \mathbf{u} + \alpha\mathbf{v}.$$

Čtenář se může dosazením přesvědčit, že $\mathbf{u}$ je řešením (původní) nehomogenní soustavy a $\mathbf{v}$ je řešením příslušné homogenní soustavy. Vektor $\mathbf{x} = \mathbf{u} + \alpha\mathbf{v}$ je tzv. *obecné řešení* dané soustavy.

6.3.4 Algoritmus GJEM.

Mějme čtvercovou matici $\mathbf{A}$ řádu n a matici $\mathbf{B}$ typu (n, r), $r \geqq 1$. Pro regulární matici $\mathbf{A}$ bude matice $\mathbf{X} = \mathbf{A}^{-1}\mathbf{B}$ [typu (n, r)] řešením rovnice $\mathbf{AX} = \mathbf{B}$. Algoritmus popsaný v příkl. 6.3.1 (pro $n = 3$, $r = 3$) lze zapsat schematicky naším obvyklým způsobem:

(6.3.3)

$$\begin{aligned}
&\text{Vstup}: \quad n,\ r,\ \mathbf{A} = (a_{ij}^{(0)}), \quad \mathbf{B} = (b_{ij}^{(0)}).\\
&\qquad \text{Pro} \quad k = 1, 2, ..., n-1:\\
&\qquad\qquad \text{Pro} \quad i = 1, 2, k-1, k+1, ..., n:\\
&\qquad\qquad\qquad m_{ik} = -\frac{a_{ik}^{(k-1)}}{a_{kk}^{(k-1)}}.\\
&\qquad\qquad\qquad \text{Pro} \quad s = 1, 2, ..., r:\\
&\qquad\qquad\qquad\qquad b_{is}^{(k)} = b_{is}^{(k-1)} + m_{ik} b_{ks}^{(k-1)}.\\
&\qquad\qquad\qquad \text{Pro} \quad j = k+1, k+2, ..., n:\\
&\qquad\qquad\qquad\qquad a_{ij}^{(k)} = a_{ij}^{(k-1)} + m_{ik} a_{kj}^{(k-1)}.\\
&\qquad \text{Pro} \quad s = 1, 2, ..., r\\
&\qquad\qquad \text{Pro} \quad i = 1, 2, ..., n:\\
&\qquad\qquad\qquad x_{is} = \frac{b_{is}^{(i-1)}}{a_{ii}^{(i-1)}}.\\
&\text{Výstup}: \quad \mathbf{X} = (x_{is}) = \mathbf{A}^{-1}\mathbf{B}.
\end{aligned}$$

Tento algoritmus vyžaduje

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

násobení, stejný počet sčítání, $n(n-1)$ a nr dělení zpětné substituce. Tedy

celkem

$$\frac{n^3}{2} + n^2 r - \frac{n}{2} = \frac{n^3}{2} + O(n^2) \qquad \text{násobení a dělení,}$$
$$\frac{n^3}{2} + n^2(r-1) - nr + \frac{n}{2} = \frac{n^3}{2} + O(n^2) \qquad \text{sčítání.}$$

Ve srování s GEM vychází tedy GJEM z hlediska počtu operací poněkud hůře.

Připomeňme ješzě, že stejně jako GEM nebude pro některé matice $\mathbf{A}$ ani GJEM realizovatelná (viz odst. 6.2.4).

6.4 Eliminace s výběrem hlavního prvku

6.4.1

V odst. 6.2.4 jsme si ukázali, že když v k-té fázi GEM ($k = 1, 2, ..., n-1$) je prvek $a_{kk}^{(k-1)}$ nulový, pak algoritmus GEM (6.2.5) nebude realizovatelný. V příkladu 6.2.7 jsme viděli, že rostl vliv zaokrouhlovacích chyb díky tomu, že $a_{22}^{(1)}$ byl velmi malý a došlo k růstu čísel $a_{33}^{(2)}$, $a_{34}^{(2)}$.

Provedené (značně hluboké a pracné) analýzy realizace algoritmu GEM (resp. GJEM) na počítači pracujícím s čísly z $M(q,t)$ ukazují, že vliv zaokrouhlovacích chyb na celý proces může být katastrofální, pokud čísla $a_{ij}^{(k)}$, $i, j > k$, během výpočtu prudce porostou.

Prvek, jehož prostřednictvím určujeme multiplikátory k-té fáze, budeme nazývat *hlavní prvek* (*pivot*) k-té fáze eliminace. V algoritmu z odst. 6.2.3 to byl prvek $a_{kk}^{(k-1)}$.

Rovnici, z níž vybíráme hlavní prvek k-té fáze (tj. jejíž násobky přičítáme k ostatním rovnicím), nazýváme *hlavní rovnicí* k-té fáze.

Abychom tedy minimilizovali vliv zaokrouhlovacích chyb, je vhodné vybírat za hlavní prvky takové prvky matice $\mathbf{A}$, které mají co největší absolutní hodnotu. Potom hlavní rovnicí násobíme čísly nejvýše rovnými jedné (v absolutní hodnotě) a každá dílčí zaokrouhlovací chyba se také násobí tímto číslem (a tedy se nezvětšuje).

Vybíráme-li hlavní prvek ze všech prvků, které v dané fázi přicházejí v úvahu, hovoříme o algoritmu GEM *s úplným výběrem* - ozn. GEMPU. Vybíráme-li hlavní prvek pouze z některých prvků, které v dané fázi přicházejí v úvahu (např. z jednoho řádku, resp. u jednoho sloupce), hovoříme a algoritmu GEM *s částečným výběrem* - ozn. GEMPR, resp. GEMPS.

6.4.2 Příklad.

Řešme soustavu [v $M(10,3)$]

$$\begin{aligned} 0,000\,100 \quad & x_1 + 1,00x_2 = 1,00, \\ 1,00 \quad & x_1 + 1,00x_2 = 2,00. \end{aligned}$$

Poznamenejme, že daná soustava je určitým způsobem patologická, umožní nám však ukázat citlivost užitých algoritmů na zaokrouhlovací chyby.

Aplikujeme-li algoritmus GEM z odst. 6.2.3 [tento algoritmus nazýváme algoritmem GEM bez výběru hlavního prvku - hlavní prvek "vybíráme" v přirozeném pořadí, tj. vždy v pozici (k,k)], bude $m_{21} = 10^4$ a tento násobek první (hlavní) rovnice přičteme k druhé rovnici. Redukovaná soustava bude mít tvar

$$\begin{aligned} 0,000\,100x_1 + 1,00x_2 &= 1,00 \\ -10\,000x_2 &= -10\,000. \end{aligned}$$

Odtud

$$\mathbf{x_c} = (0,000; 1,00)^T.$$

Jestliže však hlavní prvek zvolíme číslo v pozici $(2,1)$ (hlavní rovnicí 1. fáze bude tedy 2. rovnice) a k první rovnici přičteme m_{21}-násobek druhé rovnice, kde $m_{21} = -0,000\,100/1,00 = -0,000\,100$, dostaneme redukovanou soustavu ve tvaru

$$\begin{aligned} 1,00x_2 &= 1,00, \\ 1,00x_1 + 1,00x_2 &= 2,00. \end{aligned}$$

Jejím řešením je vektor

$$\mathbf{x_c} = (1,00; 1,00)^T.$$

Porovnání s přesnějším řešením [v $M(10,6)$] původní soustavy

$$\mathbf{x}_t = (1,000\,10; 0,999\,90)^T$$

nás přesvědčuje o tom, který z použitých algoritmů (tj. GEM bez výběru a GEMPS) se ukázal jako lepší (z hlediska přesnosti).

6.4.3 Sloupcový výběr hlavního prvku - GEMPS.

V k-té fázi eliminace ($k = 1,2,...,n-1$) vybíráme za hlavní prvek takový, který má největší absolutní hodnotu z těch prvků k-tého sloupce patřících zbývajícím $n-k+1$ řádkům, které do k-té fáze nebyly hlavními řádky.

Necht' je to prvek v pozici (p, k), tj.

$$|a_{pk}^{(k-1)}| = \max_i |a_{ik}^{(k-1)}|.$$

kde i probíhá řádkové indexy těch řádků, které do k-té fáze nebyly hlavními řádky. Pokud v k-tém sloupci je vice prvků s největší absolutní hodnotou, vezme se (např.) ten, který má nejmenší řádkový index, a tím rozhodneme o volbě hlavní rovnice. Výsledná matice redukované soustavy po všech $n-1$ krocích nebude mít trojúheníkový tvar upravit vhodným přerovnáním řádků (v počítači přečíslování řádkových indexů).

6.4.4 Řádkový výběr hlavního prvku - GEMPR.

V k-té fázi eliminace vybereme q jako nejmenší (sloupcový) index, pro který

$$|a_{kq}^{(k-1)}| = \max_i |a_{kj}^{(k-1)}|.$$

přičemž j probíhá sloupcové indexy těch zbývajících $n-k+1$ sloupců, které do k-té fáze neobsahovaly hlavní prvky předcházejících fází. Výsledná matice redukované soustavy po všech $n-1$ krocích opět nebude horní trojúhelníková, ale vhodným přerovnáním sloupců ji lze na trojúhelníkový tvar upravit (provede se příslušné přečíslování neznámých).

6.4.5 Úplný výběr hlavního prvku - GEMPU.

V k-té fázi eliminace vybereme p a q jako nejmenší indexy, pro které

$$|a_{pq}^{(k-1)}| = \max_{i,j} |a_{ij}^{(k-1)}|,$$

přičemž i, j probíhá ty indexy, které nejsou rovny indexům p, q z předcházejících fází eliminace.

Poznamenejme, že pro větší n je vyhledávání hlavního prvku s největší absolutní hodnotou náročně na čas (a tedy drahé), a proto není vždy výhodné užívat algoritmu GEMPU. Navíc u celé řady soustav, které se v praxi často vyskytují (viz odst. 6.6.1), dává algoritmus GEM zcela srovnatelné výsledky a laciněji.

6.4.6 Příklad.

Uvažujme úlohu stanovit řešení soustavy

$$\begin{aligned} 0,980x_1 + 0,990x_2 &= 1,97, \\ 0,990x_1 + 1,00x_2 &= 1,99. \end{aligned}$$

Tato úloha byla vyšetřována v odst. 4.2.3 (změněno pouze číslování neznámých a pořadí rovnic) a bylo ukázáno, že je špatně podmíněná. Přesným řešením je $\mathbf{x}_t = (1,1)^T$. Řešíme-li tuto soustavu algoritmem GEM bez výběru [v $M(10,3)$ s řezáním], bude hlavním prvkem číslo $0,980$ [v pozici $(1,1)$]. Potom $m_{21} = -1,010$ a redukovaná soustava bude mít tvar

$$\begin{aligned} 0,980x_1 + 0,990x_2 &= 1,97, \\ -0,000\,100x_2 &= 0,000\,300. \end{aligned}$$

Vypočtené řešení $\mathbf{x_c} = (5,04; -3,00)^T$ se dost podstatně liší od přesného řešení, i když vektor rezidua $\mathbf{r} = \mathbf{b} - \mathbf{A}\mathbf{x_c} = (0,000\,800; 0,000\,400)^T$ takovou odchylku na první pohled nesignalizuje.

Počítáme-li algoritmem GEMPU, pak za hlavní prvek volíme $\max\limits_{i,j} |a_{ij}| = a_{22} = 1,00$. Potom $m_{22} = -0,990$ a redukovaná soustava bude mít tvar

$$\begin{aligned} -0,000\,100x_1 &= -0,000\,100, \\ 0,990x_1 + 1,00x_2 &= 1,99. \end{aligned}$$

Odtud vypočteme $\mathbf{x_c} = (1,00; 1,00)^T$. Dokonce jsme dostali přesné řešení $\mathbf{x}_t$.

Daná soustava má ovšem podobný charakter jako soustava z příkl. 6.4.2, nicméně je zde patrné, jak užitý algoritmus podstatnou měrou ovlivňuje přesnost výsledku.

6.4.7 Příklad.

Algoritmem GEMPU řešme soustavu rovnic

$$\begin{aligned} 0,197x_1 + 0,305x_2 - 0,206x_3 - 0,084x_4 &= 0,136, \\ 0,468x_1 + 0,713x_2 - 0,474x_3 + 0,052x_4 &= 0,117, \\ 0,886x_1 + 0,764x_2 - 0,108x_3 + 0,802x_4 &= 0,251, \\ 0,145x_1 + 0,590x_2 + 0,613x_3 + 0,365x_4 &= 0,66. \end{aligned}$$

Počítáme v $M(10,4)$ a výsledky zapisujeme do tabulky (viz tab. 4). Konečná redukovaná soustava je "uložena" ve 3., 7., 9., 10. řádku. (Hlavní prvky jsou

vytištěny tučně.)

Tab. 4

	m	**A**				**b**	$\sum$
1	-0,222 3	0,197 0	0,305 0	-0,206 0	-0,084 00	0,136 0	0,348
2	-0,528 2	0,468 0	0,713 0	-0,474 0	0,052 00	0,117 0	0,876
3		**0,886 0**	0,764 0	-0,108 0	0,802 0	0,251 0	2,595
4	-0,163 7	0,145 0	0,590 0	0,613 0	0,365 0	0,660 0	2,373
5	0,288 6		0,135 2	-0,182 0	-0,262 3	0,080 2	-0,228 9
6	0,661 2		0,309 5	-0,417 0	-0,371 6	-0,015 6	-0,494 7
7			0,464 9	**0,630** 7	0,233 7	0,618 9	1,948 2
8	-0,436 7		0,269 4		-0,194 9	0,258 8	0,333 3
9			**0,616 9**		-0,217 1	0,393 6	0,793 4
10					**-0,100 1**	0,086 9	-0,013 2

Odtud

$$x_4 = -0,868\,1, \quad x_3 = 1,058, \quad x_2 = 0,332\,5, \quad x_1 = 0,911\,3.$$

6.5 Metoda LU-rozkladu

6.5.1 Příklad

Chceme vypočítat prvky matic

$$\mathbf{L} = \begin{pmatrix} 1 & 0 & 0 \\ l_{21} & 1 & 0 \\ l_{31} & l_{32} & 1 \end{pmatrix}, \qquad \mathbf{U} = \begin{pmatrix} u_{11} & u_{12} & u_{13} \\ 0 & u_{22} & u_{23} \\ 0 & 0 & u_{33} \end{pmatrix}$$

tak aby platila rovnost

$$\begin{pmatrix} 1 & 0 & 0 \\ l_{21} & 1 & 0 \\ l_{31} & l_{32} & 1 \end{pmatrix} \begin{pmatrix} u_{11} & u_{12} & u_{13} \\ 0 & u_{22} & u_{23} \\ 0 & 0 & u_{33} \end{pmatrix} = \begin{pmatrix} 2 & 5 & 6 \\ 4 & 13 & 19 \\ 6 & 27 & 50 \end{pmatrix}.$$

Podle definice součinu matic musí být

$$\left.\begin{aligned} 2 &= \mathbf{r}_1(\mathbf{L})\mathbf{s}_1(\mathbf{U}) = 1 \cdot u_{11} \Rightarrow u_{11} = 2, \\ 4 &= \mathbf{r}_2(\mathbf{L})\mathbf{s}_1(\mathbf{U}) = l_{21}u_{11} \Rightarrow l_{21} = 2, \\ 6 &= \mathbf{r}_3(\mathbf{L})\mathbf{s}_1(\mathbf{U}) = l_{31}u_{11} \Rightarrow l_{31} = 3, \end{aligned}\right\} \qquad \mathbf{s}_1(\mathbf{L}) = \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix}, \quad \mathbf{s}_1(\mathbf{U}) = \begin{pmatrix} 2 \\ 0 \\ 0 \end{pmatrix},$$

$$\left.\begin{aligned} 5 &= \mathbf{r}_1(\mathbf{L})\mathbf{s}_2(\mathbf{U}) = 1 \cdot u_{12} \Rightarrow u_{12} = 5, \\ 13 &= \mathbf{r}_2(\mathbf{L})\mathbf{s}_2(\mathbf{U}) = l_{21}u_{12} + u_{22} \Rightarrow u_{22} = 3, \\ 27 &= \mathbf{r}_3(\mathbf{L})\mathbf{s}_2(\mathbf{U}) = l_{31}u_{12} + l_{32}u_{22} \Rightarrow l_{32} = 4, \end{aligned}\right\} \mathbf{s}_2(\mathbf{L}) = \begin{pmatrix} 0 \\ 1 \\ 4 \end{pmatrix}, \mathbf{s}_2(\mathbf{U}) = \begin{pmatrix} 5 \\ 3 \\ 0 \end{pmatrix},$$

$$\left.\begin{aligned} 6 &= \mathbf{r}_1(\mathbf{L})\mathbf{s}_3(\mathbf{U}) = 1 \cdot u_{13} \Rightarrow u_{13} = 6, \\ 19 &= \mathbf{r}_2(\mathbf{L})\mathbf{s}_3(\mathbf{U}) = l_{21}u_{13} + u_{23} \Rightarrow u_{23} = 7, \\ 50 &= \mathbf{r}_3(\mathbf{L})\mathbf{s}_3(\mathbf{U}) = l_{31}u_{13} + l_{32}u_{23} + u_{33} \Rightarrow u_{33} = 4, \end{aligned}\right\} \qquad \mathbf{s}_3(\mathbf{L}) = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix},$$

$$\mathbf{s}_3(\mathbf{U}) = \begin{pmatrix} 6 \\ 7 \\ 4 \end{pmatrix}.$$

Takže máme

$$\begin{pmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ 3 & 4 & 1 \end{pmatrix} \begin{pmatrix} 2 & 5 & 6 \\ 0 & 3 & 7 \\ 0 & 0 & 4 \end{pmatrix} = \begin{pmatrix} 2 & 5 & 6 \\ 4 & 13 & 19 \\ 6 & 27 & 50 \end{pmatrix}.$$

Počítali jsme matice $\mathbf{L}$ a $\mathbf{U}$ postupně po sloupcích. Tento postup má tu výhodu, že umožňuje takovou úpravu algoritmu, která je ekvivalentní GEMPS.

6.5.2 Výklad metody.

V odstavci 6.2.4 jsme si uvedli, za jakých podmínek lze danou regulární matici $\mathbf{A}$ rozložit na součin trojúhelníkových matic $\mathbf{L} = (l_{ij})$ a $\mathbf{U} = (u_{ij})$, tj. kdy platí

$$\mathbf{A} = \mathbf{LU}. \tag{6.5.1}$$

Metoda řešení soustavy lineárních rovnic LU-rozkladem spočívá v tom, že nejdříve stanovíme matice $\mathbf{L}$ a $\mathbf{U}$ a potom řešíme dvě soustavy $\mathbf{Ly} = \mathbf{b}$, $\mathbf{Ux} = \mathbf{y}$.

V dolní trojúhelníkové matici $\mathbf{L}$ volíme na diagonále jedničky, tj. $l_{ii} = 1$, $i = 1, 2, ..., n$, a $\mathbf{U}$ je horní trojúhelníková matice.

Ze vztahu (6.5.1) plyne

$$a_{ij} = \mathbf{r}_i(\mathbf{L})\mathbf{s}_j(\mathbf{U})$$

Odtud pro $i \leqq j$ máme

$$a_{ij} = l_{i1}u_{1j} + l_{i2}u_{2j} + ... + l_{i,i-1}u_{i-1,j} + 1 \cdot u_{ij} \tag{6.5.2}$$

a pro $i > J$ máme

$$a_{ij} = l_{i1}u_{1j} + l_{i2}u_{2j} + ... + l_{i,j-1}u_{j-1,j} + l_{ij}u_{jj}. \tag{6.5.3}$$

V těchto vzorcích je $l_{ik} = 0$, pokud $i < k$, a $u_{kj} = 0$, když $k > j$.

Postupem uvedeným v příkl. 6.5.1 vypočteme u_{ij} (pro $i = 1, 2, ..., j$) ze vztahu (6.5.2) a l_{ij} (pro $i = j+1, j+2, ..., n$) ze vztahu (6.5.3).

Pokud v rozkladu (6.5.1) volíme jedničky na diagonále matice $\mathbf{U}$ (pak obecně matice $\mathbf{L}$ jedničky na diagonále mít nebude), hovoříme o *Croutově metodě*.

V souvislosti s metodou LU-rozkladu námi popsáného typu se někdy hovoří o *Doolittleově metodě* (viz [12]).

6.5.3 Algoritmus LU-rozkladu.

Výpočét provádíme a do paměti ukládáme po sloupcích. (Klademe $\sum_{j=k}^{s} a_j = 0$, když $k > s$.)

(6.5.4)

Vstup : n, $\mathbf{A} = (a_{ij})$.

Pro $j = 1, 2, ..., n$:

Pro $i = 1, 2, ..., j$:

$$u_{ij} = a_{ij} - \sum_{r=1}^{i-1} l_{ir} u_{rj}.$$

Pro $i = j+1, j+2, ..., n$:

$$l_{ij} = \frac{1}{u_{jj}} \left(a_{ij} - \sum_{s=1}^{j-1} l_{is} u_{sj} \right).$$

Tab. 5

	P_1	P_2	P_3	P_4	D_1	D_2	D_3	D_4	D_5	$\sum$
1					a_{11}	a_{12}	a_{13}	a_{14}	a_{15}	$\sigma_1 = \sum_k a_{1k}$
2					a_{21}	a_{22}	a_{23}	a_{24}	a_{25}	$\sigma_2 = \sum_k a_{2k}$
3					a_{31}	a_{32}	a_{33}	a_{34}	a_{35}	$\sigma_3 = \sum_k a_{3k}$
4					a_{41}	a_{42}	a_{43}	a_{44}	a_{45}	$\sigma_4 = \sum_k a_{4k}$
5	-1	$-l_{21}$	$-l_{31}$	$-l_{41}$	u_{11}	u_{12}	u_{13}	u_{14}	u_{15}	$\tilde{\sigma}_1 \approx \sum_k u_{1k}$
6		-1	$-l_{32}$	$-l_{42}$		u_{22}	u_{23}	u_{24}	u_{25}	$\tilde{\sigma}_2 \approx \sum_k u_{2k}$
7			-1	$-l_{43}$			u_{33}	u_{34}	u_{35}	$\tilde{\sigma}_3 \approx \sum_k u_{3k}$
8				-1				u_{44}	u_{45}	$\tilde{\sigma}_4 \approx \sum_k u_{4k}$

$-\mathbf{L}^T$ $\qquad$ $\mathbf{U}$

Realizace tohoto algoritmu vyžaduje $n^3/3 + O(n^2)$ násobení a stejný počet sčítání a $n(n-1)/2$ dělení.

Výhody metody LU-rozkladu vyniknou tehdy, řešíme-li více soustav se stejnou maticí soustavy. Provedeme rozklad matice $\mathbf{A}$ a pak již opakovaně řešíme pouze soustavy s trojúhelníkovými maticemi ($\approx n^2$ operací).

V knize [12] je podrobně popsána (str. 447) modifikace uvedeného algoritmu s užitím částečného výběru hlavního prvku.

V knize [7a] je přímo uveden program (ve Fortranu) jak LU-rozkladu (subroutine DECOMP), tak řešení rovnic $\mathbf{Ly} = \mathbf{b}$, $\mathbf{Ux} = \mathbf{y}$ (subroutine SOLVE).

6.5.4 LU-rozklad pomocí kalkulátoru.

Stejné důvody jako v odst. 6.2.6 nás vedou k tomu, abychom su ukázali, jak upravit algoritmus LU-rozkladu pro ruční výpočet. Výhodou bude (proti Gaussově eliminaci), že vapočítáváme a zapisujeme pouze ta čísla, která jsou už v každé fázi současně konečnými výsledky. V tabulce 5 vidíme způsob ulození výsledků: V sloupci D_5 (volíme $n = 4$) je uložen vektor $\mathbf{b}$ a vektor $\mathbf{L}^{-1}\mathbf{b} = (u_{1,n+1}, u_{2,n+1}, ..., u_{n,n+1})^T$, abychom mohli soustavu $\mathbf{Ax} = \mathbf{b}$ řešit přímo jako $\mathbf{Ux} = \mathbf{L}^{-1}\mathbf{b}$.

Zaplňování tabulky (algoritmus): Symbolem $P_i \cdot D_j$ označujeme součin sloupců jako vektorů uložených v 2. části tabulky - jedničky do nich nezahrnujeme! Přitom užijeme k výpočtu těch čísel těchto sloupců, která jsou v dané fázi vypočítána.

$U(1):\ u_{1j} \leftarrow a_{1j},\ j = 1, 2, ..., n, n+1;$ (opíšeme 1. řádek do 5. řádku);

$L(1):\ l_{k1} \leftarrow \dfrac{a_{k1}}{u_{11}},\ k = 2, 3, ..., n;$ (dělíme "D_1" číslem u_{11} a výsledek píšeme s opačným znaménkem);

$U(2):\ u_{2j} \leftarrow P_2 \cdot D_j + a_{2j},\ j = 2, 3, ..., n, n+1;$ (v P_2, D_j jsou zatím po jednom čísle);

$L(2):\ l_{k2} \leftarrow \dfrac{P_k \cdot D_2 + a_{k2}}{u_{22}},\ k = 3, 4, ..., n;$ (v P_k ke také zatím po jednom čísle);

$U(3):\ u_{3j} \leftarrow P_3 \cdot D_j + a_{3j},\ j = 3, 4, ..., n, n+1;$

$L(3):\ l_{k3} \leftarrow \dfrac{P_k \cdot D_3 + a_{k3}}{u_{33}};\ k = 4, ..., n;$

. . . . . .

$U(n-1):\ u_{n-1,j} \leftarrow P_{n-1} \cdot D_j + a_{n-1,j},\ j = n-1, n, n+1;$

$L(n-1):\ l_{n,n-1} \leftarrow \dfrac{P_n \cdot D_{k-1} + a_{n,n-1}}{u_{n-1,n-1}},$

$U(n):\ u_{nj} \leftarrow P_n \cdot D_j + a_{nj};\ j = n, n+1;$

Sloupec $\sum$ slouží ke kontrole přesnosti výpočtů (jako v odst. 6.2.6). Čísla $\sigma_i = \sum\limits_k a_{ik}$, $\sum\limits_k u_{ik}$ určíme sečtením čísel v i-tém a v $(n+i)$-tém řádku tab. 5. Čísla $\tilde{\sigma}_i$ vypočteme podle instrukcí U_i , tj. $\tilde{\sigma}_i \leftarrow P_i \sum + \sigma_i$ (např. $\tilde{\sigma}_2 = -l_{21}\tilde{\sigma}_1 + \tilde{\sigma}_2$, $\tilde{\sigma}_3 = -l_{31}\tilde{\sigma}_1 - l_{32}\tilde{\sigma}_2 + \sigma_3$). Nelze-li rozdíl $\tilde{\sigma}_i - \sum\limits_k u_{ik}$ vysvětlit zaorouhlovacími chybami, máme ve výpočtu chybu (omyl).

6.5.5 Příklad.

Na následujících dvou soustavách si čtenář může utvrdit mechanismus výpočtu z předchozího odstavce.

a) Matice soustavy typu $(4, 4)$ je uložena ve sloupcích D_1 až D_4 první části tab. 6. Sloupec kontrolních součtů nepíšeme.

b) Matice soustavy je uložena ve sloupcích D_1, D_2, D_3 1. části tab. 7.

Tab. 6

P_1	P_2	P_3	P_4	D_1	D_2	D_3	D_4	D_5
				2	-1	3	-1	7
				1	-1	4	-2	5
				3	2	1	4	31
				4	-3	3	-3	-5
-1	$-\frac{1}{2}$	$-\frac{3}{2}$	-2	2	-1	3	-1	7
	-1	7	-2		$-\frac{1}{2}$	$\frac{5}{2}$	$-\frac{3}{2}$	$\frac{3}{2}$
		-1	$\frac{4}{7}$			14	-5	31
			-1				$-\frac{6}{7}$	$-\frac{30}{7}$
			$-\mathbf{L}^T$			$\mathbf{U}$		

$$\mathbf{x} = \begin{pmatrix} 1 \\ 2 \\ 4 \\ 5 \end{pmatrix}.$$

Tab. 7

P_1	P_2	P_3	D_1	D_2	D_3	D_4	$\sum$
			3,6	-8,7	15,3	0,6	10,8
			0,9	2,5	6,4	11,0	20,8
			3,7	4,8	-2,5	7,8	13,8
-1	-0,250 0	-1,028	3,600	-8,700	15,30	0,600 0	10,80
	-1	-2,940		4,675	2,575	10,85	18,10
		-1			-25,80	-24,72	-50,52

$$\mathbf{x} = (0,427\,8; 1,793; 0,958\,1)^T.$$

6.6 Soustavy se speciální maticí

6.6.1 Speciální matice.

Speciální maticí budeme rozumět takovou čtvercovou matici, která má aspoň jednu z následujících vlastností:

(1) symetričnost a pozitivní definitnost,

(2) diagonální dominantnost,

(3) pásovost.

Přistupme k definicím:

Matice $\mathbf{A} = (a_{ij})$ s reálnými prvky je *symetrická*, platí-li

$$\mathbf{A}^T = \mathbf{A}, \quad \text{tj.} \quad a_{ij} = a_{ji}. \tag{6.6.1}$$

Matice $\mathbf{A} = (a_{ij})$ s komlexními prvky je *hermitovská* (*hermitovsky symetrická*), platí-li

$$\mathbf{A}^H = \mathbf{A}, \quad \text{tj.} \quad a_{ij} = \bar{a}_{ji}. \tag{6.6.1'}$$

Symetrická matice $\mathbf{A} = (a_{ij})$ je *pozitivně definitní*, platí-li

$$\mathbf{x}^T \mathbf{A} \mathbf{x} = \sum_{i=1}^{n} \sum_{j=1}^{n} a_{ij} x_i x_j > 0 \tag{6.6.2}$$

pro všechny nenulové vektory $\mathbf{x} = (x_1, x_2, ..., x_n)^T$.

Matice $\mathbf{A} = (a_{ij})$ je *diagonálně dominantní* (má *diagonální převachu*), platí-li

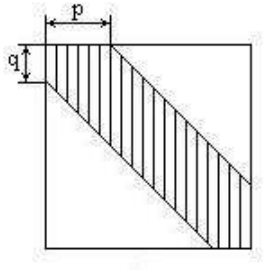
$$|a_{ii}| \geqq \sum_{\substack{j=1 \\ j \neq i}}^{n} |a_{ij}|, \quad i = 1, 2, ..., n. \tag{6.6.3}$$

Matice $\mathbf{A} = (a_{ij})$ je *pasová* dají-li se najít přirozená čísla p, q taková, že

$$a_{ij} = 0, \quad \text{když} \quad j > i + p \quad \text{nebo} \quad i > j + q.$$

Číslo $w = p + q + 1$ se nazývá *šířka pásu*.

Na obr. 3 vidíme význam čísel p, q. Je-li $p = q = 1$, pak $w = 3$ a hovoříme o *třídiagonální matici*, je-li $p = q = 2$, o *pětidiagonální matici* ($w == 5$), atd. Počet nenulových prvků v kazdém řádku nebo sloupci nepřevýší w a celkem je počet nenulových prvků menší než wn.



Obr. 3

Poznamenejme, že se speciálními maticemi u soustav lineárních rovnic se můžeme např. setkat při aplikaci metody sítí nebo metody konečných prvků k okrajovým úlohám pro obyčejné nebo parciální diferenciální rovnice (viz [10], [10a]). Ovšem nejen zde: v [8] nebo v [11] se čtenář může seznámit s metodou interpolace pomocí tzv. splinových funkcí, která vede na řešení soustavy s třídiagonální diagonálně dominantní maticí.

6.6.2 Soustavy se symetrickou a pozitivně definitní maticí.

Bez důkazu si uved'me tvrzení (důkazy např. v [18], [6], [9], [12]):

(1) *Symetrická matice* $\mathbf{A}$ *je pozitivně definitní, právě když* $\det \mathbf{A}_k > 0$, $k = 1, 2, ..., n$, *kde* $\mathbf{A}_k$ *je matice vzniklá z* $\mathbf{A}$ *vynecháním posledních* $n - k$ *řádků a* $n - k$ *sloupců* (*Sylvestrovo kritérium*).

(2) *Největší prvek symetrické pozitivně definitní matice leží na hlavní diagonále.*

(3) *Reálná symetrická matice* $\mathbf{A}$ *má reálná vlastní čísla. Je-li* $\mathbf{A}$ *navíc pozitivně definitní, jsou její vlastní čísla dokonce kladná.*

(4) *Je-li* $\mathbf{A}$ *symetrická matice a* $\mathbf{A}^{(k)}$ *redukovaná matice získaná Gaussovou eliminaci bez výběru hlavního prvku* (*odst.* 6.2.4), *potom ta část matice* $\mathbf{A}^{(k)}$, *jejíž prvky patří zbývajícím* $n - k$ *řádkům a slopcům, je symetrická matice* (*dokonce je pozitivně definitní, pokud* $\mathbf{A}$ *je pozitivně definitní*).

Tvrzení (2), (4) se dají dokázat pomocí tvrzení (1). Na základě tvrzení (4) můžeme upravit algoritmus GEM (6.2.5) tak, abychom počítali pouze prvky na halvní diagonále a na ní. Takže ve vnitřním cyklu (6.2.5) stačí brát $j = i, i + 1, ..., n, n + 1$.

Tato tzv. *symetrická Gaussova eliminace* (SGEM) vyžaduje zhruba polovinu aritmetických operací proti GEM a počet nutných pamět'ových míst se též sníží přibližně na polovinu.

Stejně jako v odst. 6.2.4 lze ukázat, že algoritmus SGEM není realizovatelný pro každou symetrickou matici, případně může být numericky nestabilní (citilivý na zaokrouhlovací chyby), nebot' neprovádíme výběr hlavního prvku.

Z tvrzení (2) však vyplývá, že pro symetrické pozitivně definitní matice je algoritmus SGEM vždy realizovatelný a je numericky stabilní.

Všimneme si nyní, jak se bude modifikovat metoda LU-rozkladu předpokladem, že matice $\mathbf{A}$ je symetrická. Platí-li následující varianta věty o rozkladu ([4]):

Je-li $\mathbf{A}$ *symetrická pozitivně definitní matice, potom existuje jediná horní trojúhelníková matice* $\mathbf{U}$ *s kladnými diagonálními prvky taková, že*

$$\mathbf{A} = \mathbf{U}^T\mathbf{U}.$$

Analogicky jako v odst. 6.5.2 řešíme místo soustavy $\mathbf{Ax} = \mathbf{b}$ dvě soustavy $\mathbf{U}^T\mathbf{y} = \mathbf{b}$, $\mathbf{Ux} = \mathbf{y}$.

V tomto případě můžeme algoritmus LU-rozkladu z odst. 6.5.3 upravit

na tzv. *Choleského algoritmus* (po sloupcích):

$$
\begin{aligned}
&\text{Vstup}: \quad n, \quad \mathbf{A} = (a_{ij}).\\
&\qquad \text{Pro} \quad i = 1, 2, ..., n:\\
&\qquad\qquad u_{jj} = \sqrt{\left(a_{jj} - \sum_{r=1}^{j-1} u_{jr}^2\right)}.\\
&\qquad\qquad \text{Pro} \quad i = j+1, j+2, ..., n:\\
&\qquad\qquad\qquad u_{ij} = \frac{1}{u_{jj}} \left(a_{ij} - \sum_{s=1}^{j-1} u_{is} u_{sj}\right).\\
&\text{Výstup}: \quad \mathbf{U} = (u_{ij}) \ \left[\mathbf{U}^T = (u_{ji}),\ u_{ji} = u_{ij}\right].
\end{aligned}
\tag{6.6.4}
$$

Pokud odmocnění počítáme jako tří násobení ([12]), pak uvedený algoritmus vyžaduje $n^3/3 + n^2 + 11n/6$ násobení a dělení.

Nemáme-li zaručenou pozitivní definitnost matice $\mathbf{A}$, mohou čísla u_{ii} v Choleského algoritmu vyjít komplexní, avšak nediagonální prvky jsou reálné nebo ryze imaginární (viz [12]).

Počet operací Choleského algoritmu je zhruba poloviční ve srovnání s algoritmem (6.5.4). Totéž platí o počtu nutných pamet'ových míst.

6.6.3 Poznámka.

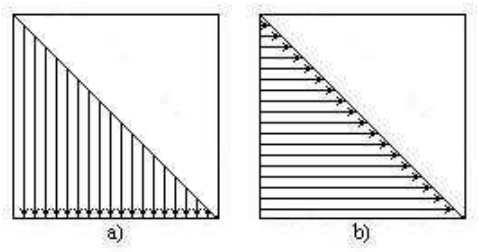
Algoritmus (6.6.4) je vhodný v těch případech, kdy matice $\mathbf{A}$ je v paměti počítače uložena po sloupcích (máme na mysli především velké matice). Potom se bude matice $\mathbf{U}^T$ též ukládat po sloupcích.

Pokud je naopak matice $\mathbf{A}$ uložena po řádcích, pak je vhodnější algoritmus.

$$
\begin{aligned}
&\text{Pro} \quad i = 1, 2, ..., n:\\
&\qquad u_{ii} = \sqrt{\left(a_{ii} - \sum_{k=1}^{i-1} u_{ki}^2\right)}.\\
&\qquad \text{Pro} \quad j = i+1, i+2, ..., n:\\
&\qquad\qquad u_{ij} = \frac{1}{u_{ii}} \left(a_{ij} - \sum_{k=1}^{i-1} u_{ki} u_{kj}\right).
\end{aligned}
$$

V tomto případě je $\mathbf{U}^T$ uložena také po řádcích. Obě situace jsou schematicky znázorněny na obr. 4 (obr. 4a - uložení po sloupcích, obr. 4b - uložení

po řádcích).



Obr. 4

6.6.4 Soustavy s pásovou maticí.

Uvažujme matici $\mathbf{A} = (a_{ij})$, pro niž $a_{ij} = 0$, když $|i - j| > p$; zde $w = 2p + 1$ je šířka pásu ($q = p$, viz obr. 3). Lze-li provést LU-rozklad takové matice, zjistíme, že

$$l_{ij} = 0, \quad \text{když} \quad j > i \quad \text{a když} \quad j < i - p,$$

a

$$u_{ij} = 0, \quad \text{když} \quad i > j \quad \text{a když} \quad i < j - p.$$

To znamená, že matice $\mathbf{L}$ a $\mathbf{U}$ budou pásové. Aplikujeme-li metodu LU-rozkladu k soustavě $\mathbf{Ax} = \mathbf{b}$ s regulární maticí $\mathbf{A}$, můžeme analogicky jako v odst. 6.5.3 odvodit algoritmus (po sloupcích) řešení této soustavy (tj. včetně zpětného chodu). U tohoto algoritmu je počet násobení a dělení roven číslu $np(p + 1)$ a pro zpětnou substituci $n(2p + 1)$. V případě, že p je mnohem menší než n, jde o podstatnou úsporu počtu operací. Při vhodné organizaci paměti uspoříme i pamět'ová místa. Pro srovnání připomeňme, že pro plné matice jsou počty těchto operací (řádově) dány čísly $n^3/3$ pro LU-rozklad a n^2 pro zpětnou substituci. Tato úspora počtu operací a pamet'ových míst (nuly mimo pás se do paměti neukládají) umožňuje při dané kapacitě řešit mnohem rozsáhlejší soustavy rovnic.

Poznamenejme, že algoritmus (6.6.3) se ještě zjednoduší, pokud matice $\mathbf{A}$ je třídiagonální ($p = 1$). Doporučuji čtenáři, aby si zjednodušenou verzi algoritmu (6.6.3) sám odvodil.

Po sloupcích:

(6.6.3)

$$
\begin{aligned}
&\text{Vstup}: \quad n,\ p,\ \mathbf{A},\ \mathbf{b}.\\
&\qquad \text{Pro} \quad j = 1, 2, ..., n:\\
&\qquad\quad \alpha = \max(1, j-p);\\
&\qquad\quad \gamma = \min(1, j+p);\\
&\qquad\quad \text{Pro} \quad i = \alpha, \alpha+1, ..., j:\\
&\qquad\qquad u_{ij} = a_{ij} - \sum_{r=\alpha}^{i-1} l_{ir} u_{rj}.\\
&\qquad\quad \text{Pro} \quad i = j+1, j+2, ..., \gamma:\\
&\qquad\qquad \beta = \max(1, j-p);\\
&\qquad\qquad l_{ij} = \frac{1}{u_{jj}} \left(a_{ij} - \sum_{s=\beta}^{j-1} l_{is} u_{sj} \right).
\end{aligned}
$$

$$
\begin{aligned}
&\text{Pro} \quad j = 1, 2, ..., n: \quad \text{(řešení rovnice } \mathbf{Ly} = \mathbf{b})\\
&\qquad \alpha = \max(1, j-p);\\
&\qquad y_j = b_j - \sum_{k=\alpha}^{j-1} l_{jk} y_k.
\end{aligned}
$$

$$
\begin{aligned}
&\text{Pro} \quad j = n, n-1, ..., 2, 1: \quad \text{(řešení rovnice } \mathbf{Ux} = \mathbf{y})\\
&\qquad \gamma = \min(n, j+p);\\
&\qquad x_j = \frac{1}{u_{jj}} \left(y_j - \sum_{k=j+1}^{\gamma} u_{jk} x_k \right).
\end{aligned}
$$

6.6.5 Symetrické pozitivně definitní pásové matice.

Soustavy s maticemi tohoto typu se nejčastěji vyskytují při řešení okrajových úloh pro parciální diferenciální rovnice eliptického typu (viz např. [10a]) např. metodou konečných prvků.

Pro takové soustavy je nejvýhodnější použít některé varianty LU-rozkladu. Obecně lze říci, že dáváme přednost eliminačním metodám tehdy, jestliže nám to kapacita paměti počítače dovolí.

Choleského algoritmus pro pásové matice lze získat úpravou algoritmu (6.6.3).

Obecně však nelze čekat, že matice $\mathbf{U}$, resp. $\mathbf{L}$ bude mít nulový prvek v téže pozici jako matice $\mathbf{A}$ (uvnitř pásu), tj. řídkost matice se obecně nezachovává.

Existuje řada publikací, kde čtenář nalezne vhodné algoritmy eliminační metody a jejich modifikací. Kromě [7], [7a], [14] jmenujme ještě: Agejev,M.I. a kol.: Biblioteka algoritmov, Moskva 1975, 1976, 1978.

Velké soustavy s maticemi výše uvedeného typu se v praxi často řeší iteračními metodami, nichž bude řeč v čl. 7. Při jejich použití však obvykle narazíme na problém pomalé konvergence metody.

6.7 Cvičení

Ve cvičeních 6.7.1 až 6.7.3 řešte soustavy lineárních rovnic s následujícími pozšířenými maticemi soustavy.

6.7.1

a) $\left(\begin{array}{rrr|r} 2 & -1 & 7 & 11 \\ 8 & -2 & 5 & 25 \\ 4 & 3 & -2 & 16 \end{array}\right)$; b) $\left(\begin{array}{rrrr|r} 1 & -1 & 1 & -2 & 8 \\ 2 & -1 & 2 & 1 & 5 \\ -1 & 1 & 2 & -4 & 10 \\ 1 & 2 & 4 & 1 & 5 \end{array}\right)$.

Užijte metod LU-rozkladu, GEM, GJEM, GEMPS, GEMPR, GEMPU. Počítejte v $\mathbf{R}$.

[a) $\mathbf{x}_t = (3,2,1)^T$; b) $\mathbf{x}_t = (1,-1,2,-2)^T$.]

6.7.2

$$\left(\begin{array}{rrr|r} 0,264\,1 & 0,173\,5 & 0,864\,2 & -0,752\,1 \\ 0,941\,1 & -0,017\,5 & 0,146\,3 & 0,631\,0 \\ -0,864\,1 & -0,424\,3 & 0,071\,1 & 0,250\,1 \end{array}\right).$$

Užijte opět metod podle pokynů ze cvič. 6.7.1. Určete $\mathbf{r} = \mathbf{b} - \mathbf{A}\mathbf{x}_\mathbf{c}$.

[$\mathbf{x}_\mathbf{c} = (0,731\,5; -2,189; -0,654\,4)^T$ v $M(10,4)$;
$\mathbf{x}_\mathbf{c} = (0,731\,519\,21; -2,188\,859\,6;\ -0,654\,393\,75)^T$ v $M(10,8)$ - s drobnými odchylkami na posledním desetinném místě v závislosti na užitém algoritmu.]

6.7.3

$$\left(\begin{array}{rrr|r} 0,763\,4 & 0,926\,5 & -1,753\,2 & 4,128\,7 \\ 2,152\,4 & 6,375\,4 & 1,817\,4 & 10,285\,3 \\ 0,723\,2 & -5,917\,6 & 2,314\,6 & -5,128\,7 \end{array}\right).$$

Užijte a) GEMPU; b) LU-rozkladu. Vypočítejte $\mathbf{r} = \mathbf{b} - \mathbf{A}\mathbf{x_c}$.

[a) $\mathbf{x_c} = (2,667\,546; 0,915\,041\,6; -0,709\,850\,3)^T$, $\mathbf{r} = (2 \cdot 10^{-7}; 0; 10^{-7})^T$ v $M(10,7)$; b) $\mathbf{x_c} = (2,667\,6; 0,915\,03; -0,070\,980)^T$, $\mathbf{r} = (4 \cdot 10^{-5}; -1 \cdot 10^{-4}; -2,3 \cdot 10^{-4})^T$ v $M(10,5)$.]

6.7.4

Dokažte, že pro libovolnou (reálnou) matici $\mathbf{A}$ a jsou matice $\mathbf{A}\mathbf{A}^T$, $\mathbf{A}^T\mathbf{A}$ symetrické a pro regulární matici $\mathbf{A}$ dokonce pozitivně definitní.

[Návod. Užijte vztahu $(\mathbf{A}\mathbf{B})^T = \mathbf{B}^T\mathbf{A}^T$ z čl. 5.1 a odst. 6.6.1.]

6.7.5

Je-li $\mathbf{A}$ symetrická matice a $\mathbf{Q}$ ortogonální matice, tj. taková, že $\mathbf{Q}^T\mathbf{Q} = \mathbf{I}$, pak $\mathbf{Q}^{-1}\mathbf{A}\mathbf{Q}$ je symetrická matice. Dokažte.

6.7.6

Proved'te LU-rozklad matice

$$\left(\begin{array}{ccc} 4 & 6 & 1 \\ 2 & 3 & 1 \\ 1 & 1 & 1 \end{array}\right).$$

Lze LU-rozklad provést?

6.7.7

Vypočtěte determinant matice soustavy ze cvič. 6.7.1 až 6.7.3. Jak se vypočte determinant, užijeme-li metody s výběrem hlavního prvku?

6.7.8

Řešte soustavu

$$\begin{aligned} x_1 + 2x_2 + 2x_3 &= 3, \\ 2x_1 - 7x_2 + 10x_3 &= 2, \\ 9x_1 - 26x_2 + 42x_3 &= t, \quad t \quad \text{je parametr.} \end{aligned}$$

[Pro $t \neq 11$ nemá soustava řešení; pro $t = 11 : \mathbf{x} = \frac{1}{11}(25, 4, 0)^T + \alpha(-34, 6, 11)^T$. Užijte GJEM.]

7 Iterační metody řešení soustav lineárních rovnic

V článku 6.6 jsme se seznámili s některými přímými metodami řešení velkých soustav. Zde si uvedeme metody, které dávají návod, jak řešit takové soustavy postupným přibližováním se k přesnému řešení. Existuje řada více či méně rafinovaných způsobů konstrukce takové posloupnosti aproximací, jejíž limitou je přesné řešení $\mathbf{x}_t$ dané soustavy rovnic.

Speciálních iteračních metod se také užívá ke zpřesnění řešení získaného eliminační metodou.

Ilustrujme si princip iterační metody nejdříve na jednoduchém příkladě.

7.1 Příklad.

Chceme iterační metodou najít řešení soustavy rovnic.

$$\begin{aligned} 11x_1 + 2x_2 + x_3 &= 15, \\ x_1 + 10x_2 + 2x_3 &= 16, \\ 2x_1 + 3x_2 - 8x_3 &= 1 \end{aligned} \quad \text{resp.} \quad \begin{pmatrix} 11 & 2 & 1 \\ 1 & 10 & 2 \\ 2 & 3 & -8 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 15 \\ 16 \\ 1 \end{pmatrix}. \tag{7.1.1}$$

Soustavu (7.1.1) musíme nejprve upravit na tvar vhodný pro provádění iterací [$\mathbf{x} = \mathbf{F}(\mathbf{x})$, viz odst. 1.4.6]. Provádí se to tak, že z každé rovnice vyčleníme jednu neznámou a ostatní členy převedeme na druhou stranu. Např.

$$\begin{aligned} x_1 &= \frac{1}{11}(15 - 2x_2 - x_3), \\ x_2 &= \frac{1}{10}(16 - x_1 - 2x_3), \\ x_3 &= \frac{1}{8}(-1 + 2x_1 + 3x_2), \end{aligned} \tag{7.1.2}$$

tj.

$$\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \underbrace{\begin{pmatrix} 0 & -\frac{2}{11} & -\frac{1}{11} \\ -\frac{1}{10} & 0 & -\frac{1}{5} \\ \frac{1}{4} & \frac{3}{8} & 0 \end{pmatrix}}_{\mathbf{H_J}} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} + \begin{pmatrix} \frac{15}{11} \\ \frac{8}{5} \\ -\frac{1}{8} \end{pmatrix}.$$

Jiná možnost:

$$\begin{aligned} x_1 &= 15 - 10x_1 - 2x_2 - x_3, \\ x_2 &= 16 - x_1 - 9x_2 - 2x_3, \\ x_3 &= -1 + 2x_1 + 3x_3 - 7x_3, \end{aligned} \tag{7.1.3}$$

tj.

$$\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \underbrace{\begin{pmatrix} -10 & -2 & -1 \\ -1 & -9 & -2 \\ 2 & 3 & -7 \end{pmatrix}}_{\mathbf{H_B}} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} + \begin{pmatrix} 15 \\ 16 \\ -1 \end{pmatrix}.$$

Takových možností je nekonečně mnoho, ale jenom některé vedou ke konvergentním posloupnostem aproximací.

Z rovnic (7.1.2) a (7.1.3) dostaneme rekurentní vztahy tak, že k neznámým na levé straně připíšeme index $k+1$ a k neznámým na pravé straně index k. Takže máme

$$\begin{aligned} x_1^{(k+1)} &= \frac{1}{11}(15 - 2x_2^{(k)} - x_3^{(k)}), \\ x_2^{(k+1)} &= \frac{1}{10}(16 - x_1^{(k)} - 2x_3^{(k)}), \qquad \text{tj} \quad \mathbf{x}^{(k+1)} = \mathbf{H_J}\mathbf{x}^{(k)} + \mathbf{g_J}, \\ x_3^{(k+1)} &= \frac{1}{8}(-1 + 2x_1^{(k)} + 3x_2^{(k)}), \end{aligned} \tag{7.1.4}$$

a nebo

$$\begin{aligned} x_1^{(k+1)} &= 15 - 10x_1^{(k)} - 2x_2^{(k)} - x_3^{(k)}, \\ x_2^{(k+1)} &= 16 - x_1^{(k)} - 9x_2^{(k)} - 2x_3^{(k)}, \qquad \text{tj} \quad \mathbf{x}^{(k+1)} = \mathbf{H_B}\mathbf{x}^{(k)} + \mathbf{g_B}. \\ x_3^{(k+1)} &= -1 + 2x_1^{(k)} + 3x_2^{(k)} - 7x_3^{(k)}, \end{aligned} \tag{7.1.5}$$

Zvolíme nyní $\mathbf{x}^{(0)} = (x_1^{(0)}, x_2^{(0)}, x_3^{(0)})^T$, např. $(0,0,0)^T$. Dosadíme tyto počáteční hodnoty do pravé strany rekurentních vztahů (7.1.4) a vypočteme

$$\mathbf{x}^{(1)} = (\frac{15}{11}, \frac{16}{10}, -\frac{1}{8})^T.$$

Pokračujeme-li, pak

$$\mathbf{x}^{(2)} = (1,084\,0; 1,488\,6; 0,815\,90)^T,$$
$$\mathbf{x}^{(3)} = (1,018\,8; 1,328\,4; 0,704\,22)^T$$
$$\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots$$

Provedeme-li několik iterací, zjistíme, že u dalších iterací se číslice na prvních m místech nemění; zdá se tedy oprávněná domněnka, že vektor

$$\mathbf{x} = (1,056; 1,364; 0,651)^T$$

bude aproximovat přesné řešení.

Počítáme-li analogicky podle rekurentních vztahů (7.1.5), dostáváme:

$$\mathbf{x}^{(1)} = (15, 16, -1)^T,$$
$$\mathbf{x}^{(2)} = (-166, -145, 84)^T,$$
$$\mathbf{x}^{(3)} = (1\,881, 1\,655, -1\,190)^T \text{ atd.}$$

Zde žádnou tendenci ke konvergenci nevidíme a je pravděpodobné, že posloupnost iterací diverguje.

Vzhledem k dalšímu výkladu je užitečné si všimnout, že soustavu rovnic (7.1.2) jsme maticově zapsali jako

$$\mathbf{x} = \mathbf{H_J x} + \mathbf{g_J} \tag{7.1.6}$$

a příslušná iterační formule pak měla tvar

$$\mathbf{x}^{(k+1)} = \mathbf{H_J x}^{(k)} + \mathbf{g_J}$$

Totéž můžeme udělat u soustavy (7.1.3).

7.2 Obecně o iteračních metodách.

Uvažujme soustavu lineárních rovnic s reálnou maticí soustavy

$$\mathbf{A x} = \mathbf{b} \tag{7.2.1}$$

a předpokládejme, že existuje jediné (přesné) řešení $\mathbf{x}_t = \mathbf{A}^{-1}\mathbf{b}$. Rovnici (7.2.1) přepíšeme na tvar vhodný k iteraci

$$\mathbf{x} = \mathbf{H x} + \mathbf{g} \tag{7.2.2}$$

tak, aby jejim řešením byl také vektor $\mathbf{x}_t = \mathbf{A}^{-1}\mathbf{b}$. [11]) Matici $\mathbf{H}$ nazýváme *iterační maticí.*

Sestrojíme posloupnost iterací $\mathbf{x}^{(k)} = (x_1^{(k)}, x_2^{(k)}, ..., x_n^{(k)})$ podle formule

$$\mathbf{x}^{(k+1)} = \mathbf{H}\mathbf{x}^{(k)} + \mathbf{g}, \quad k = 0, 1, 2, \ldots . \tag{7.2.3}$$

Jestliže existuje $\lim_{k\to\infty} \mathbf{x}^{(k)} = \mathbf{x}^*$, potom přechodem k limitě ve vztahu (7.2.3) dostaneme $\mathbf{x}^* = \mathbf{H}\mathbf{x}^* + \mathbf{g}$, a tedy $\mathbf{x}^*$ je řešením rovnice (7.2.2). Z našeho předpokladu pak vyplývá, že $\mathbf{x}^* = \mathbf{x}_t$ a iterace $\mathbf{x}^{(k)}$ je aproximací $\mathbf{x}_t$.

Předepsat iterační formuli nestačí. Musíme ještě připojit instrukci, jak výpočet zahájit - tj. volbu $\mathbf{x}^{(0)}$ - a jak ho ukončit. Ukončení výpočtu můžeme provést dvěma způsoby:

(i) stanovit kolik iterací máme počítat;

(ii) zastavovací podmínkou: zvolíme číslo $\delta > 0$ (dostatečně malé) a výpočet ukončíme, bude-li (v nějaké vektorové normě)

$$\left\|\mathbf{x}^{(k)} - \mathbf{x}^{(k-1)}\right\| < \delta. \tag{7.2.4}$$

Z metodických důvodů uděláme následující dohodu: Je-li $\mathbf{x}^{(k)}$ aproximace řešení určená tak, že vyhovuje podmínce (7.2.4), řekneme, že řešení $\mathbf{x}_t$ je touto aproximací určeno s přesností δ. Je-li aproximace $\mathbf{x}^{(k)}$ určena tak, že

$$\left\|\mathbf{x}_t - \mathbf{x}^{(k)}\right\| < \varepsilon,$$

řekneme, že $\mathbf{x}_t$ je touto aproximací určeno s chybou (s odhadem chyby) ε. V odstavci 7.5.3 si ukážeme, že ε lze určit pomocí δ.

7.2.1 Poznámka.

Aproximace $\mathbf{x}^{(k)}$ řešení $\mathbf{x}_t$ lze také vypočítávat podle obecnější iterační formule

$$\mathbf{x}^{(k+1)} = \mathbf{H}_k\mathbf{x}^{(k)} + \mathbf{g}_k, \quad k = 0, 1, 2, \ldots . \tag{7.2.5}$$

ve které se jak matice $\mathbf{H}_k$, tak vektor $\mathbf{g}_k$ mohou při každém kroku měnit. V takových případech hovoříme o *nestacionárním iteračním procesu* na rozdíl od procesu s maticí $\mathbf{H}$ nezávislou na k, který nazýváme *stacionárním iteračním procesem.* Podrobněji viz [6], [12].

[11]) Musí tedy řešení být splněna podmínka $\mathbf{A}^{-1}\mathbf{b} = \mathbf{H}\mathbf{A}^{-1}\mathbf{b} + \mathbf{g}$, tj. $(\mathbf{I} - \mathbf{H})\mathbf{A}^{-1}\mathbf{b} = \mathbf{g}$ (tzv. *podmínka konzistence.)*

7.3 Jacobiova metoda

7.3.1 Odvození iterační formule.

Zobecníme postup z příkl. 7.1. Napíšeme si i-tou rovnici soustavy $\mathbf{Ax} = \mathbf{b}$

$$a_{i1}x_1+a_{i2}x_2+...+a_{in}x_n = b_i, \quad i = 1, 2, ..., n. \tag{7.3.1}$$

Když $a_{ii} \neq 0$, pak dostáváme

$$x_i = \frac{1}{a_{ii}} \left(b_i - \sum_{\substack{j=1 \\ j \neq i}}^{n} a_{ij}x_j \right), \quad i = 1, 2, ..., n. \tag{7.3.2}$$

tedy z i-té rovnice jsme vypočítali i-tou neznámou. *Jacobiova iterační formule* má tvar ($k = 0, 1, 2, ...$):

$$x_i^{(k+1)} = \frac{1}{a_{ii}} \left(b_i - \sum_{j=1}^{n} a_{ij}x_j^{(k)} \right), \quad i = 1, 2, ..., n. \tag{7.3.3}$$

Maticově

$$\mathbf{x}^{(k+1)} = \mathbf{H_J}\mathbf{x}^{(k)}+\mathbf{g_J}, \quad k = 0, 1, 2, ... \tag{7.3.4}$$

kde

$$\mathbf{H_J} = \begin{pmatrix} 0 & -\frac{a_{12}}{a_{11}} & -\frac{a_{13}}{a_{11}} & \dots & -\frac{a_{1n}}{a_{11}} \\ -\frac{a_{21}}{a_{11}} & 0 & -\frac{a_{23}}{a_{11}} & \dots & -\frac{a_{2n}}{a_{11}} \\ \dots & \dots & \dots & \dots & \dots \\ -\frac{a_{n1}}{a_{11}} & -\frac{a_{n2}}{a_{11}} & -\frac{a_{n3}}{a_{11}} & \dots & 0 \end{pmatrix}, \qquad \mathbf{g_J} = \begin{pmatrix} \frac{b_1}{a_{11}} \\ \frac{b_2}{a_{11}} \\ \dots \\ \frac{b_n}{a_{11}} \end{pmatrix}.$$

Tab. 8

$\mathbf{x}^{(0)}$	$\mathbf{x}^{(1)}$	$\mathbf{x}^{(2)}$	$\mathbf{x}^{(3)}$	$\mathbf{x}^{(4)}$	$\mathbf{x}^{(5)}$	$\mathbf{x}^{(6)}$	$\mathbf{x}^{(\infty)} = \mathbf{x}_t$
0	0,25	0,375	0,437 5	0,468 75	0,484 38	0,492 19	0,5
0	0,5	0,625	0,687 5	0,718 75	0,734 38	0,742 19	0,75
0	0	0,125	0,187 5	0,218 75	0,234 38	0,242 19	0,25
0	0,25	0,375	0,437 5	0,468 75	0,484 38	0,492 19	0,5

7.3.2 Příklad

Řešíme soustavu $\mathbf{Ax} = \mathbf{b}$ Jacobiovu metodou, kde

$$\mathbf{A} = \begin{pmatrix} 4 & -1 & -1 & 0 \\ -1 & 4 & 0 & -1 \\ -1 & 0 & 4 & -1 \\ 0 & -1 & -1 & 4 \end{pmatrix}, \qquad \mathbf{b} = \begin{pmatrix} 1 \\ 2 \\ 0 \\ 1 \end{pmatrix}.$$

Výsledky zapíšeme do tabulky (viz tab. 8). Iterace konvergují, ale poněkud pomalu.

7.4 Gaussova-Seidelova metoda

7.4.1 Odvození iterační formule.

Použijme opět i-té rovnice (7.3.1) a formule (7.3.2), kterou si však nyní zapíšeme v rozepsané podobě

$$(7.4.1)\; x_i = \frac{1}{a_{ii}} \left(b_i - a_{i1}x_1 - a_{i2}x_2 - ... - a_{i,i-1}x_{i-1} - a_{i,i+1}x_{i+1} - ... - a_{in}x_{in} \right),$$

resp.

$$(7.4.2) \qquad x_i = \frac{1}{a_{ii}} \left(b_i - \sum_{j=1}^{i-1} a_{ij}x_j - \sum_{j=i+1}^{n} a_{ij}x_j \right), \quad i = 1, 2, ..., n.$$

Připojíme-li iterační indexy ($i = 0, 1, 2, ...$) následujícím způsobem

$$(7.4.3) \qquad x_i^{(k+1)} = \frac{1}{a_{ii}} \left(b_i - \sum_{j=1}^{i-1} a_{ij}x_j^{(k+1)} - \sum_{j=i+1}^{n} a_{ij}x_j^{(k)} \right), \quad i = 1, 2, ..., n,$$

dostáváme *Gaussovu-Seidelovu iterační formuli*. Rozdíl od Jacobiovy formule spočívá v tom, že iterací složek je k dalšímu výpočtu užito ihned, jakmile jsou vypočítány. Maticově budeme tuto formuli zapisovat ve tvaru

$$(7.4.4) \qquad \mathbf{x}^{(i+1)} = \mathbf{H_S}\mathbf{x}^{(k)} + \mathbf{g_S}, \quad k = 0, 1, 2, ...$$

Konkrétní podoba matice $\mathbf{H_S}$ a vektoru $\mathbf{g_S}$ se dá také stanovit pomocí prvků a_{ij} matice $\mathbf{A}$ (viz odst. 7.5.5).

7.4.2 Příklad.

Řešme nyní soustavu z příkl. 7.3.2 Gaussovou-Seidelovou metodou. Počítáme tedy iteračních formulí

$$
\begin{aligned}
x_1^{(k+1)} &= \frac{1}{4}(1 + x_2^{(k)} + x_3^{(k)}),\\
x_2^{(k+1)} &= \frac{1}{4}(2 + x_1^{(k+1)} + x_4^{(k)}),\\
x_3^{(k+1)} &= \frac{1}{4}(x_1^{(k+1)} + x_4^{(k)}),\\
x_4^{(k+1)} &= \frac{1}{4}(1 + x_2^{(k+1)} + x_3^{(k+1.}),
\end{aligned}
$$

Výsledky zapíšeme opět do tabulky (tab. 9). Iterace opět konvergují, ale ve srovnání s Jacobiovou metodou rychleji, tzn. zde $\mathbf{x}^{(5)}$ je lepší aproximací řešení než tatáž iterace u Jacobiovy metody. Později posoudíme, zda tento jev má obecný charakter, nebo se to týká některých případů. Navíc je Gaussova-Seidelova metoda z hlediska uložení neznámých v paměti jednodušší a úspornější než Jacobiova metoda.

Tab. 9

$\mathbf{x}^{(0)}$	$\mathbf{x}^{(1)}$	$\mathbf{x}^{(2)}$	$\mathbf{x}^{(3)}$	$\mathbf{x}^{(4)}$	$\mathbf{x}^{(5)}$	...	$\mathbf{x}^{(\infty)} = \mathbf{x}_t$
0	0,25	0,406 25	0,476 56	0,494 14	0,498 54		0,5
0	0,562 5	0,703 12	0,738 28	0,747 07	0,749 27		0,75
0	0,062 5	0,203 12	0,238 28	0,247 07	0,249 27		0,25
0	0,406 25	0,476 56	0,494 14	0,498 54	0,499 63		0,5

7.4.3 Algoritmus.

Výpočet podle iterační formule (7.4.3) zapíšeme schematicky:

(7.4.5)

$$
\begin{aligned}
&\text{Vstup}: \quad n,\ \mathbf{A},\ \mathbf{b},\ \mathbf{x}^{(0)} = (x_1^{(0)}, x_2^{(0)}, ..., x_n^{(0)})^T,\ \delta,\ (m).\\
&\qquad \text{Pro} \quad k = 1, 2, 3, ..., (m).\\
&\qquad\qquad \text{Pro} \quad i = 1, 2, ..., n:\\
&\qquad\qquad\qquad x_i^{(k)} = \frac{1}{a_{ii}} \left(b_i - \sum_{j=1}^{i-1} a_{ij} x_j^{(k)} - \sum_{j=i+1}^{n} a_{ij} x_j^{(k-1)} \right).\\
&\text{Výstup}: \quad \mathbf{x}^m = (x_1^m, x_2^m, ..., x_n^m)^T.
\end{aligned}
$$

Výpočet ukončujeme zastavovací podmínkou (7.2.4). V každém kroku vnějšího cyklu kontrolujeme, zda je splněna. Počáteční aproximace volíme nejčastěji $\mathbf{x}^{(0)} = \mathbf{b}$. Ukončení výpočtu parametrem m přichází v úvahu při velmi pomalé konvergenci.

7.5 Konvergence iteračního procesu.

7.5.1 Nutná a postačující podmínka konvergence.

Vratíme se k odst. 7.2 a *vektor chyby* k-té iterace označíme

$$\mathbf{e}^{(k)} = \mathbf{x}^{(k)} - \mathbf{x}_t. \tag{7.5.1}$$

Protože

$$\mathbf{x}_t = \mathbf{H}\mathbf{x}_t + \mathbf{g},$$
$$\mathbf{x}^{(k)} = \mathbf{H}\mathbf{x}^{(k-1)} + \mathbf{g},$$

pak odečtením těchto vztahů dostaneme

$$\mathbf{e}^{(k)} = \mathbf{H}\mathbf{e}^{(k-1)} = \mathbf{H}^2\mathbf{e}^{(k-2)} = \ldots = \mathbf{H}^k\mathbf{e}^{(0)}. \tag{7.5.2}$$

Protože $\lim_{k\to\infty} \mathbf{x}^{(k)} = \mathbf{x}_t$, právě když $\lim_{k\to\infty} \mathbf{e}^{(k)} = \mathbf{0}$ (= nulový vektor), musíme vyšetřit konvergenci mocnin $\mathbf{H}^{(k)}$ matice $\mathbf{H}$.

Konvergenční věta: Iterace $\mathbf{x}^{(k)} = \mathbf{H}\mathbf{x}^{(k-1)} + \mathbf{g}$ *konvergují pro libovolné* $\mathbf{x}^{(0)}$, *tj.* $\lim_{k\to\infty} \mathbf{H}^{(k)}\mathbf{e}^{(0)} = \mathbf{0}$, *právě když*

$$\rho(\mathbf{H}) = \max_i |\lambda_i(\mathbf{H})| < 1, \tag{7.5.3}$$

kde číslo $\rho(\mathbf{H})$ *se nazývá spektrální poloměr matice* $\mathbf{H}$ *a* $\lambda_i(\mathbf{H})$ *jsou vlastní čísla matice* $\mathbf{H}$

Uvedeme si hlavní myšlenku důkazu konvergenční věty. Jsou-li $\lambda_1, \lambda_2, ..., \lambda_n$ vlastní čísla matice $\mathbf{H}$ (včetně násobných vlastních čísel), existuje regulární matice $\mathbf{T}$ taková, že matici $\mathbf{H}$ lze vyjádřit ve tvaru

$$\mathbf{H} = \mathbf{T}\mathbf{J}\mathbf{T}^{-1},$$

kde $\mathbf{J}$ je tzv. *Jordanova matice* (viz [6], [9]), což je blokově diagonální matice s bloky

$$\mathbf{J}_i = \begin{pmatrix} \lambda_i & 1 & 0 & \ldots & 0 \\ & \lambda_i & 1 & \ddots & \vdots \\ & & \ddots & \ddots & 0 \\ & 0 & & \ddots & 1 \\ & & & & \lambda_i \end{pmatrix}.$$

Řád matic $\mathbf{J}_i$ je určen tzv. elementárními děliteli matice $\mathbf{A}$.[12]) Potom

$$\mathbf{H}^k = \mathbf{T}\mathbf{J}^k\mathbf{T}^{-1},$$

Platí-li (7.5.3), potom $\mathbf{J}^k$ konverguje k nulové matici (když $k \to \infty$), a tedy $\mathbf{H}^k\mathbf{e}^{(0)} \to \mathbf{0}$ pro libovolný vektor $\mathbf{e}^{(0)}$. Naopak, je-li $\lim_{k\to\infty} \mathbf{H}^k \to \mathbf{0}$, potom musí být $\lim_{k\to\infty} \mathbf{J}^k = \mathbf{0}$, a odtud plyne $|\lambda_i| < 1$ pro všechna i.

Protože podle vztahu (7.5.2) je $\mathbf{e}^{(k)}$ lineární funkcí $\mathbf{e}^{(k-1)}$, říkáme často, že *konvergence* uvažovaného procesu *je lineární.*

7.5.2 Postačující podmínka konvergence.

Je-li splněna podmínka

$$\|\mathbf{H}\| \leqq q < 1, \tag{7.5.4}$$

Potom posloupnost $\left\{\mathbf{x}^{(k)}\right\}_{k=0}^{\infty}$, *určená formulí* $\mathbf{x}^{(k)} = \mathbf{H}\mathbf{x}^{(k-1)} + \mathbf{g}$, *konverguje při libovolné volbě vektoru* $\mathbf{x}^{(0)}$ *a je*

$$\lim_{k\to\infty} \mathbf{x}^{(k)} = (\mathbf{I} - \mathbf{H})^{-1}\mathbf{g} = \mathbf{x}_t.$$

Z iterační formule postupným dosazováním dostaneme

$$\mathbf{x}^{(k)} = \mathbf{H}^k\mathbf{x}^{(0)} + (\mathbf{I} + \mathbf{H} + \mathbf{H}^2 + \mathbf{H}^3 + ... + \mathbf{H}^{k-1})\mathbf{g} = \mathbf{H}^k\mathbf{x}^{(0)} + \mathbf{S}_k\mathbf{g},$$

kde jsme označili

$$\mathbf{S}_k = \mathbf{I} + \mathbf{H} + \mathbf{H}^2 + ... + \mathbf{H}^{k-1}.$$

Protože

$$\mathbf{H}\mathbf{S}_k = \mathbf{H} + \mathbf{H}^2 + \mathbf{H}^3 + ... + \mathbf{H}^k,$$

potom

$$(\mathbf{I} - \mathbf{H})\mathbf{S}_k = \mathbf{I} - \mathbf{H}^k.$$

Předpoklad (7.5.4) zaručuje, že matice $\mathbf{H}^k$ konvergují k nulové matici (nebot' $\left\|\mathbf{H^k}\right\| \leqq \|\mathbf{H}\|^k$ a $\lim_{k\to\infty} \|\mathbf{H}\|^k = 0$). Pak ovšem

$$\lim_{k\to\infty} (\mathbf{I} - \mathbf{H})\mathbf{S}_k = \lim_{k\to\infty} (\mathbf{I} - \mathbf{H}^k) = \mathbf{I},$$

a tedy

$$\mathbf{S} = \lim_{k\to\infty} \mathbf{S}_k = (\mathbf{I} - \mathbf{H})^{-1}.$$

Proto

$$\lim_{k\to\infty} \mathbf{x}^k = \lim_{k\to\infty} [\mathbf{H}^k\mathbf{x}^{(0)} + \mathbf{S}_k\mathbf{g}] = (\mathbf{I} - \mathbf{H})^{-1}\mathbf{g}.$$

Existuje tedy vektor, který je limitou posloupnosti $\left\{\mathbf{x}^{(k)}\right\}_{k=0}^{\infty}$.

[12]) Pro některé matice (např. symetrické) je $\mathbf{J} = \mathrm{diag}(\lambda_1, \lambda_2, ..., \lambda_n)$.

7.5.3 Odhad chyby.

Předpokládáme, že platí (7.5.4); pro odhad chyby k-té iterace $\mathbf{x}^{(k)}$ užijeme rovnosti

$$\mathbf{x}^{(k)} - \mathbf{x}_t = \mathbf{H}(\mathbf{x}^{(k-1)} - \mathbf{x}_t) = \mathbf{H}(\mathbf{x}^{(k)} - \mathbf{x}_t) - \mathbf{H}(\mathbf{x}^{(k)} - \mathbf{x}^{(k-1)}).$$

Odtud dostáváme

$$\left\|\mathbf{x}^{(k)} - \mathbf{x}_t\right\| \leqq q \left\|\mathbf{x}^{(k)} - \mathbf{x}_t\right\| + q \left\|\mathbf{x}^{(k)} - \mathbf{x}^{(k-1)}\right\|.$$

tj.

$$(1-q) \left\|\mathbf{x}^{(k)} - \mathbf{x}_t\right\| \leqq q \left\|\mathbf{x}^{(k)} - \mathbf{x}^{(k-1)}\right\|,$$

a tedy pro $1 - q > 0$ bude

$$\left\|\mathbf{x}^{(k)} - \mathbf{x}_t\right\| \leqq \frac{q}{1-q} \left\|\mathbf{x}^{(k)} - \mathbf{x}^{(k-1)}\right\|. \tag{7.5.5}$$

Jestliže $\left\|\mathbf{x}^{(k)} - \mathbf{x}^{(k-1)}\right\| < \delta$, potom

$$\left\|\mathbf{x}^{(k)} - \mathbf{x}_t\right\| \leqq \frac{q}{1-q}\delta = \varepsilon.$$

7.5.4 Kritérium konvergence.

Podmínky (7.5.3), (7.5.4) se v praxi těžko ověřují. Proto je užitečné formulovat postačující podmínky konvergence přímo vzhledem k matici $\mathbf{A}$.

Uvažujme třídy reálných matic $\mathbf{A} = (a_{ij})$ s vlastnostmi:

(P1): Ostrá diagonální dominantnost (převaha), tj. platí

$$|a_{ii}| > \sum_{\substack{k=1 \\ k \neq i}}^{n} |a_{ik}| \quad i = 1, 2, ..., n. \tag{7.5.6}$$

(P2): Symetričnost a pozitivní definitnost (odst. 6.6.1).

(P3): a) Neostrá diagonální dominantnost [vztah (6.6.3)] a ostrá alespoň pro jedno i.

b) Nerozpadlost (irreducibilita)[13]).

[13]) Matice $\mathbf{A}$ soustavy $\mathbf{Ax} = \mathbf{b}$ je *rozpadlá* (*reducibilní*), jestliže n_1 ($n_1 < n$) složek vektoru $\mathbf{x}$ je jednoznačně určeno stejným počtem složek vektoru $\mathbf{b}$, tj. těchto n_1 tovnic soustavy lze řešit nezávislé na ostatních rovnicích.

Soustavy, jejichž matice soustavy mají vlastnosti (P2) nebo (P3), se nejčastěji vyskytují při numerickém řešení okrajových úloh pro parciální diferenciální rovnice (viz [10a]).

Dají se dokázat (viz např. [5], [6]) následující kritéria konvergence:

(1) *Má-li matice* $\mathbf{A}$ *vlastnost* (P1), *pak Jacobiova i Gaussova-Seidelova metoda konvergují pro libovolný počáteční vektor* $\mathbf{x}^{(0)}$.
(2) *Má-li matice* $\mathbf{A}$ *vlastnost* (P2), *pak Gaussova-Seidelova metoda konverguje pro libovolný počáteční vektor* $\mathbf{x}^{(0)}$.
(3) *Má-li matice* $\mathbf{A}$ *vlastnost* (P3) *a matice* $\mathbf{H_J}$ (*viz odst.* 7.3.1) *má nezáporné prvky, potom platí právě jedna z následujících podmínek*:

$$(a) \qquad \rho(\mathbf{H_J}) = \rho(\mathbf{H_S}) - 0,$$

$$(b) \qquad 0 < \rho(\mathbf{H_S}) < \rho(\mathbf{H_J}) < 1,$$

$$(c) \qquad \rho(\mathbf{H_J}) = \rho(\mathbf{H_S}) = 1,$$

$$(d) \qquad 1 < \rho(\mathbf{H_J}) < \rho(\mathbf{H_S}).$$

Například pro soustavu s maticí

$$\mathbf{A} = \begin{pmatrix} 2 & -1 & 0 & 0 \\ -1 & -2 & -1 & 0 \\ 0 & -1 & 2 & -1 \\ 0 & 0 & -1 & 2 \end{pmatrix}$$

jsou splněny vlastnosti (P2) a (P3). Podle kritéria (2) Gaussova-Seidelova metoda bude konvergovat, a protože matice

$$\mathbf{H_J} = \begin{pmatrix} 0 & \frac{1}{2} & 0 & 0 \\ \frac{1}{2} & 0 & \frac{1}{2} & 0 \\ 0 & \frac{1}{2} & 0 & \frac{1}{2} \\ 0 & 0 & \frac{1}{2} & 0 \end{pmatrix}$$

má nezáporné prvky, pak podle kritéria (3b) bude konvergovat i Jacobiova metoda, ovšem pomaleji.

V příkladu 7.1 splňuje matice soustavy kritérium (1), v příkl. 7.3.2, 7.4.2 jsou splněna všechna tři kritéria, a tedy podle (3b) bude opět Jacobiova metoda konvergovat pomaleji.

Poznamenejme na závěr odstavce, že pro některé obecné matice $\mathbf{A}$ může Jacobiova metoda konvergovat, avšak Gaussova-Seidelova metoda konvergovat nemusí.

7.5.5 Konkrétní tvary iteračních matic.

Stanovíme tvar iterační matice $\mathbf{H_J}$ (Jacobiovy) a $\mathbf{H_S}$ (Gaussovy-Seidelovy). Regulární matici $\mathbf{A}$ napíšeme ve tvaru

$$\mathbf{A} = \mathbf{M} + \mathbf{D} + \mathbf{N},$$

kde

$$\mathbf{M} = \begin{pmatrix} 0 & 0 & \dots & \dots & 0 \\ a_{21} & 0 & & & \vdots \\ a_{31} & a_{32} & 0 & & \vdots \\ \vdots & & \ddots & \ddots & \vdots \\ a_{n1} & \dots & \dots & a_{n,n-1} & 0 \end{pmatrix},$$

$$\mathbf{N} = \begin{pmatrix} 0 & a_{12} & a_{13} & \dots & a_{1n} \\ 0 & 0 & a_{23} & \dots & \vdots \\ \vdots & & \ddots & \ddots & \vdots \\ \vdots & & & \ddots & a_{n-1,n} \\ 0 & \dots & \dots & \dots & 0 \end{pmatrix}.$$

Nyní soustavu $(\mathbf{M} + \mathbf{D} + \mathbf{N})\mathbf{x} = \mathbf{b}$ napíšeme bud' ve tvaru

$$\mathbf{D}\mathbf{x} = -(\mathbf{M}+\mathbf{N})\mathbf{x}+\mathbf{b}, \tag{7.5.7}$$

nebo ve tvaru

$$(\mathbf{M}+\mathbf{D})\mathbf{x} = -\mathbf{N}\mathbf{x}+\mathbf{b}. \tag{7.5.8}$$

Z rovnice (7.5.7) dostaneme Jacobiovu iterační formuli

$$\mathbf{D}\mathbf{x}^{(k+1)} = -(\mathbf{M} + \mathbf{N})\mathbf{x}^{(k)} + \mathbf{b}$$

a z rovnice (7.5.8) Gaussovu-Seidelovu formuli

$$(\mathbf{M} + \mathbf{D})\mathbf{x}^{(k+1)} = -\mathbf{N}\mathbf{x}^{(k)} + \mathbf{b}.$$

Proto

$$\mathbf{H_J} = -\mathbf{D}^{-1}(\mathbf{M} + \mathbf{N}), \qquad \mathbf{g_J} = \mathbf{D}^{-1}\mathbf{b},$$
$$\mathbf{H_S} = (\mathbf{M} + \mathbf{D})^{-1}\mathbf{N}, \qquad \mathbf{g_S} = (\mathbf{M} + \mathbf{D})^{-1}\mathbf{b}.$$

7.6 Zrychlení iteračního procesu

7.6.1 Rychlost konvergence.

Podmínka (7.5.3), resp. (7.5.4) zaručuje konvergenci iteračního procesu, avšak nezískáme z ní žádnou informaci o tom, kolik iterací musíme udělat, abychom dosáhli zadané přesnosti. Lehce vypočteme, že k realizaci jedné iterace potřebujeme n^2 nasobení a n dělení (u řídkých matic je toto číslo menší).

Protože ze vztahu (7.5.2) vyplývá

$$\left\|\mathbf{e}^{(k)}\right\| \leqq \|\mathbf{H}\| \left\|\mathbf{e}^{(k-1)}\right\| \leqq \|\mathbf{H}\|^k \left\|\mathbf{e}^{(0)}\right\|.$$

je vidět, že pro $\|\mathbf{H}\|$ blizké jedničce budeme potřebovat hodně iterací (velké k), aby číslo $\left\|\mathbf{e}^{(k)}\right\|$ bylo malé.

Je-li např. $\left\|\mathbf{e}^{(k-1)}\right\| = 10^{-2}$ a $\left\|\mathbf{e}^{(k)}\right\| = 10^{-4}$, tj. jednou iterací jsme získali dvě platná desetinná místa, pak se jeví účelným posuzovat rychlost konvergence (pro větší k) číslem

$$R = -\log(\left\|\mathbf{e}^{(k)}\right\| / \left\|\mathbf{e}^{(k-1)}\right\|), \tag{7.6.1}$$

které určuje počet platných desetinných míst získaných v každém iteračním kroku (číslo $1/R$ určuje pak počet iterací potřebných k zisku jednoho desetinného místa).

U celé řady praktických úloh, především u zmíněných okrajových úloh pro parciální diferenciální rovnice, je číslo R velmi malé. Proto vzniká nutnost zrychlení iteračního procesu.

7.6.2 Aitkenova extrapolační formule.

Jestliže posloupnost iterací $\left\{\mathbf{x}^{(k)}\right\}$ určená formulí $\mathbf{x}^{(k+1)} = \mathbf{H}\mathbf{x}^{(k)} + \mathbf{g}$ konverguje k vektoru $\mathbf{x}$, potom posloupnost chyb $\mathbf{e}^{(k)} = \mathbf{x}^{(k)} - \mathbf{x}$ má vzhledem k (7.5.2) charakter geometrické posloupnosti (při větších k), tj. pro složky platé

$$\frac{x_i^{(k+1)} - x_i}{x_i^{(k)} - x_i} \approx \frac{x_i^{(k)} - x_i}{x_i^{(k-1)} - x_i}, \quad i = 1, 2, ..., n.$$

Jednoduchou úpravou odtud dostaneme

$$x_i \approx x_i^{(k+1)} - \frac{(x_i^{(k+1)} - x_i^{(k)})^2}{x_i^{(k+1)} - 2x_i^{(k)} + x_i^{(k-1)}}.$$

Dá se ukázat, že vektor $\mathbf{z}^{(k+1)}$, jehož složky jsou určeny pravou stranou předchozího vztahu, bude lepší aproximací vektoru $\mathbf{x}$ než vektor $\mathbf{x}^{(k+1)}$.

V případě, že posloupnost iterací $\mathbf{x}^{(k)}$ konverguje k $\mathbf{x}$ pomalu, využijeme tohoto faktu k urychlování iteračního procesu následujícím způsobem: Stanovíme

$$\mathbf{z}_i^{(k+1)} = \mathbf{x}_i^{(k+1)} - \frac{(x_i^{(k+1)} - x_i^{(k)})^2}{x_i^{(k+1)} - 2x_i^{(k)} + x_i^{(k-1)}}, \quad i = 1, 2, ..., n. \tag{7.6.2}$$

a podle základní formule vypočteme $\mathbf{z}^{(k+2)} = \mathbf{H}\mathbf{z}^{(k+1)} + \mathbf{g}$, $\mathbf{z}^{(k+3)} = \mathbf{H}\mathbf{z}^{(k+2)} + \mathbf{g}$ a opět zlepšíme aproximaci $\mathbf{z}^{(k+3)}$ pomocí vzorce (7.6.2). V tomto procesu pokračujeme až do splnění zastovovací podmínky.

7.6.3 Relaxační metoda.

Iterační formuli (7.4.3) Gaussovy-Seidelovy metody lze také psát ve tvaru

$$x_i^{(k+1)} = x_i^{(k)} + r_i^{(k)}, \quad i = 1, 2, ..., n, \tag{7.6.3}$$

kde

$$r_i^{(k)} \frac{1}{a_{ii}} \left(b_i - \sum_{j=1}^{i-1} a_{ij} x_j^{(k+1)} - \sum_{j=i}^{n} a_{ij} x_j^{(k)} \right)$$

je tzv. *i-té reziduum.*

Budeme nyní uvažovat iterační formuli [pro $\omega = 1$ totožnou s formulí (7.6.3)]:

$$x_i^{(k+1)} = x_i^{(k)} + \omega r_i^{(k)}, \tag{7.6.4}$$

resp.

$$x_i^{(k+1)} = \frac{\omega}{a_{ii}} \left(b_i - \sum_{j=1}^{i-1} a_{ij} x_j^{(k+1)} - \sum_{j=i+1}^{n} a_{ij} x_j^{(k)} \right) + (1-\omega) x_i^{(k)}, \tag{7.6.4'}$$

kde ω je tzv. *relaxační parametr.* Tento relaxační parametr se snažíme vybrat tak, aby konvergence iteračního procesu byla co nejrychlejší.

Iterační formulí (7.6.4) lze zapsat maticově

$$\mathbf{x}^{(k+1)} = \mathbf{H}_\omega \mathbf{x}^{(k)} + \mathbf{g}_\omega \tag{7.6.5}$$

kde (užíváme označení z odst. 7.5.5)

$$\mathbf{H}_\omega = (\mathbf{D} + \omega\mathbf{M})^{-1}[(1-\omega)\mathbf{D} - \omega\mathbf{N}],$$
$$\mathbf{g}_\omega = (\mathbf{D} + \omega\mathbf{M})^{-1}\omega\mathbf{b}.$$

Protože $\mathbf{D}(\mathbf{I}+\omega\mathbf{D}^{-1}\mathbf{M}) = \mathbf{D}+\omega\mathbf{M}$, $[(1-\omega)\mathbf{I}-\omega\mathbf{N}\mathbf{D}^{-1}]\mathbf{D} = (1-\omega)\mathbf{D}-\omega\mathbf{N}$, potom

$$\mathbf{H}_\omega = \mathbf{D}^{-1}(\mathbf{I}+\omega\mathbf{D}^{-1}\mathbf{M})^{-1}[(1-\omega)\mathbf{I}-\omega\mathbf{N}\mathbf{D}^{-1}]\mathbf{D},$$

a tedy ($\det \mathbf{D}^{-1}\mathbf{B}\mathbf{D} = \det \mathbf{B}$ - viz odst. 5.1.1).

$$\det \mathbf{H}_\omega = \ textdet\,(\mathbf{I}+\omega\mathbf{D}^{-1}\mathbf{M})^{-1}\det\,[(1-\omega)\mathbf{I}-\omega\mathbf{N}\mathbf{D}^{-1}] = (1-\omega)^n,$$

nebot' $(\mathbf{I}+\omega\mathbf{D}^{-1}\mathbf{M})^{-1}$ je dolní trojúhelníková matice s jedničkami na diagonále a $(1-\omega)\mathbf{I}-\omega\mathbf{N}\mathbf{D}^{-1}$ je horní trojúhelníková matice s diagonálními prvky $1-\omega$.

Zjistíme, pro jaké ω je splněna podmínka konvergence procesu (7.6.5). Napíšeme matici $\mathbf{H}_\omega$ ve tvaru (viz odst. 7.5.1)

$$\mathbf{H}_\omega = \mathbf{T}\mathbf{J}\mathbf{T}^{-1},$$

kde $\mathbf{J}$ je Jordanova matice. Protože $\det \mathbf{H}_\omega = \det \mathbf{J} = \lambda_1\lambda_2...\lambda_n$ (λ_i jsou vlastní čísla matice $\mathbf{H}_\omega$; každé číslo počítáme tolikrát, kolik je jeho násobnost), plyne odtud

$$\max_i |\lambda_i| \geqq |1-\omega|.$$

Má-li iterační proces (7.6.5) konvergovat, nemůže být $|1-\omega| > 1$. Proto podmínka

$$0 < \omega < 2 \tag{7.6.6}$$

je nutnou podmínkou konvergence [vyplývá ze (7.5.3)]. Dá se dokázat (viz [16]), že pro symetrické pozitivně definitní matice s kladnými diagonálními prvky je (7.6.6) i podmínkou postačující [dokáže se splnění podmínky (7.5.3)].

Metodu, kterou jsme právě popsali, nazýváme často *superrelaxační metodou*. Je to ovšem pouze jedna z veliké třídy metod, jejichž cílem je urychlit konvergenci.

Podrobnější informace o relaxačních metodách najde čtenář např. v [6], [12], [5], [16]. Pro některé třídy pásových matic se dá stanovit optimální hodnota parametru ω (který se nazývá *superrelaxační faktor*), při níž proces (7.6.5) konverguje nejrychleji (viz např. [4]).

7.6.4 Poznámka k zaokrouholvacím chybám.

Protože vždy počítáme v nějakém systému $M(q,t)$, mohou nám zaokrouhlovací chyby narušit konvergenci tehdy, jestliže $\rho(\mathbf{H})$ je velmi blizké jedničce. Potom totiž teoretické zlepšení iterace v k-tém kroku může být tak malé, že jej celkový vliv zaokrouhlovacích chyb může zcela kompenzovat a proces fakticky konvergovat nebude. Viz také odst. 9.2.

7.7 Cvičení

7.7.1

Řešte soustavu

$$\begin{pmatrix} 5,21 & 1,52 & -2,37 \\ 1,72 & -2,97 & 0,21 \\ 2,01 & 0,92 & 3,89 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 1,68 \\ 6,21 \\ 7,78 \end{pmatrix}$$

a) Jacobiovou metodou; b) Gaussovou-Seidelovou metodou. Počítejte v systému $M(10,5)$ a výpočet ukončete podmínkou $\left\|\mathbf{x}^{(k)} - \mathbf{x}^{(k-1)}\right\| \leqq 10^{-4}$. Volte $\mathbf{x}^{(0)} = (1,-1,1)^T$. Užijte také Aitkenovy formule k zrychlení iteračního procesu.

a) $\mathbf{x}^{(24)} = (1,379\,6; -1,181\,1; 1,566\,4)^T$; b) k dosažení stejného výsledku stačí 16 iterací. Aitkenovy formule užijte po splnění podmínky $\left\|\mathbf{x}^{(k)} - \mathbf{x}^{(k-1)}\right\|_{\mathbf{R}} \leqq 10^{-2}$.

7.7.2

Řešte soustavu

$$\begin{pmatrix} 4 & 1 & 1 \\ 1 & 6 & 2 \\ 1 & 2 & 4 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} -1 \\ 0 \\ 1 \end{pmatrix}$$

a) Gaussovou-Seidelovou metodou; b) relaxační metodou pro $\omega = 1,5$.

7.7.3

Výpočtem ukažte, že pro soustavu

$$\begin{pmatrix} 1 & 2 & -2 \\ 1 & 1 & 1 \\ 2 & 2 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 1 \\ 3 \\ 5 \end{pmatrix}$$

Jacobiova metoda konverguje, kdežto Gaussova-Seidellova diverguje.
$[\mathbf{x}_t = (1,1,1)^T.]$

7.7.4

Výpočtem ukažte, že pro soustavu

$$\begin{pmatrix} 5 & 3 & 4 \\ 3 & 6 & 4 \\ 4 & 4 & 5 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 12 \\ 13 \\ 13 \end{pmatrix}$$

Jacobiova metoda diverguje, kdežto Gaussova-Seidellova konverguje.
$[\mathbf{x}_t = (1, 1, 1)^T.]$

7.7.5

Řešte soustavu

$$\begin{pmatrix} 4 & -1 & 0 & 0 \\ -1 & 4 & -1 & 0 \\ 0 & -1 & 4 & -1 \\ 0 & 0 & -1 & 4 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} = \begin{pmatrix} 2 \\ 5 \\ 5 \\ 2 \end{pmatrix}$$

Jacobiovou a Gaussovou-Seidelovou metodou. Posud'te rychlost konvergence obou metod.
$[\mathbf{x}_t = (1, 2, 2, 1)^T.]$

8 Inverze matic

Již dříve jsme se zmínili (odst. 6.3.2) o metorě výpočtu inverzní matice (GJEM). K určení matice $\mathbf{A}^{-1}$ lze ovšem užít také algoritmu GEM: Řešíme soustavy $\mathbf{A}\mathbf{s}_j = \mathbf{e}_j$, $j = 1, 2, ..., n$, kde $\mathbf{e}_j$ je j-tý sloupec jednotkové matice $\mathbf{I}$. Potom $\mathbf{s}_j$ jsou sloupce matice $\mathbf{A}^{-1}$.

8.1 Inverze LU-rozkladem.

Předpokládáme, že máme proveden LU-rozklad regulární matice $\mathbf{A}$, tj. $\mathbf{A} = \mathbf{L}\mathbf{U}$. Vzhledem k tomu, že $\mathbf{A}^{-1} = \mathbf{U}^{-1}\mathbf{L}^{-1}$, stačí stanovit $\mathbf{U}^{-1}$ a $\mathbf{L}^{-1}$.

Budeme proto hledat takové matice $\mathbf{Y}$ a $\mathbf{Z}$, že $\mathbf{LY} = \mathbf{I}$ a $\mathbf{UZ} = \mathbf{I}$. Dá se ukázat, že matice $\mathbf{Y}$, $\mathbf{Z}$ budou mít tvar (prověřte!)

$$\mathbf{Y} = \begin{pmatrix} y_{11} & & & \\ y_{21} & y_{22} & & 0 \\ y_{31} & y_{32} & y_{33} & \\ \cdots & \cdots & \cdots & \cdots \\ y_{n1} & y_{n2} & \cdots & y_{nn} \end{pmatrix},$$

$$\mathbf{Z} = \begin{pmatrix} z_{11} & z_{12} & z_{13} & \cdots & z_{1n} \\ & z_{22} & z_{23} & \cdots & z_{2n} \\ & & z_{33} & \cdots & z_{3n} \\ & 0 & & \cdots & \cdots \\ & & & & z_{nn} \end{pmatrix}.$$

Z definice součinu matic vyplývá, že

$$1 = \mathbf{r}_i(\mathbf{L})\mathbf{s}_i(\mathbf{Y}) = 1 \cdot y_i,$$

$$0 = \mathbf{r}_i(\mathbf{L})\mathbf{s}_j(\mathbf{Y}) = l_{i,i}y_j + l_{i,j+1}y_{j+1,j} + ... + l_{i,i-1}y_{i-1,j} + y_{ij}, \quad j < i.$$

Tedy

$$y_{i,j} = \delta_{i,j} - \sum_{k=j}^{i-1} l_{ik}y_{kj}, \quad i = j, j+1, ..., n; \tag{8.1.1}$$

$$\delta_{ij} = \begin{cases} 1, & i = j, \\ 0, & i \neq j. \end{cases}$$

Algoritmus pro výpočet $\mathbf{Y} = \mathbf{L}^{-1}$ má tvar:

$$\begin{aligned} &\text{Vstup}: \quad n, \mathbf{L} = (l_{ij}). \\ &\qquad \text{Pro} \quad j = 1, 2, ..., n: \\ &\qquad\qquad \text{Pro} \quad i = j, j+1, ..., n: \\ &\qquad\qquad\qquad y_{ij} = \delta_{ij} - \sum_{k=j}^{i-1} l_{ik}y_{kj}. \\ &\text{Výstup}: \quad \mathbf{Y} = (y_{ij}) = \mathbf{L}^{-1}. \end{aligned} \tag{8.1.2}$$

Analogicky sestavíme algoritmus pro výpočet matice $\mathbf{Z} = \mathbf{U}^{-1}$:

(8.1.3)
$$\begin{aligned}
&\text{Vstup}: \quad n, \mathbf{U} = (u_{ij}).\\
&\qquad \text{Pro} \quad j = 1, 2, ..., n:\\
&\qquad\qquad \text{Pro} \quad i = j, j-1, ..., 1:\\
&\qquad\qquad\qquad z_{ij} = \frac{1}{u_{ii}} \left(\delta_{ij} - \sum_{k=i+1}^{j} u_{ik} z_{kj} \right).\\
&\text{Výstup}: \quad \mathbf{Z} = (z_{ij}) = \mathbf{U}^{-1}.
\end{aligned}$$

K inverzi matic $\mathbf{L}$, $\mathbf{U}$ tedy potřebujeme $2n^3/3$ operací. Poznamenejme, že k inverźi pomocí GEM potřebujeme řádově také tolik operací.

8.2 Cvičení

8.2.1

Invertujte matici

$$\mathbf{A} = \begin{pmatrix} 2,475\,9 & 1,623\,5 & 4,623\,1 \\ 1,472\,5 & 0,958\,9 & -1,325\,3 \\ 2,695\,1 & 2,896\,5 & -1,479\,4 \end{pmatrix}$$

Počítejte v $M(10,t)$, $t = 5, 6$. Proved'te kontrolu $\mathbf{A}\mathbf{A}^{-1} = \mathbf{I}$. [Řádky matice $\mathbf{A}^{-1}$ (v $M(10,10)$): $(0,208\,769\,758\,4; 1,376\,034\,419; -0,572\,558\,917\,6)$, $(-0,118\,599\,141\,4; -1,406\,206\,69; 0,877\,250\,923\,8)$; $(0,146\,147\,100\,7;$ $-0,243\,115\,454\,7; -0,001\,431\,523\,7)$.]

8.2.2

Je-li $\mathbf{X}_n$ dobrá aproximace matice $\mathbf{A}^{-1}$, tj. $\mathbf{A}\mathbf{X}_n = \mathbf{I} + \mathbf{E}_n$, kde $\|\mathbf{E}\| < 1$ je malé číslo, potom ukažte, že matice $\mathbf{X}_{n+1} = \mathbf{X}_n(2\mathbf{I} - \mathbf{A}\mathbf{X}_n)$ je lepší aproximací matice $\mathbf{A}^{-1}$, tj. že $\|\mathbf{E}_{n+1}\| \leqq \|\mathbf{E}_n\|$.

Návod. $\mathbf{E}_{n+1} = \mathbf{A}\mathbf{X}_{n+1} - \mathbf{I} = -\mathbf{E}_n^2$; dokonce platí $\|\mathbf{E}_{n+1}\| \leqq \|\mathbf{E}_n\|^2$.

8.2.3

Stanovte $\mathbf{A}^{-1}$, když

$$\mathbf{A} = \begin{pmatrix} \frac{1}{2} & \frac{1}{3} \\ \frac{1}{3} & \frac{1}{4} \end{pmatrix}$$

Užijte GJEM nebo LU-rozkladu. Počítejte v $M(10,3)$. [Návod. Invertujte matici

$$\tilde{\mathbf{A}} = \begin{pmatrix} 0,500 & 0,333 \\ 0,333 & 0,250 \end{pmatrix}, \qquad \mathbf{A}^{-1} = \begin{pmatrix} 18 & -24 \\ -24 & 36 \end{pmatrix};$$

$$\tilde{\mathbf{A}}^{-1} = \begin{pmatrix} 17,7 & -23,6 \\ -23,6 & 35,6 \end{pmatrix}; \quad \mathbf{A} - \tilde{\mathbf{A}} = \begin{pmatrix} 0,000 & 3,33 \cdot 10^{-4} \\ 3,33 \cdot 10^{-4} & 0,000 \end{pmatrix};$$

$$\left\| \mathbf{A} - \tilde{\mathbf{A}} \right\|_{\mathbf{R}} = 3,33 \cdot 10^{-4}; \quad \mathbf{A}^{-1} - \tilde{\mathbf{A}}^{-1} = \begin{pmatrix} 3 \cdot 10^{-1} & -4 \cdot 10^{-1} \\ -4 \cdot 10^{-1} & 5 \cdot 10^{-1} \end{pmatrix};$$

$$\left\| \mathbf{A}^{-1} - \tilde{\mathbf{A}}^{-1} \right\|_{\mathbf{R}} = 9,5 \cdot 10^{-1}; \quad \tilde{\mathbf{A}}\tilde{\mathbf{A}}^{-1} = \begin{pmatrix} 0,991 & 2,15 \cdot 10^{-2} \\ -5,9 \cdot 10^{-3} & 1,01 \end{pmatrix};$$

$$\left\| \mathbf{A}\mathbf{A}^{-1} - \tilde{\mathbf{A}}\tilde{\mathbf{A}}^{-1} \right\|_{\mathbf{R}} = 3,05 \cdot 10^{-2}.]$$

8.2.4

Proveřte na příkladech, že inverzní matice k pásové matici (např. k třídiagonální matici) je obecně plná matice. Zdůvodněte tento poznatek!

9 Chyby řešení soustav lineárních rovnic

V článku 4 jsme hovořili o podmíněnosti úloh a o stabilitě algoritmů. Zde nás bude z tohoto hlediska zajímat úloha stanovit řešení soustavy rovnic $\mathbf{Ax} = \mathbf{b}$ jednak algoritmem Gaussovy eliminace, jednak iteračními algoritmy.

Při řešení uvažované úlohy na konkrétním počítači vznikají nepřesnosti (chyby) ze dvou příčin:

a) nepřesností vstupních dat [např. zobrazením vstupních dat do množiny $M(q,t)$],

b) zaokrouhlováním během výpočtu [tj. realizací aritmetických operací v množině $M(q,t)$].

Nejdříve vyšetříme podmíněnost dané úlohy. O stabilitě algoritmů jsme se zmínili v odst. 6.2.5, 6.4.1, 6.4.2, 6.6.2.

9.1 Podmíněnost.

Necht' vstupní chyby prvků regulární matice $\mathbf{A}$ tvoří matici, kterou označíme $\Delta\mathbf{A}$,a vstupní chyby složek vektoru $\mathbf{b}$ tvoří vektor, který označíme $\Delta\mathbf{b}$. Je-li $\mathbf{x}_t$ přesné řešení soustavy $\mathbf{A}\mathbf{x} = \mathbf{b}$, potom přesné řešení soustavy $(\mathbf{A} - \Delta\mathbf{A})\mathbf{x} = \mathbf{b} + \Delta\mathbf{b}$ označíme $\mathbf{x}_t + \Delta\mathbf{x}$, tj. platí

$$(\mathbf{A} + \Delta\mathbf{A})(\mathbf{x}_t + \Delta\mathbf{x}) = \mathbf{b} + \Delta\mathbf{b}. \tag{9.1.1}$$

9.1.1

Uvažujme nejdřívě situaci, kdy matice $\mathbf{A}$ je dána přesně, tj. $\Delta\mathbf{A} = \mathbf{0}$. Z rovnosti (9.1.1) pak plyne

$$\mathbf{A}\Delta\mathbf{x} = \Delta\mathbf{b},$$

tj.

$$\Delta\mathbf{x} = \mathbf{A}^{-1}\Delta\mathbf{b}$$

a odtud dostaneme odhad chyby řešení [zde i níže užíváme (5.2.8)]

$$\|\Delta\mathbf{x}\| \leqq \left\|\mathbf{A}^{-1}\right\| \|\Delta\mathbf{b}\| . \tag{9.1.2}$$

Protože $\mathbf{b} = \mathbf{A}\mathbf{x}_t$, potom $\|\mathbf{b}\| \leqq \|\mathbf{A}\| \|\mathbf{x}_t\|$, tj.

$$\|\mathbf{x}_t\| \geqq \frac{\|\mathbf{b}\|}{\|\mathbf{A}\|}.$$

Pro relativní chyby proto dostáváme

$$\frac{\|\Delta\mathbf{x}\|}{\|\mathbf{x}_t\|} \leqq \left\|\mathbf{A}^{-1}\right\| \|\mathbf{A}\| \frac{\|\Delta\mathbf{b}\|}{\|\mathbf{b}\|}. \tag{9.1.3}$$

Číslo

$$C_p = \left\|\mathbf{A}^{-1}\right\| \|\mathbf{A}\| \tag{9.1.4}$$

je mírou citlivosti relativní chyby řešení na relativní chybě vstupních dat a nazývá se (v souladu s odst. 4.2) *číslem podmíněnosti* matice $\mathbf{A}$, resp. uvažované úlohy.

9.1.2

Máme-li nyní naopak $\Delta\mathbf{b} = \mathbf{0}\,,\ \Delta\mathbf{A} \neq \mathbf{0}$ tj. pravá strana je dána přesně, potom (9.1.1) má tvar

$$(\mathbf{A}-\Delta\mathbf{A})(\mathbf{x}_t-\Delta\mathbf{x}) = \mathbf{b}, \tag{9.1.5}$$

a tedy

$$\Delta\mathbf{x} = -\mathbf{A}^{-1}\Delta\mathbf{A}(\mathbf{x}_t - \Delta\mathbf{x}).$$

Odtud dostáváme odhad pro relativní chyby

$$\frac{\|\mathbf{x}\|}{\|\mathbf{x}_t + \Delta\mathbf{x}\|} \leqq \left\|\mathbf{A}^{-1}\right\| \|\mathbf{A}\| \frac{\|\Delta\mathbf{A}\|}{\|\mathbf{A}\|}. \tag{9.1.6}$$

Opět byl vztah mezi vstupní chybou vyjádřen pomocí čísla $C_p = \|\mathbf{A}^{-1}\| \|\mathbf{A}\|$.

9.1.3

Pro obecný případ ($\Delta\mathbf{A} \neq \mathbf{0}\,,\ \Delta\mathbf{b} \neq \mathbf{0}$) se dá odvodit odhad (viz [3],[12]):

$$\frac{\|\Delta\mathbf{x}\|}{\|\mathbf{x}_t\|} \leqq C_p \frac{\frac{\|\Delta\mathbf{A}\|}{\|\mathbf{A}\|} + \frac{\|\Delta\mathbf{b}\|}{\|\mathbf{b}\|}}{1 - C_p \frac{\|\Delta\mathbf{A}\|}{\|\mathbf{A}\|}}. \tag{9.1.7}$$

9.1.4 Příklad.

Chceme určit číslo podmíněnosti C_p matice

$$\mathbf{A} = \begin{pmatrix} 4,1 & 2,8 \\ 9,7 & 6,6 \end{pmatrix}$$

aniž bychom počítali $\mathbf{A}^{-1}$.

Zvolíme-li $\mathbf{b} = (4,1; 9,7)^T$, potom přesným řešením soustavy $\mathbf{Ax} = \mathbf{b}$ je vektor $\mathbf{x}_t = (1,0)^T$. Změníme-li pravou stranu soustavy na $\mathbf{b} + \Delta\mathbf{b} = = (4,11; 9,70)^T$, bude přesným řešením vektor $\mathbf{x}_t + \Delta\mathbf{x} = (0,34; 0,97)^T$ (přesvědčte se o tom!).

Již odtud je patrné, že úloha řešit soustavu s uvedenou maticí soustavy bude špatně podmíněná (malá změna vstupních dat vyvolala velkou změnu řešení).

K výpočtu čísla podmíněnosti užijeme nerovnosti (9.1.3). Ve sloupcové normě dostaneme $\|\Delta\mathbf{x}\| = 1,63\,,\ \|\Delta\mathbf{b}\| = 0,01\,,\ \|\mathbf{x}_t\| = 1\,,\ \|\mathbf{b}\| = 13,8\,,$

$\|\Delta\mathbf{x}\| / \|\mathbf{x}_t\| = 1,63$, $\|\Delta\mathbf{b}\| / \|\mathbf{b}\| = 0,000\,724\,6$. Z nerovnosti $1,63 \leqq C_p \cdot 0,000\,724\,6$ dostáváme že

$$C_p \geqq 2\,249,5.$$

V ostatních normách bychom dostali podobný výsledek.

9.2 Vliv zaokrouhlovacích chyb.

Základní výsledky o vlivu zaokrouhlovacích chyb při Gaussově eliminaci jsou uvedeny v publikaci Wilkinson, J. H.: Rounding Eroors in Algebraic Processes, N. p. L. Notes on Applied Science, No 32, H. M. S. O., London 1963. Čtenář je najde také např. v [7].

Uved'me si stručně tyto výsledky (viz [4]).

Označíme

$$g_n = \frac{\max\limits_{i,j,k} \left|\bar{a}_{ij}^{(k)}\right|}{\max\limits_{i,j} |a_{ij}|},$$

kde $\bar{a}_{ij}^{(k)}$ jsou vypočtené prvky redukované matice získané během eliminace.

Je-li $\tilde{\mathbf{L}}\tilde{\mathbf{U}} = \mathbf{A} + \mathbf{E}$ *vypočtený LU-rozklad matice* $\mathbf{A}$ (*s částečným nebo úplným výběrem*), *potom platí odhad*

$$\|\mathbf{E}\|_{\mathbf{R}} \leqq n^2 g_n \|\mathbf{A}\|_{\mathbf{R}} q^{-t}. \tag{9.2.1}$$

Je-li $\mathbf{x}_c$ *vypočtené řešení soustavy* $\mathbf{Ax} = \mathbf{b}$ *získané řešením soustav* $\tilde{\mathbf{L}}\mathbf{y} = \mathbf{b}$, $\tilde{\mathbf{U}}\mathbf{x} = \mathbf{y}_c$ *potom existuje matice* $\Delta\mathbf{A}$ (*závisející jak na* $\mathbf{A}$, *tak na* $\mathbf{b}$) *pro niž platí odhad*

$$\|\Delta\mathbf{A}\|_{\mathbf{R}} \leqq (n^3+3n) g \|\mathbf{A}\|_{\mathbf{R}} q^{-1} \tag{9.2.2}$$

a taková, že $\mathbf{x}_c$ (*vypočtené řešení soustavy* $\mathbf{Ax} = \mathbf{b}$) *je přesným řešením soustavy*

$$(\mathbf{A}-\Delta\mathbf{A})\mathbf{x}_c = \mathbf{b}. \tag{9.2.3}$$

Matice $\Delta\mathbf{A}$ tedy reprezentuje zaokrouhlovací chyby akumulované během realizace Gaussovy eliminace. Užijeme-li postupu z odst. 9.1.2, dostaneme ($\mathbf{x}_t = \mathbf{A}^{-1}\mathbf{b}$)

$$\frac{\|\mathbf{x}_t - \mathbf{x}_c\|}{\|\mathbf{x}_c\|} \leqq C_p \varrho(n) q^{-1}, \tag{9.2.4}$$

kde jsme označili $\varrho(n) = (n^3 + 3n) g_n$.

Ze vztahu (9.2.4) je patrné, že k posouzení vlivu zaokrouhlovacích chyb při realizaci eliminační metody potřebujeme určit nebo alespoň odhadnout číslo C_p. Obvykle je přibližný výpočet čísla podmíněnosti součástí programu eliminační metody (viz [7a]).

Zde uvádená hodnota $\varrho(n)$ je pesimistická. Velmi vzácně bývá $\varrho(n)$ větší než q. Ovšem Wilkinson sestrojil matici $\mathbf{A}$ (viz např. [16]), u které se při sloupcovém výběru dostane

$$0 < g_n \leqq 2^{n-1}.$$

9.3 Poznámka.

Ukázali jsme si, že zaokrouhlovací chyby vznikající v průběhu výpočtu lze převést na jisté (umělé) chyby ve vstupních datech, a pokud víme něco o podmíněnosti úlohy, můžeme podle vztahu (9.2.4) posoudit vliv zaokrouhlovacích chyb na řešení úlohy.

Pro špatně podmíněnou úlohu (velké C_p) malá změna ve vstupních datech vyvolá velkou změnu řešení, přičemž tato změna řešení je ještě zesiluje vlivem zaokrouhlovacích chyb.

Jsou-li vstupní data špatně podmíněné soustavy rovnic získána např. měřením, musíme řešení této soustavy získané jakoukoliv numerickou metodou interpretovat velmi opatrně. Rozumnější je vždy se snažit danou úlohu přeformulovat tak, abychom špatně podmíněné soustavy nemuseli řešit.

Vzniká ovšem přirozená otázka, zda lze získat i pro špatně podmíněné soustavy řešení s vyhovující přesností. Odpoved' je kladná, pokud investujeme do řešení soustavy určitou práci navíc. Výpočet ve dvojnásobné aritmetice je ovšem netností. Určité informace o této problematice najde čtenář např. v [4], [8], [12] a především v [7].

K Získání určité informace o vlivu zaokrouhlovacích chyb na řešení soustavy se v praxi užívá jednoduchého obratu. Spolu se soustavou $\mathbf{Ax} = \mathbf{b}$ se řeší soustava $\mathbf{Ax} = \mathbf{d}$, kde

$$d_i = \sum_{k=1}^{n} a_{ik} \qquad i = 1, 2, ..., n,$$

jejímž přesným řešením je vektor $(1, 1, 1, ..., 1)^T$.

Metoda experimentálních perturbací nám také může poskytnout představu o přesnosti vypočteného řešení.

9.4 Zaokrouhlovací chyby u iterační metody.

Uvažujeme uteračí proces určený formulí

$$\mathbf{x}^{(k+1)} = \mathbf{H}\mathbf{x}^{(k)}+\mathbf{g} \tag{9.4.1}$$

a předpokládejme, že tento proces konverguje k přesnému řešení $\mathbf{x}_t$, tj.

$$\mathbf{x}_t = \mathbf{H}\mathbf{x}_t+\mathbf{g}. \tag{9.4.2}$$

Předpokládejme, že počáteční aproximace $\mathbf{x}^{(0)}$ je dána přešeně. V důsledku nepřesné realizace aritmetických operací v počítači se při výpočtu $\mathbf{x}^{(1)}$ podle formule (9.4.1) nevypočte přesná iterace $\mathbf{H}\mathbf{x}^{(0)} + \mathbf{g} = \mathbf{x}^{(1)}$, ale iterace $\tilde{\mathbf{x}}^{(1)}$, která není přesně rovna iteraci $\mathbf{x}^{(1)}$, tj.

$$\tilde{\mathbf{x}}^{(1)} = \mathbf{H}\mathbf{x}^{(0)} + \mathbf{g} + \eta^{(1)},$$

kde $\eta^{(1)} = (\eta_{11}, \eta_{21}, ..., \eta_{n1})^T$ (η_{n1} je chyba, s jakou vypočteme i-tou složku vektoru $\mathbf{x}^{(1)}$). V dalším kroku vypočteme iteraci $\tilde{\mathbf{x}}^{(2)}$, která se opět liší od přesné hodnoty $\mathbf{H}\tilde{\mathbf{x}}^{(1)} + \mathbf{g}$ - nyní o vektor $\tilde{\eta}^{(2)}$, tj.

$$\tilde{\mathbf{x}}^{(2)} = \mathbf{H}\tilde{\mathbf{x}}^{(1)} + \mathbf{g} + r^{(2)}.$$

V k-tém kroku máme tedy

$$\tilde{\mathbf{x}}^{(k+1)} = \mathbf{H}\tilde{\mathbf{x}}^{(k)}+\mathbf{g}+\eta^{(k+1)}, \tag{9.4.3}$$

kde $\eta^{(k+1)}$ reprezentuje zaokrouhlovací chybu při výpočtu $(k+1)$-ní iterace. Odečteme-li rovnosti (9.4.3) a (9.4.2), pak máme

$$\tilde{\mathbf{x}}^{(k+1)} - \mathbf{x}_t = \mathbf{H}(\tilde{\mathbf{x}}^{(k)} - \mathbf{x}_t) + \eta^{(k+1)}.$$

Označíme-li

$$\xi^{(k)} = \tilde{\mathbf{x}}^{(k)} - \mathbf{x}_t,$$

potom dostáváme

$$\xi^{(k+1)} = \mathbf{H}^{(k+1)}\xi^{(0)} + \mathbf{H}^{(k)}\eta^{(1)} + \mathbf{H}^{(k-1)}\eta^{(2)} + ... + \mathbf{H}^{(k)} + \eta^{(k+1)}.$$

Protože $\xi^{(0)} = \mathbf{x}^{(0)} - \mathbf{x}_t = \mathbf{e}^{(0)}$ (viz odst. 7.5.2), dostaneme z předchozí rovnosti odhad celkové chyby

$$\left\|\xi^{(k+1)}\right\| \leqq \left\|\mathbf{e}^{(k+1)}\right\|+\|\mathbf{H}\|^k \left\|\eta^{(1)}\right\|+\|\mathbf{H}\|^{k-1} \left\|\eta^{(2)}\right\|+...+\|\mathbf{H}\| \left\|\eta^{(k)}\right\|+\left\|\eta^{(k+1)}\right\|.$$

Po úpravě

$$\left\|\xi^{(k+1)}\right\| \leqq \left\|\mathbf{e}^{(k+1)}\right\| + C\frac{1-\|\mathbf{H}\|^{k+1}}{1-\|\mathbf{H}\|}: \quad C = \max_j \|\eta(j)\|\,. \tag{9.4.4}$$

Konstanta C je horní hranice zaokrouhlovacích chyb a závisí jak na typu počítače [na $M(q,t)$], tak na matici $\mathbf{H}$ a vektoru $\mathbf{g}$. Pro $k \to \infty$ je $\left\|\mathbf{e}^{(k+1)}\right\| \to 0$ a $\left\|\xi^{(k+1)}\right\|$ se bude od nuly lišit řádově o C, pokud nebude $\mathbf{H} \approx 1$.

Druhý člen v nerovnosti (9.4.4) bude činit potíže pouze při pomalé konvergenci procesu (9.4.1) (to ovšem může být dost často!). Pro velká k se totiž může skutečná chyba $\xi^{(k)}$ lišit od teoretické chyby metody $\mathbf{e}^{(k)}$ (viz odst. 7.5.3) dokonce o několik řádů.

9.5 Cvičení.

9.5.1

Určete číslo podmíněnosti soustavy $\mathbf{Ax} = \mathbf{b}$, kde

$$\mathbf{A} = \begin{pmatrix} 1,296\,9 & 0,864\,8 \\ 0,216\,1 & 0,144\,1 \end{pmatrix}, \qquad \mathbf{b} = \begin{pmatrix} 0,864\,2 \\ 0,144\,0 \end{pmatrix}$$

Vypočtete $\mathbf{r} = \mathbf{b} - \mathbf{Az}$, když $\mathbf{z} = (0,991\,1, -0,487\,0)^T$. Porovnejte řešení vypočtené eliminační metodou s přesným řešením $\mathbf{x}_t = (2,-2)^T$.

9.5.2

Řešte ekvivalentní soustavy

$$\begin{array}{llll} a) & x_1 + 10\,000 & x_2 = 10\,000; & b) \quad 0,000\,1x_1 + x_2 = 1, \\ & x_1 + & 0,000\,1x_2 = 1. & x_1 + 0,000\,1x_2 = 1. \end{array}$$

Užijte algoritmu řádkového výběru. Vypočtete číslo podmíněnosti obou úloh. Posud'te podmíněnost obou úloh a užitého algoritmu. Počítejte v $M(10,4)$.

$\mathbf{x}_t = (0,999\,9; 0,999\,9)^T$, a) $\mathbf{x}_c = (1,000; 0,000\,0)^T$; b) $\mathbf{x}_c = (1,00; 1,000)^T$.

10 Vlastní čísla a vlastní vektory

S pojmem vlastního čísla matice jsme se již setkali v odst. 7.5.1, kdy jsme podle velikosti vlastních čísel iterační matice $\mathbf{H}$ posuzovali konvergenci iterační metody. S úlohou na vlastní čísla kterou jsme formulovali v odst. 5.3.3, se však můžeme setkat i v aplikacích při řešení celé řady technických nebo fyzikálních problémů. Jmenujme např. problém vzpěrné pružnosti problém chvění fyzikálních soustav některé úlohy z kvantové fyziky atd. (viz např. [10], [10a]).

Připomeňme si úlohu na vlastní čísla na jednoduchém příkladu.

10.1 Příklad.

Stanovme taková čísla λ, pro která má homogenní soustavy $\mathbf{Av} = \lambda\mathbf{v}$ nenulové řešení, a určeme toto řešení. Volme

$$\mathbf{A} = \begin{pmatrix} 2 & 0 & 0 \\ 2 & 2 & 1 \\ 1 & 1 & 2 \end{pmatrix}.$$

Řešíme tedy soustavu

$$(\mathbf{A} - \lambda\mathbf{I})\mathbf{v} = \begin{pmatrix} 2-\lambda & 0 & 0 \\ 2 & 2-\lambda & 1 \\ 1 & 1 & 2-\lambda \end{pmatrix} \begin{pmatrix} v_1 \\ v_2 \\ v_3 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}.$$

Aby homogenní soustava měla nenulové řešení, musí být determinant soustavy nulový. Najdeme proto taková λ, aby

$$\det(\mathbf{A} - \lambda\mathbf{I}) = -\lambda^3 + 6\lambda^2 - 11\lambda + 6 = 0.$$

Dostali jsme tak algebraickou rovnici 3. stupně a pouze pro její kořeny

$$\lambda_1 = 3, \quad \lambda_2 = 2, \quad \lambda_3 = 1, \tag{10.1.1}$$

nazýváme vlastní čísla matice $\mathbf{A}$, bude mít uvažovaná soustava nenulové řešení.

Stanovíme nyní tedy nenulové řešení tři homogenních soustav

$$(\mathbf{A} - \lambda_i\mathbf{I})\mathbf{v} = 0, \qquad i = 1, 2, 3.$$

odpovídajících číslům (10.1.1). Řešíme tedy tři soustavy

$$\begin{pmatrix} -1 & 0 & 0 \\ 2 & -1 & 1 \\ 1 & 1 & -1 \end{pmatrix} \begin{pmatrix} v_1 \\ v_2 \\ v_3 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}; \quad \begin{pmatrix} 0 & 0 & 0 \\ 2 & 0 & 1 \\ 1 & 1 & 0 \end{pmatrix} \begin{pmatrix} v_1 \\ v_2 \\ v_3 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix};$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 2 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} v_1 \\ v_2 \\ v_3 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}.$$

Řádkovými úpravami (viz též odst. 6.3.8) zjistíme, že pro libovolná nenulová čísla r, s, t jsou hledaným řešením tyto vektory (v uvedeném pořadí) - tzv. vlastní vektory:

$$(0, r, r)^T, \quad (s, -s, -2s)^T, \quad (0, t, -t)^T;$$

např.

$$\mathbf{v}^{(1)} = (0, 1, 1)^T, \quad \mathbf{v}^{(2)} = (1, -1, -2)^T, \quad \mathbf{v}^{(3)} = (0, 1, -1)^T.$$

Stojí za povšimnutí, že vlastní čísla dané matice jsou navzájem různá a odpovídající vlastní vektory jsou lineárně nezávislé.

10.2 Základní poznatky.

10.2.1 Definice.

Mějme čtvercovou matici $\mathbf{A}$ řádu n. Číslo λ (obecně komplexní), pro které má homogenní soustava

(10.2.1) $$\mathbf{A}\mathbf{v} = \lambda\mathbf{v}, \quad \text{resp.} \quad (\mathbf{A}-\lambda\mathbf{I})\mathbf{v} = \mathbf{0}$$

nenulové řešení, se nazývá *vlastní číslo matice* $\mathbf{A}$ [značíme $\lambda = \lambda(\mathbf{A})$] a jemu odpovídající nenulové řešení $\mathbf{v} = (v_1, v_2, ..., v_n)^T$ *vlastní vektor matice* $\mathbf{A}$.

Vlastním číslem matice $\mathbf{A}$ je právě to číslo λ, které je kořenem *charakteristického polynomu*

(10.2.2) $$p_{\mathbf{A}}(\lambda) = \det(\mathbf{A}-\lambda\mathbf{I}) = (-1)^n\lambda^n + b_1\lambda^{n-1} + ... + b_{n-1}\lambda + b_n.$$

Odtud vyplývá, že každá čtvercová matice řádu n má právě n vlastních čísel $\lambda_1, \lambda_2, ..., \lambda_n$, přičemž každé vlastní číslo počítáme tolikrát, jaká je

jeho násobnost. *Jednoduché*, resp. *k-násobné číslo* je jednoduchým, resp. k-násobným kořenem polynomu $p_{\mathbf{A}}(\lambda)$.

Vlastní čísla matice můžeme tedy vypočitat jako kořeny charakteristického polynomu metodami z kap. III. Tento postup však narází na ynačné potíže při větších n (velký objem výpočtů), pokud koeficienty polynomu chceme počítat přímo z definice determinantu.

Matici $\mathbf{A} = \mathrm{diag}\,(\lambda_1, \lambda_2, ..., \lambda_n)$ nazýváme *spektrální maticí* matice $\mathbf{A}$.

Je-li $\mathbf{v}^{(i)}$ vlastní vektor matice $\mathbf{A}$ odpovídající vlastnímu číslu λ_i, potom rovnosti $\mathbf{A}\mathbf{v}^{(i)} = \lambda_i \mathbf{v}^{(i)}$, $i = 1, 2, ..., n$, vyplývající z (10.2.1), můžeme zapsat ve tvaru maticové rovnosti

$$\mathbf{A}\mathbf{X} = \mathbf{X}\mathbf{\Lambda}, \tag{10.2.3}$$

kde sloupce matice $\mathbf{X}$ jsou vlastní vektory matice $\mathbf{A}$, tj. $\mathbf{s}_i(\mathbf{X}) = \mathbf{v}^{(i)}$.

Pro matici $\mathbf{A}$ z příkl. 10.1 je

$$\mathbf{X} = \begin{pmatrix} 0 & 1 & 0 \\ 1 & -1 & 1 \\ 1 & -2 & -1 \end{pmatrix}, \quad \mathbf{\Lambda} = \begin{pmatrix} 3 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

Úloze najít všechna vlastní čísla dané matice se říká *úplný problém vlastních čísel*. Úloze najít pouze některá vlastní čísla dané matice (obvykle s největší nebo s nejmenší absolutní hodnotou) se říká *částečný problém vlastních čísel*.

Vlastní čísla diagonální a trojúhelníkové matice jsou rovna diagonálním prvkům této matice, nebot' charakteristický polynom má v tomto případě tvar $p_{\mathbf{A}}(\lambda) = (a_{11} - \lambda)(a_{22} - \lambda)...(a_{nn} - \lambda)$.

10.2.2 Příklad.

Uvažujme matice

$$\mathbf{A} = \left(\begin{array}{c|cc} 2 & 0 & 0 \\ \hline 0 & 2 & 0 \\ 0 & 0 & 2 \end{array}\right), \quad \mathbf{B} = \left(\begin{array}{cc|c} 2 & 1 & 0 \\ 0 & 2 & 0 \\ \hline 0 & 0 & 2 \end{array}\right),$$

$$\mathbf{C} = \left(\begin{array}{c|cc} 2 & 0 & 0 \\ \hline 0 & 2 & 1 \\ 0 & 0 & 2 \end{array}\right), \quad \mathbf{D} = \begin{pmatrix} 2 & 1 & 0 \\ 0 & 2 & 1 \\ 0 & 0 & 2 \end{pmatrix}.$$

Pro tyto matice je

$$p_{\mathbf{A}}(\lambda) = p_{\mathbf{B}}(\lambda) = p_{\mathbf{C}}(\lambda) = p_{\mathbf{D}}(\lambda) = (2 - \lambda)^3,$$

tzn. že číslo $\lambda = 2$ je vlastním číslem (trojnásobným) všech čtyř matic. Postupem z příkl. 10.1 zjistíme, že vektory

$$\mathbf{v}^{(1)} = (1,0,0)^T, \quad \mathbf{v}^{(2)} = (0,1,0)^T, \quad \mathbf{v}^{(3)} = (0,0,1)^T$$

jsou vlastními vektory matice $\mathbf{A}$ odpovídající vlastnímu číslu 2. Každá nenulová lineární kombinace vektorů $\mathbf{v}^{(1)}, \mathbf{v}^{(2)}, \mathbf{v}^{(3)}$ je také vlastním vektorem odpovídajícím témuž vlastnímu číslu.

U matice $\mathbf{B}$ jsou vlastními vektory pouze vektory $\mathbf{v}^{(1)}, \mathbf{v}^{(3)}$ (a jejich libovolná lineární kombinace). U matice $\mathbf{C}$ pak pouze vektory $\mathbf{v}^{(1)}, \mathbf{v}^{(2)}$ a u matice $\mathbf{D}$ pouze vektor $\mathbf{v}^{(1)}$. Vidíme, že počet lineárně nezávislých vlastních vektorů není u některých matic roven řádu matice, ale je menší!

10.2.3 Podobné matice.

Říkáme, že matice $\mathbf{A}$ a $\mathbf{B}$ jsou *podobné*, existuje-li regulární matice $\mathbf{P}$ taková, že

$$\mathbf{P}^{-1}\mathbf{A}\mathbf{P} = \mathbf{B}, \quad \text{resp.} \quad \mathbf{A} = \mathbf{P}\mathbf{B}\mathbf{P}^{-1}. \tag{10.2.4}$$

Protože $\det(\mathbf{B} - \lambda\mathbf{I}) = \det \mathbf{P}^{-1}(\mathbf{A} - \lambda\mathbf{I})\mathbf{P} = \det(\mathbf{A} - \lambda\mathbf{I})$ (viz odst. 5.1.1), plyne odtud, že podobné matice mají tatáž vlastní čísla.

Z rovnosti vlastních čísel matic však nevyplývá jejich podobnost. Např. matice z příkl. 10.2.2 nejsou navzájem podobné!

Je-li $\mathbf{v}$ *vlastní vektor matice* $\mathbf{A}$, *potom* $\mathbf{P}^{-1}\mathbf{v}$ *je vlastní vektor matice* $\mathbf{B}$ *odpovídající témuž vlastnímu číslu.* (Prověřte!)

Pokud vlastní vektory $\mathbf{v}^{(1)}, \mathbf{v}^{(2)}, ..., \mathbf{v}^{(n)}$ *matice* $\mathbf{A}$ *jsou lineárně nezávislé, bude matice* $\mathbf{X}$ *v rovnosti* (10.2.3) *regulární a plyne odtud*

$$\mathbf{X}^{-1}\mathbf{A}\mathbf{X} = \mathbf{\Lambda}, \tag{10.2.5}$$

tzv. matice $\mathbf{A}$ *je v tomto případě podobná diagonální matici.*

Tato situace nastává např., když matice $\mathbf{A}$ má všechna vlastní čísla různá nebo když je symetrická.

Ovšem ne každá matice je podobná diagonální maticí. Dá se však dokázat (viz např. [9]), že libovolná matice $\mathbf{A}$ řádu n je podobná tzv. *Jordanově matici*

$$\mathbf{J} = \operatorname{diag}(\mathbf{J}_1, \mathbf{J}_2, ..., \mathbf{J}_r), \quad r \leqq n,$$

kde $\mathbf{J}_k$, $k = 1, 2, ..., r$, je matice s vlastním číslem λ_k matice $\mathbf{A}$ na diagonále a s jedničkami nad hlavní diagonálou a s ostatními prvky rovnými nule. Maticím $\mathbf{J}_k$ říkáme *Jordanovy bloky.*

Je-li řád všech Jordanových bloků roven jedné, potom

$$\mathbf{J} = \mathrm{diag}\,(\lambda_1, \lambda_2, ..., \lambda_n) = \mathbf{\Lambda}.$$

U matic z příkl. 10.2.2 jsou Jordanova bloky čárkovaně ohraničeny; matice $\mathbf{D}$ je přímo Jordanovým blokem.

Řada numerických metod řešení úplného problému vlastních čísel matice $\mathbf{A}$ využívá vztahu podobnosti (10.2.4).

10.2.4

Uved'me si ještě některé další užitečné poznatky z lineární algebry, vyplyvající [až na (6)] přímo z (10.2.1):

(1) $\lambda(\mathbf{A}^{-1}) = (\lambda(\mathbf{A}))^{-1}$.
(2) $\lambda(\mathbf{A}^k) = (\lambda(\mathbf{A}))^k$, k *celé číslo.*
(3) $\lambda(\mathbf{A}^H) = \bar{\lambda}(\mathbf{A})$, $\bar{\lambda}$ *je číslo komplexně sdružené k* λ.
(4) *Vlastní čísla symetrické matice* ($\mathbf{A}^H = \mathbf{A}$) *jsou reálná.*
(5) *Vlastní vektory symetrické matice odpovídající různým vlastním číslům jsou ortogonální.*
(6) *k-násobnému vlastnímu číslu symetrické matice odpovídá právě k lineárně nezávislých vlastních vektorů, tj. symetrická matice řádu n má právě n lineárně nezávislých vlastních vektorů a lze je vybrat ortogonální. Potom také platí (10.2.5), kde v tomto případě je* $\mathbf{X}$ *ortogonální matice (pro komplexní matice se užívá termínu unitární.* Shrnuto: *Symetrická matice je podobná diagonální matici.*
(7) *Symetrická pozitivně definitní matice má všechna vlastní čísla kladná.*

Pokud vektor $\mathbf{x}$ není vlastním vektorem reálné symetrické matice $\mathbf{A}$, potom číslo

$$\lambda_{\mathbf{R}} = \frac{\mathbf{x}^T\mathbf{A}\mathbf{x}}{\mathbf{x}^T\mathbf{x}} \tag{10.2.6}$$

se nazývá *Rayleighův podíl* a hraje významnou roli v aplikacích.

Pokud $\mathbf{v}$ je vlastním vektorem odpovídajícím vlastnímu číslu λ reálné symetrické matice $\mathbf{A}$, potom

$$\lambda = \frac{\mathbf{v}^T\mathbf{A}\mathbf{v}}{\mathbf{v}^T\mathbf{v}} \tag{10.2.7}$$

10.2.5 Odhad polohy vlastních čísel.

Ve [12] je dokázáno následující tvrzení: *Všechna vlastní čísla matice* $\mathbf{A} = (a_{ij})$ *lezí ve sjednocení kruhů (v komplexní rovině)*

$$|z-a_{ii}| \leqq \sum_{\substack{j=1 \\ j\neq i}}^{n} |a_{ij}|. \tag{10.2.7}$$

Kromě toho platí pro nejmenší (v absolutní hodnotě) vlastní číslo odhad (viz [3])

$$|\lambda_{min}| \leqq \sqrt[n]{r_1 r_2 r_3 ... r_n}\,, \quad r_i = \left(\sum_{j=1}^{n} |a_{ij}|^2 \right)^{1/2}. \tag{10.2.8}$$

10.2.6 Příklad.

Odhadněme polohu vlastních čísel matic

$$\mathbf{A} = \begin{pmatrix} 5 & 1 & 1 \\ -1 & 1 & 0 \\ 0 & 0,5 & 0 \end{pmatrix}, \quad \mathbf{B} = \begin{pmatrix} 2 & 0 & 0 \\ 2 & 2 & 1 \\ 1 & 1 & 2 \end{pmatrix}$$

U matice $\mathbf{A}$ máme podle (10.2.7) (nakreslete si obrázek!) $|z-5| \leqq |1|+|1| = 2$, $|z-1| \leqq |-1|+|0| = |1|$, $|z-0| \leqq |0|+|-0,5| = 0,5$.
U matice $\mathbf{B}$ máme: $|z-2| \leqq 0$, $|z-2| \leqq 3$, $|z-2| \leqq 2$.
Některou z níže uvedených metod můžeme zjistit [v $M(10,4)$], že $\lambda_1(\mathbf{A}) = 0,108\,5$, $\lambda_2(\mathbf{A}) = 1,148$, $\lambda_3(\mathbf{A}) = 4,786$.
Z příkladu 10.1 víme, že $\lambda_1(\mathbf{B}) = 3$, $\lambda_2(\mathbf{B}) = 2$, $\lambda_3(\mathbf{B}) = 1$

10.2.7 Podmíněnost úlohy na vlastní čísla.

Omezíme se na případ, kdy matice má = lineárně nezávislých vlastních vektorů $\mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_n$ odpovídajících vlastním číslům $\lambda_1, \lambda_2, ..., \lambda_n$.

Označíme Δa_{kj} malé změny v prvcích matice $\mathbf{A} = (a_{ij})$, $(|\Delta a_{kj} \leqq \varepsilon|)$. Necht' dále $\lambda_i(\varepsilon) = \lambda_i + \Delta\lambda_i$ jsou vlastní čísla porušené matice $\mathbf{A}(\varepsilon) = \mathbf{A}+\Delta\mathbf{A}$. Za uvedených předpokladů se dá odvodit (viz např. [3]) přibližný odhad

$$|\lambda_i(\varepsilon)-\lambda_i| \lesssim \chi_i \varepsilon, \tag{10.2.9}$$

kde $\chi_i = 1/|\cos\alpha_i|$ a α_i je úhel vektoru $\mathbf{v}_i$ a vlastního vektoru matice $\mathbf{A}^H$ odpovídajícího vlastnímu číslu $\bar{\lambda}_i$. Pro symetrickou matici (hermitovsky symetrickou) je $\alpha_i = 0$ a tedy platí odhad

$$|\lambda_i(\varepsilon) - \lambda_i| \lesssim \varepsilon,$$

z jehož usoudíme, že úloha na vlastní čísla je v tomto případě dobře podmíněná.

Jestliže χ_i bude velké číslo, může být uvažovaná úloha špatně podmíněná.

Zajímavý je následující příklad. Charakteristický polynom matice

$$\mathbf{A} = \begin{pmatrix} 20 & 20 & 0 & \dots & \dots & \dots & 0 \\ 0 & 19 & 20 & \ddots & & & \vdots \\ 0 & 0 & 18 & 20 & \ddots & & \vdots \\ \vdots & & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & & \ddots & \ddots & \ddots & 0 \\ 0 & & & & \ddots & 2 & 20 \\ \varepsilon & 0 & \dots & \dots & \dots & 0 & 1 \end{pmatrix}.$$

je $p_{\mathbf{A}}(\lambda) = \det(\mathbf{A} - \lambda\mathbf{I}) = (20-\lambda)(19-\lambda)...(2-\lambda)(1-\lambda) - 20^{19}\varepsilon$. Když $\varepsilon = 0$ bude $\lambda_{20} = 1$ nejmenší vlastní číslo. Zvolíme-li $\varepsilon = 20! \cdot 20^{-19} \approx 5 \cdot 10^{-7}$, bude $\lambda_{20} = 0$.

Z tohoto příkladu lze usoudit, že vlastní čísla nesymetrických matic budou citlivá na změny prvků matice. Podrobnějším rozborem lze ukázat, že tato citlivost se zvětšuje se "vzdáleností" od hlavní diagonále vlastní čísla příliš citlivá nejsou.

10.3 Částečný problém vlastních čísel

10.3.1 Mocninná metoda.

Zabývejme se úlohou určit vlastní číslo matice $\mathbf{A}$ s největší absolutní hodnotou (tzv. *dominantní vlastní číslo*).

Předpokládáme, že matice $\mathbf{A}$ má n lineárně nezávislých vlastních vektorů, jediné dominantní vlastní číslo a ostatní vlastní čísla jsou seřazena následujícím způsobem:

$$|\lambda_1| > |\lambda_2| \geqq |\lambda_3| \geqq ... \geqq |\lambda_n|.$$

Zvolíme vektor $\mathbf{y}^{(0)}$ a vyjádříme jej ve tvaru lineární kombinace vlastních vektorů

$$\mathbf{y}^{(0)} = \alpha_1\mathbf{v}_1 + \alpha_2\mathbf{v}_2 + ... + \alpha_n\mathbf{v}_n.$$

Sestrojíme nyní posloupnost vektroů $\{\mathbf{y}^{(k)}\}$ pomocí rekurentního vztahu

$$\mathbf{y}^{(k)} = \mathbf{A}\mathbf{y}^{(k-1)}, \quad k = 1, 2, 3, \ldots . \tag{10.3.1}$$

Protože je $\mathbf{y}^{(k)} = \mathbf{A}^k\mathbf{y}^{(0)}$ a $\mathbf{A}\mathbf{v}_i = \lambda_i\mathbf{v}_i$, pak odtud plyne

$$\mathbf{y}^{(k)} = \alpha_1\mathbf{A}^k\mathbf{v}_1 + \alpha_2\mathbf{A}^k\mathbf{v}_2 + \ldots + \alpha_n\mathbf{A}^k\mathbf{v}_n = \alpha_1\lambda_1^k\mathbf{v}_1 + \alpha_2\lambda_2^k\mathbf{v}_2 + \ldots + \alpha_n\lambda_n^k\mathbf{v}_n$$

neboli

$$\mathbf{y}^{(k)} = \lambda_1^k(\alpha_1\mathbf{v}_1 + \varepsilon_k), \quad \varepsilon_k = \sum_{i=2}^{n} \alpha_i \left(\frac{\lambda_i}{\lambda_1}\right)^k \mathbf{v}_i, \tag{10.3.2}$$

a také

$$\mathbf{y}^{(k+1)} = \mathbf{A}\mathbf{y}^{(k)} = \lambda_1^{k+1}(\alpha_1\mathbf{v}_1 + \varepsilon_{k+1}), \quad \varepsilon_{k+1} = \frac{1}{\lambda_1}\mathbf{A}\varepsilon_k. \tag{10.3.3}$$

Pro j-tou složku vektoru $\mathbf{y}^{(k)}$ z (10.3.2) plyne:

$$y_j^{(k)} = \lambda_1^k(\alpha_1 v_{1j} + \varepsilon_{kj}),$$

kde v_{1j}, ε_{kj} jsou =-té složky vektorů ==. Protože podle předpokladu je

$$|\lambda_i/\lambda_1| < 1, \quad i \geqq 2, \text{potom} \quad \lim_{k\to\infty} \varepsilon_k = \lim_{k\to\infty} \varepsilon_{k+1} = \mathbf{0},$$

a bude

$$\lambda_1 = \lim_{k\to\infty} \frac{y_j^{(k+1)}}{y_j^{(k)}} = \lim_{k\to\infty} \frac{\lambda_1^{k+1}(\alpha_1 v_{1j} + \varepsilon_{kj})}{y_j^k(\alpha_1 v_{1j} + \varepsilon_{k+1,j})}. \tag{10.3.4}$$

Pro dostatečně velká k budou podíly $\lambda_1^{(k)} = y_j^{(k+1)}/y_j^{(k)}$, $j = 1, 2, \ldots, n$ aproximovat vlastní číslo λ_1 ($\varepsilon_{kj} \to 0$ pro $k \to \infty$.

Podle (10.3.2) se $\lim_{k\to\infty} \mathbf{y}^{(k)}$ liší od vlastního vektoru $\mathbf{v}_1$ pouze multiplikativní konstantou.

Protože v praktických úlohách obvykle nemáme informace, zda jsou splněny předpoklady metody, může nám konvergence procesu (10.3.1) činit potíže. Např. nevhodná volba vektoru $\mathbf{y}^{(0)}$ muže znemožnit výpočet. Třeba volíme-li $\mathbf{y}^{(0)}$ tak, že bude $\alpha_1 = 0$ nebo bude blízké nule. K zastavení procesu iterací obvykle slouží podmínka $|\lambda_1^{(k)} - \lambda_1^{(k+1)}| < \delta$. Je velmi obtížné určit odhad chyby metody. Abychom se vyvarovali přeplnění počítače, doporučuje se v každé iteraci normalizovat vektory $\mathbf{y}^{(k)}$ (např. aby měly největší složku rovnou jedné).

Mocninnou metodu lze modifikovat tak, abychom mohli počítat i vícenásobná dominantní vlastní čísla se stejnou absolutní hodnotou, případně další vlastní čísla $\lambda_1, \lambda_2, ...,$ pokud máme předcházející čísla vypočtena. Ve všech těchto případech nelze obecně zaručit konvergenci metody a je třeba provést podrobnější analýzu metody. V praktických úlohách většinou iterace konvergují pomalu a je třeba použit nějakého urychlovacího procesu (viz [12]). Také skutečnost, že na jednu iteraci potřebujeme řádově n^2 operací, může pro větší n podstatně omezit efektivnost metody. Pro pásové matice je v tomto případě situace lepší.

10.3.2 Metoda Rayleighova podílu.

Předpokládáme navíc (ve srovnání s odst. 10.3.1), že matice $\mathbf{A}$ je symetrická (reálná): Potom vlastní vektory jsou ortonormální, tj. $\mathbf{v}_i^T \mathbf{v}_j^T = 0$, $i \neq j$. Potom ze vztahů (10.3.2), (10.3.3) dostaneme

$$\mathbf{y}^{(k)^T} \mathbf{y}^{(k)} = \lambda_1^{2k} (\alpha_1 \mathbf{v}_1^T + \varepsilon_k^T)(\alpha_1 \mathbf{v}_1 + \varepsilon_{k+1}) = \lambda_1^{2k+1} (\alpha_1^2 + \varepsilon_k^T \varepsilon_k),$$

$$\mathbf{y}^{(k)^T} \mathbf{A} \mathbf{y}^{(k)} = \lambda_1^{k} (\alpha_1 \mathbf{v}_1^T + \varepsilon_k^T) \lambda_1^{(k+1)} (\alpha_1 \mathbf{v}_1 + \varepsilon_k) = \lambda_1^{2k} (\alpha_1^2 + \varepsilon_k^T \varepsilon_{k+1}),$$

Protože opět $\lim\limits_{k\to\infty} \varepsilon_k^T \varepsilon_k = \lim\limits_{k\to\infty} \varepsilon_k^T \varepsilon_{k+1} = 0$, když $|\lambda_i/\lambda_1| < 1$, $i \geqq 2$, pak

$$\lambda_1 = \lim_{k\to\infty} \frac{\mathbf{y}^{(k)^T} \mathbf{A} \mathbf{y}^{(k)}}{\mathbf{y}^{(k)^T} \mathbf{y}^{(k)}} = \lim_{k\to\infty} \frac{\mathbf{y}^{(k)^T} \mathbf{y}^{(k+1)}}{\mathbf{y}^{(k)^T} \mathbf{y}^{(k)}}. \tag{10.3.5}$$

Protože $\varepsilon_k^T \varepsilon_k$, resp. $\varepsilon_k^T \varepsilon_{k+1}$ konvergují k nule (pro $k \to \infty$) zhruba dvakrát rychleji než ε_k, bude metoda Razleighova podílu dávat zpravidla lepší aproximace vlastního čísla. Vlastní vektor $\mathbf{v}_1$ se určí stejně jako v odst. 10.3.1.

10.3.3 Příklad.

Stanovme dominantní vlastní číslo matice

$$\mathbf{A} = \begin{pmatrix} 4 & 1 & 0 \\ 1 & 2 & 1 \\ 0 & 1 & 1 \end{pmatrix}.$$

Volme $\mathbf{y}^{(0)} = (1, 1, 1)^T$. Sestrojíme posloupnost vektorů $\mathbf{y}^{(k)}$ podle vztahu

(10.3.1):

$$\begin{aligned}
\mathbf{y}^{(1)} &= \mathbf{A}\mathbf{y}^{(0)} = (5,4,2)^T,\\
\mathbf{y}^{(2)} &= \mathbf{A}\mathbf{y}^{(1)} = (24,15,6)^T,\\
\mathbf{y}^{(3)} &= \mathbf{A}\mathbf{y}^{(2)} = (111,60,21)^T,\\
&\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\\
\mathbf{y}^{(7)} &= \mathbf{A}\mathbf{y}^{(6)} = (45\,423, 21\,141, 6\,201)^T,\\
\mathbf{y}^{(8)} &= \mathbf{A}\mathbf{y}^{(7)} = (202\,833, 93\,906, 27\,342)^T,\\
&\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots
\end{aligned}$$

Protože matice $\mathbf{A}$ je symetrická, můžeme užít vztahu (10.3.5) (Rayleighův podíl):

$$\lambda_1 \approx \frac{\mathbf{y}^{(7)^T}\mathbf{y}^{(7)}}{\mathbf{y}^{(7)^T}\mathbf{y}^{(7)}} = 4,459\,656\,908\ldots .$$

Vlastní vektor $\mathbf{v}_1$ určíme z přibližného vztahu

$$\mathbf{v}_1 \approx \frac{\mathbf{y}^{(8)}}{\|\mathbf{y}^{(8)}\|_{\mathbf{E}}} = (0,900\,75; 0,417\,02; 0,121\,42)^T; \quad \|\mathbf{v}_1\|_{\mathbf{E}} = 1.$$

Pro srovnání proved'me výpočet čísla λ_1 také ze vztahů (10.3.4):

$$\frac{y_1^{(8)}}{y_1^{(7)}} = \frac{202\,833}{45\,423} = 4,465\,425\,004; \quad \frac{y_2^{(8)}}{y_2^{(7)}} = 4,441\,890\,166;$$

$$\frac{y_3^{(8)}}{y_3^{(7)}} = 4,409\,288\,824.$$

Aritmetická průměr těchto čísel $4,438\,867\,996$ můžeme vzít za aproximaci čísla λ_i. Dá se ukázat, že tato aproximace je horší než aproximace získaná pomocí Razleighova podílu. Zkušenosti však ukazují, že poměrnejvětších složek vektorů $\mathbf{y}^{(k+1)}$ obvykle dává nejlepší aproximaci λ_1. [Přesná hodnota $\lambda_1 = 4,460\,504\,864$ - v $M(10,10)$.]

10.4 Úplný problém vlastních čísel.

Metody řešení úplného problému vlastních čísel můžeme zhruba rozdělit do dvou kategorií:

a) metody založené na výpočtu vlastních čísel pomocí charakteristického polynomu,

b) metody využívající vztahu podobnosti matic.
V prvém případě ovšem nelze (pro větší n) počítat $p_{\mathbf{A}}(\lambda) = \det(\mathbf{A} - \lambda\mathbf{I})$ přímo z definice determinantu. Existují však metody (např. Krylovova, Le Verrierova), které určují iteračním způsobem koeficienty charakteristického polynomu. Efektivnost těchto metod však není velká, nebot' např. Krylovova metody vyžaduje řádově $\frac{4}{3}n^3$ operací. Vyhodnější je situace u tridiagonálních matic, kde lze odvodit jednoduchou rekurenci pro výpočet charakteristického polynomu:

$$
\begin{aligned}
&f_{-1}(\lambda) = 0; \quad f_0(\lambda) = 1;\\
&f_k(\lambda) = (a_k - \lambda)f_{k-1}(\lambda) - b_k c_{k-1} k_{k-2}(\lambda), \quad k = 1, 2, ..., n;\\
&f_n(\lambda) = p_{\mathbf{A}}(\lambda).
\end{aligned}
$$

Zde $a_1, a_2, ..., a_n$ jsou prvky hlavní diagonály, $c_1, c_2, ..., c_{n-1}$ jsou prvky nad hlavní diagonálou a $b_2, b_3, ..., b_n$ prvky pod hlavní diagonálou. Pro symetrické třřídiagonální matice je tato metoda podrobně popsána ve [12] v souvislosti s tzv. Givensovou metodou. Protože právě Givensova metoda a dále ještě efektivnější householderova metoda převádějí libovolnou symetrickou matici na třídiagonální tvar, lze naznačenou metodou určit vlastní čísla dosti široké třídy matic. Pro nesymetrickou matici lze převod na třídiagonální tvar realizovat např. tzv. Lanczosovou metodou. Podrobný výklad všech zmíněných postupů včetně numerických aspektů najde čtenář ve [12]. Druhá kategorie metod využívá kaftu, že podrobné matice mají stejná vlastní čísla. Vyložíme zde dvě metody této kategorie. V obou se konstruuje posloupnost navzájem podobných matic, která konverguje k takové matici, jejíž vlastní čísla se dají jednoduchým způsobem určit.

10.4.1 Metoda LU-rozkladu.

Tato metoda je v literatuře obvykle nazývána LR-transformací nebo LR-algoritmem.

Je-li $\mathbf{LU} = \mathbf{A}$ trojúhelníkový rozklad matice $\mathbf{A}$ ve smyslu odst. 6.2.4, tj. s jednotkami na diagonále matcie $\mathbf{L}$, potom matice $\mathbf{B} = \mathbf{UL}$ je podobná matici $\mathbf{A}$ (má tatáž vlastní čísla). Skutečně, $\mathbf{U} = \mathbf{L}^{-1}\mathbf{A}$, a proto $\mathbf{B} = \mathbf{UL} = \mathbf{L}^{-1}\mathbf{AL}$.

Sestrojíme posloupnost matic $\mathbf{A}_k$ následujícím způsobem: Necht' $\mathbf{A} = \mathbf{A}_0 = \mathbf{L}_)\mathbf{U}_0$ je Lu-rozklad matice $\mathbf{A}$. Stanovíme matici

$$\mathbf{A}_1 = \mathbf{U}_0\mathbf{L}_0,$$

která vznikne vynásobením matic $\mathbf{L}_0$, $\mathbf{U}_0$ v opačném pořadí a opět ji rozložíme na součin trojúhelníkových matic, tj.

$$\mathbf{A}_1 = \mathbf{L}_1\mathbf{U}_1.$$

Pokračujeme v tomto procesu a dostáváme posloupnost navzájem podobných matic

$$\mathbf{A}_k = \mathbf{U}_{k-1}\mathbf{L}_{k-1} = \mathbf{L}_k\mathbf{U}_k, \quad k = 1, 2, \ldots . \tag{10.4.1}$$

Dá se dokázat ([12]), že když matice $\mathbf{B}_k = \mathbf{L}_1\mathbf{L}_2 \ldots \mathbf{L}_k$ konvergují k regulární matici, potom matice $\mathbf{A}_k$ také konvergují, a to k horní trojúhelníkové matici s vlastními čísly matice $\mathbf{A}$ na diagonále.

Je-li matice $\mathbf{A}$ symetrická a pozitivně definitní a provádíme-li LU-rozklad ve smyslu Choleského algoritmu, tj.

$$\mathbf{A}_k = \mathbf{L}_{k-1}^T\mathbf{L}_{k-1} = \mathbf{L}_k\mathbf{L}_k^T, \quad k = 1, 2, \ldots . \tag{10.4.2}$$

potom matice $\mathbf{A}_k$ konvergují k diagonální matici.

Nevýhody metody Lu-rozkladu jsou patrně trojího typu: (i) obecně pomálá konvergence posoupnosti $\{\mathbf{A}_k\}$, (ii) velký počet operací potřebných k realizaci algoritmu pro matice vyšších řádů, (iii) možnost nestability procesu pro obecnou matici $\mathbf{A}$.

Počet operací bude v únosných mezích, pokud tuto metodu aplikujjeme na pásovou matici. Potom matice $\mathbf{A}_k$ budou též pásové, např. třídiagonální.

Zvlaště jednoduchá je situace v případě třídiagonální symetrické pozitivně definitní matice $\mathbf{A}$. Potom matice $\mathbf{A}_k$ v (10.4.2) je opět třídiagonální a můžeme odvodit jednoduchý (stabilní) algoritmus výpočtu prvků matice $\mathbf{A}_{k-1}$ pomocí prvků matice $\mathbf{A}_k$: Předpokládáme, že je dán rozklad

$$\mathbf{A}_k = \mathbf{L}_k\mathbf{L}_k^T.$$

Nenulové prvky matice $\mathbf{L}_k$ označíme $a_{ii}, a_{i+1,i}$, $i = 1, 2, ..., n$. Nenulové prvky matice $\mathbf{L}_{k+1}$, které určuje vztah

$$\mathbf{A}_{k+1} = \mathbf{L}_{k+1}\mathbf{L}_{k+1}^T,$$

označíme $b_{ii}, b_{i+1,i}$, $i = 1, 2, ..., n$, a vypočteme je algoritmem:

$$\begin{aligned}
&\text{Vstup}: \quad n, a_{ii}, a_{i+1,i} \quad (i = 1, 2, ..., n).\\
&\qquad a_{n+1,i} = b_{0,i} = 0.\\
&\qquad \text{Pro} \quad i = 1, 2, ..., n\\
&\qquad\qquad b_{ii}^2 = a_{ii}^2 + a_{i+1,i}^2 - b_{i-1,i}^2,\\
&\qquad\qquad b_{i+1,i}^2 = \frac{a_{i+1,i}^2 a_{i+1,i+1}^2}{b_{ii}^2}.\\
&\text{Výstup}: \quad b_{ii}, b_{i+1,i} \quad (i = 1, 2, ..., n).
\end{aligned} \tag{10.4.3}$$

Tento algoritmus jej jistou modifikací tzv. QD-algoritmu uvedeného ve [12].

10.4.2 Příklad.

Ilustrujme si algoritmus (10.4.3) na úloze stanovot všechna vlastní čísla symetrické pozitivně definitní matice

$$\mathbf{A} = \begin{pmatrix} 2 & -1 & 0 \\ -1 & 2 & -1 \\ 0 & -1 & 1 \end{pmatrix}.$$

Prvky matice $\mathbf{A} \equiv \mathbf{A}_0$ označíme c_{ij}, prvky matice $\mathbf{L}_0$ v rozkladu $\mathbf{A}_0 = \mathbf{L}_0\mathbf{L}_0^T$ označíme a_{ij} a vypočteme je Choleského algoritmem, tj. $a_{11} = \sqrt{c_{11}}$, $a_{21} = c_{21}/a_{11}$, $a_{22} = \sqrt{c_{22} - a_{21}^2}$, $a_{32} = c_{32}/a_{22}$, $a_{33} = \sqrt{c_{33} - a_{32}^2}$. Máme tedy

$$\mathbf{L}_0 = \begin{pmatrix} a_{11} & 0 & 0 \\ a_{21} & a_{22} & 0 \\ 0 & a_{32} & a_{33} \end{pmatrix}.$$

Prvky matice

$$\mathbf{L}_1 = \begin{pmatrix} b_{11} & 0 & 0 \\ b_{21} & b_{22} & 0 \\ 0 & b_{32} & b_{33} \end{pmatrix}$$

vypočteme algoritmem (10.4.3) přímo pomocí prvků matice $\mathbf{L}_0$, tj. pomocí vztahů $b_{11} = \sqrt{a_{11}^2 + a_{21}^2}$, $b_{21} = a_{21}a_{22}/b_{11}$, $b_{22} = \sqrt{a_{22}^2 + a_{32}^2 - b_{21}^2}$, $b_{32} = = a_{32}a_{33}/b_{22}$, $b_{33} = \sqrt{a_{33}^2 - b_{32}^2}$. Dále vypočteme $\mathbf{L}_1\mathbf{L}_1^T = \mathbf{A}_1$. Přeznačíme-li b_{ij} na a_{ij}, stanovíme prvky b_{ij} matice $\mathbf{L}_2$ opět algoritmem (10.4.3). V každé fázi výpočtu počítáme $\mathbf{L}_k\mathbf{L}_k^T = \mathbf{A}_k$. Proces zastavíme například požadavkem, aby největší (v absolutní hodnotě) nediagonální prvek matice $\mathbf{A}_k$ byl menší než nějaké předem zvolené číslo δ (lze ovšem volit i jiná zastavovací kretéria).

Níže jsme zazanamenali pouze sled výpočtů. Pro srovnání uved'me, že kořeny polynomu $\det(\mathbf{A} - \lambda\mathbf{I}) = -\lambda^3 + 5\lambda^2 - 6\lambda + 1$ jsou (s chybou menší než 10^{-4}) $\lambda_1 \doteq 3,247\,0$; $\lambda_2 \doteq 1,550\,0$, $\lambda_3 \doteq 0,198\,1$.

$$\lambda_1 \approx 3,217\,7; \quad \lambda_2 \approx 1,595\,9; \quad \lambda_3 \approx 0,199\,7.$$

Jestliže dostatečně velké k je $\mathbf{A}_k \approx \Lambda$ (diagonální matice), potom vlastní vektory jsou (opět přibližné) sloupce matice $\mathbf{L}_0\mathbf{L}_1\mathbf{L}_2...\mathbf{L}_k$.

10.5 Metody ortogonálních transformací.

Budeme sestrojovat posloupnost $\mathbf{A}_0\mathbf{A}_1\mathbf{A}_2, ...$ navzájem podobných matic, tak že

$$\mathbf{A}_{k+1} = \mathbf{Q}_k^T \mathbf{A}_k \mathbf{Q}_k, \quad k = 0, 1, 2, ...,$$

která konverguje k matici, jejíž vlastní čísla se dají lehce určit. Ortogonální matice $\mathbf{Q}_k$ speciálním způsobem vybíráme. Předností metod ortogonálních transformací je numerická stabilita příslušných algoritmů. Pro symetrickou matici $\mathbf{A} \equiv \mathbf{A}_0$ vede tento postup k tzv. *Jacobiově metodě*, kterou si v dalším výkladu vyložíme podrobněji. Pro obecnou matici používáme *metody QU-rozkladu*[14]). == V této metodě v každém kroku rozkládáme matici $\mathbf{A}_k$ na součin $\mathbf{A}_k = \mathbf{Q}_k\mathbf{U}_k$ ortogonální matice $\mathbf{Q}_k$ a horní trojúhelníkové matice $\mathbf{U}_k$ (určení matic $\mathbf{Q}_k$, $\mathbf{U}_k$ se nazývá *QU-rozklad*) a určíme $\mathbf{A}_{k+1} = \mathbf{U}_k\mathbf{Q}_k$. Za dosti obecných předpokládů konvergují matice $\mathbf{A}_k$ k horní trojúhelníkové matici s vlastními čísly matice $\mathbf{A}$ na diagonále (viz [12]).

10.5.1 Příklad.

Příkladem ortogonální matice je matice rovinné rotace o úhel α:

$$\mathbf{Q}_{12}(\alpha) = \begin{pmatrix} c & -s \\ s & c \end{pmatrix}, \quad \begin{matrix} c = \cos\alpha, \\ s = \sin\alpha. \end{matrix}$$

Stanovte matici $\mathbf{B} = \mathbf{Q}_{12}^T(\alpha)\mathbf{A}\mathbf{Q}_{12}(\alpha)$, podobnou dané symetrické matici

$$\mathbf{A} = \begin{pmatrix} 2 & 1 \\ 1 & 3 \end{pmatrix}.$$

Jednoduchým výpočtem zjistíme, že

$$b_{11} = 2c^2 - 2cs + 3s^2, \quad b_{12} = c^2 - s^2 + cs = b_{21}, \quad b_{22} = 2s^2 + 3c^2 - 2cs.$$

Vybereme α tak, aby bylo $b_{12} = \cos^2\alpha = \sin^2\alpha + \sin 2\alpha = 0$. Z požadavku $2\alpha = -2$ určíme $c^2 = \cos^2\alpha = (5 - \sqrt{5})/10$, $s^2 = \sin^2\alpha = (5 + \sqrt{5})/10$, $cs = \cos\alpha\sin\alpha = \sqrt{5}/5$, a tedy

$$\mathbf{B} = \begin{pmatrix} (5+\sqrt{5})/2 & 0 \\ 0 & (5-\sqrt{5})/2 \end{pmatrix}.$$

Diagonální prvky matice $\mathbf{B}$ jsou vlastní čísla matice $\mathbf{A}$.

[14] Užívá se často termínu QR-transformace. Podrobný vyklad najde čtenář v [12].

10.5.2 Jacobiova diagonalizace.

Budeme se opírat o následující tvrzení (viz [12]): *Je-li* $\mathbf{A}$ *reálná symetrická matice, potom existuje ortogonální matice* $\mathbf{Q}$ (*tj.* $\mathbf{Q}^T = \mathbf{Q}^{-1}$), *že matice* $\mathbf{Q}^T\mathbf{A}\mathbf{Q} = \Lambda$ *je diagonální s vlastními čísly matice* $\mathbf{A}$ *na diagonále.*

Matici $\mathbf{Q}$ budeme sestrojovat postupně pomocí posloupnosti ortogonálních matic $\mathbf{Q}_{pq}(\alpha)$, kde

$$\mathbf{Q}_{pq}(\alpha) = \begin{array}{c} \\ \\ \\ \\ \\ p \\ \\ \\ \\ \\ q \\ \\ \\ \\ \end{array} \begin{pmatrix} 1 & & & & \vdots & & & & \vdots & & & \\ & 1 & & & \vdots & & & & \vdots & & & \\ & & \ddots & & \vdots & & & & \vdots & & & \\ & & & 1 & \vdots & & & & \vdots & & & \\ \cdots & \cdots & \cdots & \cdots & s & \cdots & \cdots & \cdots & -s & \cdots & \cdots & \cdots \\ & & & & \vdots & 1 & & & \vdots & & & \\ & & & & \vdots & & \ddots & & \vdots & & & \\ & & & & \vdots & & & 1 & \vdots & & & \\ \cdots & \cdots & \cdots & \cdots & s & \cdots & \cdots & \cdots & c & \cdots & \cdots & \cdots \\ & & & & \vdots & & & & \vdots & 1 & & \\ & & & & \vdots & & & & \vdots & & \ddots & \\ & & & & p & & & & q & & & 1 \end{pmatrix};$$

$$c = \cos\alpha,$$

$$s = \sin\alpha,$$

$$\mathbf{Q}_{pq}^T(\alpha) = \mathbf{Q}_{pq}(-\alpha);$$

matice $\mathbf{Q}_{pq}(\alpha)$ se liší od jednotkové matice pouze v pozicích (p,p), (q,q), (p,q), (q,p). Vybereme úhel α tak, abychom nulovali prvek $b_{pq} = b_{qp}$ matice

$$\mathbf{B} = \mathbf{Q}_{pq}^T(\alpha)\mathbf{A}\mathbf{Q}_{pq}(\alpha)$$

Protože $b_{pq} = (a_{qq} - a_{pp})cs + a_{pq}(c^2 - s^2)$, vede požadavek $b_{pq} = 0$ k podmínce

$$\operatorname{tg} 2\alpha = \frac{2a_{pq}}{a_{pp} - a_{qq}}.$$

Odtud pak dostaneme

$$
\begin{aligned}
& c^2 = \cos^2\alpha = \frac{1}{2} + \frac{a_{pp} - a_{qq}}{2r}, \quad s^2 = \sin^2\alpha = \frac{1}{2} - \frac{a_{pp} - a_{qq}}{2r}, \\
& sc = \frac{a_{pq}}{r}, \quad r^2 = (a_{pp} - a_{qq})^2 + 4a_{pq}^2.
\end{aligned} \tag{10.5.1}
$$

Při konkrétním výpočtu není třeba počítat úhel α, ale pomocí vzorců (10.5.1) odvodit formule pro přímý výpočet prvků matice $\mathbf{B}$ (volíme $|\alpha| \leqq \pi/4$):

$$
\begin{aligned}
& b_{pp} = (a_{pp} + a_{qq} + r)/2, \\
& b_{qq} = (a_{pp}a_{qq} - a_{pq}^2)/b_{pp}, \\
& b_{ip} = a_{ip}c + a_{iq}s = b_{pi}, \\
& b_{iq} = -a_{ip}s + a_{iq}c = b_{qi}, \\
& b_{ij} = a_{ij} \quad \text{v pstatních pozicích.}
\end{aligned} \tag{10.5.2}
$$

Přeznačením prvků b_{ij} na a_{ij} můžeme v popsánem procesu polračovat, nyní ovšem s jinými indexy p, q. Nulové prvky z předcházejících fází však obecně nulové nezůstanou, ale dá se ukázat, že u takto sestrojené posloupnosti matic $\mathbf{A} = \mathbf{A}_0, \mathbf{A}_1, \mathbf{A}_2, \ldots$ se mimodiagonální prvky zmenšují. Přesněji - součet kvadrátů mimodiagonálních prvků konverguje k nule (viz [12]), čehož využíváme k zastavení uvedeného iteračního procesu.

Volbu znaménka u $\sin\alpha$, $\cos\alpha$ počítaných z (10.5.1)provádíme tak, aby sign $\cos\alpha = \operatorname{sign} a_{pq}$.

Zbývá zodpovědět otázku jak volit strategii při výběru indexů p, q (viz [12]). Bud' provádíme výběr tak, abychom nulovali vždy největší (v absolutní hodnotě) nediagonální prvek, nebo nulujeme postupně. Druhý způsob se snáze algoritmuje, ale dává obvykle pomalejší konvergenci matic $\mathbf{A}_k$ k diagonální matici (rychlost konvergence závisí - stejně jako u mocninné metody - na λ_i/λ_{i-1}).

Vlastní vektory matice $\mathbf{A}$ jsou sloupce matice $\mathbf{Q}$, kterou stanovíme jako součin matic $\mathbf{Q}_{pq}$ v tom pořadí, jaké bylo zvolenou strategií určeno.

10.5.3 Příklad.

Užitím vztahů (10.5.1), (10.5.2) realizujeme Jacobiovu diagonalizaci k dané symetrické matici $\mathbf{A} = \mathbf{A}_0$. Indexy p, q volíme strategií podle prvku s největší absolutní hodnotou. Zaznamenáme sled výsledků:

$$
\mathbf{A}_0 = \begin{pmatrix} 3,5 & -6 & 5 \\ -6 & 8,5 & -9 \\ 5 & -9 & 8,5 \end{pmatrix} \overset{(p,q)=(2,3)}{\rightarrow} \begin{pmatrix} 3,5 & -7,778\,2 & -0,707\,1 \\ -7,778\,2 & 17,5 & 0 \\ -0,707\,1 & 0 & -0,5 \end{pmatrix} \overset{(1,2)}{\rightarrow}
$$

$$\overset{(1,2)}{\to} \begin{pmatrix} 20,964\,3 & 0 & 0,287\,7 \\ 0 & 0,035\,8 & -0,645\,9 \\ 0,287\,7 & -0,645\,9 & -0,5 \end{pmatrix} \overset{(2,3)}{\to}$$

$$\overset{(2,3)}{\to} \begin{pmatrix} 20,964\,3 & -0,159\,8 & 0,239\,2 \\ -0,159\,8 & 0,467\,2 & 0 \\ 0,239\,2 & 0 & -0,931\,4 \end{pmatrix} \overset{(1,3)}{\to}$$

$$\overset{(1,3)}{\to} \begin{pmatrix} 20,968\,1 & 0,000\,0 & 0 \\ 0,000\,0 & 0,465\,9 & 0,001\,7 \\ 0 & 0,001\,7 & -0,934\,0 \end{pmatrix} \overset{(1,2)}{\to}$$

$$\overset{(1,2)}{\to} \begin{pmatrix} 20,968\,1 & 0 & 0,000\,0 \\ 0 & 0,465\,9 & 0,001\,7 \\ 0,000\,0 & 0,001\,7 & -0,934\,0 \end{pmatrix} \overset{(2,3)}{\to}$$

$$\overset{(2,3)}{\to} \begin{pmatrix} 20,968\,1 & 0,000\,0 & 0,000\,0 \\ 0,000\,0 & 0,465\,9 & 0 \\ 0,000\,0 & 0 & -0,934\,0 \end{pmatrix};$$

$$\lambda_1 \approx 20,968\,1,$$
$$\lambda_2 \approx 0,465\,9,$$
$$\lambda_3 \approx -0,934\,0.$$

10.5.4 Poznámka.

Podotkněme, že podobnostní metody se většinou používají k maticím v tridiagonálním nebo v tzv. Hessenbergově tvaru. Aby se zmenšil celkový počet operací, obvykle se nejdříve obecná matice převede (metodami uvedenými např. v [12]) na některý ze zmíněných typů a pak se aplikuje některá metoda podrobnostních transformací (Jacobiova diagonalizace nebo QR-rozklad, resp. LU-rozklad). Dáváme přednost metodě ortogonálních transformací, nebot' je numericky stabilní.

Poznamenejme na závěr, že velmi efektivní algoritmy těchto metod najde čtenář v knize [14].

10.6 Cvičení.

10.6.1

Stanovte vlastní čísla a vlastní vektory matice

$$\mathbf{A} = \begin{pmatrix} 2 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 2 \end{pmatrix}.$$

[Návod. Užijte postupu z příkl. 10.1. $\lambda_1 = \lambda_2 = 1$, $\mathbf{v}^{(1)} = (1, 0, -1)^T$, $\mathbf{v}^{(2)} = (0, 1, -1)^T$; $\lambda_3 = 4$, $\mathbf{v}^{(3)} = (1, 1, 1)^T$.]

10.6.2

Stanovte vlastní čísla a vlastní vektory matice

$$\begin{pmatrix} 2 & 1+i \\ 1-i & 2 \end{pmatrix}.$$

[$\lambda_1 = 2+\sqrt{2}$, $\mathbf{v}_1 = (\sqrt{2}, 1)^T + i(0, -1)^T$; $\lambda_2 = 2-\sqrt{2}$, $\mathbf{v}_1 = (-\sqrt{2}, 1)^T + i(0, -1)^T$.]

10.6.3

Stanovte vlastní čísla matic

$$\text{a)} \begin{pmatrix} 6 & 4 & 4 & 1 \\ 4 & 6 & 1 & 4 \\ 4 & 1 & 6 & 4 \\ 1 & 4 & 6 & 4 \end{pmatrix}; \quad \text{b)} \begin{pmatrix} 5 & 4 & 1 & 1 \\ 4 & 5 & 1 & 1 \\ 1 & 1 & 4 & 2 \\ 1 & 1 & 2 & 4 \end{pmatrix}.$$

[a) $\lambda_1 = 15$, $\lambda_2 = 5$, $\lambda_3 = 5$, $\lambda_4 = -1$; $\mathbf{v}_1 = (1, 1, 1, 1)^T$, $\mathbf{v}_2 = (0, 1, -1, 0)^T$, $\mathbf{v}_3 = (1, 0, 0, -1)^T$, $\mathbf{v}_4 = (1, -1, -1, 1)^T$;
b) $\lambda_1 = 10$, $\lambda_2 = 5$, $\lambda_3 = 2$, $\lambda_4 = 1$; $\mathbf{v}_1 = (1, 1, \frac{1}{2}, \frac{1}{2})^T$, $\mathbf{v}_2 = (-\frac{1}{2}, -\frac{1}{2}, 1, 1)^T$, $\mathbf{v}_3 = (0, 0, 1, -1)^T$, $\mathbf{v}_4 = (1, -1, 0, 0)^T$.]

11 Zobecněná řešení soustav lineárních rovnic

V současné numerické analýze hraje důležitou roli pojem *singulárního rozkladu matice*. Pokusíme se dát čtenáři několik základních informací o tomto pojmu.

11.1 Singulární rozklad matice.

K reálné matici $\mathbf{A}$ *typu* (m,n) *existují takové ortogonální matice* $\mathbf{U}$ *a* $\mathbf{V}$ [$\mathbf{U}$ *je typu* (m,m), $\mathbf{V}$ *je typu* (n,n)], *že prvky* σ_{ij} *matice*

$$\mathbf{\Sigma} = \mathbf{U}^T\mathbf{A}\mathbf{V} \tag{11.1.1}$$

[*typu* (m,n)] *mají tuto vlastnost*:
a) $\sigma_{ij} = 0$, když $i \neq j$,
b) $\sigma_{ij} \equiv \sigma_i \geqq 0$.
Platí tedy

$$\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T.^{15}) \tag{11.1.2}$$

Číslům σ_i říkáme *singulární čísla* matice $\mathbf{A}$ a sloupcům matic $\mathbf{U}$, $\mathbf{V}$ *levé a pravé sigulární vektory. Singulárním rozkladem matice* $\mathbf{A}$ rozumíme stanovení matic $\mathbf{U}$, $\mathbf{\Sigma}$, $\mathbf{V}$. Dá se ukázat, že matice $\mathbf{A}\mathbf{A}^T$ [typu (m,m)] a matice $\mathbf{A}^T\mathbf{A}$ [typu (n,n)] mají n stejných vlastních čísel (předpokládali jsme, že $n \leqq m$) a nenulová singulární čísla matice $\mathbf{A}$ jsou odmocniny z těchto vlastních čísel. K odvození efektivných a stabilních algoritmů singulárního rozkladu však tohoto faktu nelze užít. Jejich odvození přesahuje rámec této publikace; jsou popsány v [7a](ve Fortranu) a v [14](v Algolu). Singulární čísla matice $\mathbf{A}$ jsou mimořádně málo citlivá na poruchy v prvcích matice, což se nedá říci o vlastních číslech nesymetrické matice (viz odst. 10.2.7).

11.1.1 Příklad.

Mějme dánu matici $\mathbf{A}$. Její singulární rozklad je dán maticemi $\mathbf{U}$, $\mathbf{\Sigma}$, $\mathbf{V}^T$:

$$\mathbf{A} = \begin{pmatrix} 1 & 6 & 11 \\ 2 & 7 & 12 \\ 3 & 8 & 13 \\ 4 & 9 & 14 \\ 5 & 10 & 15 \end{pmatrix},$$

[15]) Omezujeme se na reálné matice, ovšem celý váklad lze provést i pro komplexní matice. Vztah (11.1.2) pak má tvar $\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^H$, kde $\mathbf{U}$, $\mathbf{V}$ jsou unitární matice, tj. takové, že $\mathbf{U}^H = \mathbf{U}^{-1}$, $\mathbf{V}^H = \mathbf{V}^{-1}$ (viz odst. 5.1.2).

$$\mathbf{U} = \begin{pmatrix} 0,3555 & -0,689 & 0,541 & 0,193 & 0,265 \\ 0,399 & -0,376 & -0,802 & -0,113 & 0,210 \\ 0,443 & -0,062 & 0,160 & -0,587 & -0,656 \\ 0,487 & 0,251 & -0,079 & 0,742 & -0,378 \\ 0,531 & 0,564 & 0,180 & -0,235 & 0,559 \end{pmatrix},$$

$$\mathbf{\Sigma} = \begin{pmatrix} 35,127 & 0 & 0 \\ 0 & 2,465 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \quad \mathbf{U} = \begin{pmatrix} 0,202 & 0,890 & 0,408 \\ 0,517 & 0,257 & -0,816 \\ 0,832 & -0,376 & 0,408 \end{pmatrix}.$$

Singulární čísla matice $\mathbf{A}$ jsou: $\sigma_1 = 35,127$, $\sigma_2 = 2,465$, $\sigma_3 = 0$. Prvky matice $\mathbf{A} - \mathbf{U\Sigma V}^T$ jsou zde 10^{-2}.

11.2 Hodnost matice a číslo podmíněnosti.

Nemožnost numericky určit hodnost atice vedla k použití singulárního rozkladu. Víme z lineární algebry, že matici $\mathbf{A}$ typu (n,n) o hodnosti $h(\mathbf{A}) = r$ lze eliminační metodou s úplným výběrem upravit na takový trojúhelníkový tvar $\mathbf{U}$, že v posledních $n-r$ řádcích budou nuly. Numericky bychom mohli uvažovat podobně: Bude-li $\mathbf{A}$ blizká matici o hodnosti r, měla by mít posledních $n-r$ řádků malých (v normě). To však není pravda, jak ihned uvidíme.

Matice

$$\mathbf{A} = \begin{pmatrix} 1 & -1 & \cdots & \cdots & \cdots & -1 \\ 0 & 1 & -1 & \cdots & \cdots & -1 \\ 0 & 0 & 1 & -1 & \cdots & -1 \\ \vdots & & \ddots & \ddots & \ddots & \vdots \\ \vdots & & & \ddots & \ddots & -1 \\ 0 & \cdots & \cdots & \cdots & \cdots & 1 \end{pmatrix}$$

má hodnost $h(\mathbf{A}_n) = n$. Přidáme-li do pozice $(n,1)$ (levý dolní roh) prvek -2^{2-n}, dostaneme matici $\mathbf{A}_n + \mathbf{Z}$, $\|\mathbf{Z}\|_{\mathbf{F}} = 2^{2-n}$ [$\mathbf{Z}$ se liší od nulové matice pouze prvkem -2^{2-n} v pozici $(n,1)$], pro niž $h(\mathbf{A}_n + \mathbf{Z}) = n-1$, přičemž nelze ani jeden řádek matice $\mathbf{A}_n + \mathbf{Z}$ považovat za malý.

Proto k určení hodnosti obecné matice $\mathbf{A}$ užijeme singulárního rozkladu, kdy $h(\mathbf{A}) = h(\Sigma)$; hodnost je rovna počtu nenulových singulárních čísel matice $\mathbf{A}$. Tento postup je založen na faktu, že ortogonální transformace (tj. násobení ortogonální maticí) nemění lineární nezávislost vektor. Výhody singulárního rozkladu spočívají v tom, že rozhodujeme o jednotlivých číslech,

nikoliv o vektorech či souborech vektorů. V numerické praxi užíváme termínu *efektivní hodnost*; je to počet singulárních čísel větších než nějaké předem zadané (malé) číslo ε.

Pomocí singulárních čísel lze dát jinou definici *čísla podmíněnosti matice*:

$$C_{\mathbf{A}} = \frac{\sigma_{max}}{\sigma_{min}}.$$

Je-li $\sigma_{min} = 0$, řekneme, že číslo podmíněnosti matice $\mathbf{A}$ je nekonečně velké. Matice $\mathbf{A}$ v tomto případě nemá plnou hodnost, tj. $h(\mathbf{A}) < n$.

Bude-li $C_{\mathbf{A}} \approx 1$, budeme sloupce matice $\mathbf{A}$ považovat za *velmi nezávislé*. Pro $C_{\mathbf{A}}$ velké budou sloupce $\mathbf{A}$ *skoro závislé*. Příkladem velmi nezávislých vektorů jsou vektory ortogonální. Pro ortogonální matici $\mathbf{A}$ je $C_{\mathbf{A}} = 1$. Proto se ortogonálních matic užívá při konstrukci stabilních algoritmů.

11.3 Řešení obecných soustav lineárních rovnic.

Ukážeme si postup, jak pomocí singulárního rozkladu řešit soustavu rovnic s obecně obdélníkovou maticí, přičemž připoustíme i singulární čtvercové matice.

Necht' matice $\mathbf{A}$ je typu (m, n). Chceme najít všechny n-rozměrné vektory $\mathbf{x}$, pro které

$$\mathbf{Ax} = \mathbf{b}, \tag{11.3.1}$$

kde $\mathbf{b}$ je m-rozměrný vektor [matice typu $(m, 1)$].

Stanovíme singulární rozklad matice $\mathbf{A}$ a v soustavě

$$\mathbf{U\Sigma V}^T\mathbf{x} = \mathbf{b} \tag{11.3.1}$$

Označíme $\mathbf{V}^T\mathbf{x} = \mathbf{z}$, $\mathbf{d} = \mathbf{U}^T\mathbf{b}$. Potom místo soustavy (11.3.1) řešíme soustavu

$$\mathbf{\Sigma z} = \mathbf{d} \tag{11.3.2}$$

V rozepsané podobě

$$\begin{aligned} \sigma_j z_j &= d_j, \quad j \leqq n, \\ 0 &= d_j, \quad j > n. \end{aligned} \tag{11.3.3}$$

Tato soustava má jediné řešení $\mathbf{z} = (z_1, z_2, ..., z_n)^T$, pokud $h(\mathbf{A}) \equiv r = n$, a tedy $\sigma_j \neq 0$, $j = 1, 2, ..., n$, a když $d_{n+1} = d_{n+2} = ... = d_m = 0$.

Soustava má $n-r$ parametrický systém řešení, když $r < n$ a $d_{r+1} = d_{r+2} = \\ = ... = d_m = 0$. V tomto případě totiž máme r rovnic $\sigma_j z_j = d_j$ kde $\sigma_j \neq 0$ a $n-r$ rovnic typu $0 \cdot z_j = d_j$, $r < j \leqq n$.

Když nebude splněná nulovost pravých stran d_j v uvažovaných situacích, nebude mít soustava řešení.

Řešení původní soustavy je dáno vztahem

$$\mathbf{x} = \mathbf{V}\mathbf{z}.$$

11.3.1 Příklad.

Uvažujme soustavu $\mathbf{A}\mathbf{x} = \mathbf{b}$, kde $\mathbf{b} = (5,5,5,5,5)^T$ a matice $\mathbf{A}$ je dána v příkl. 11.1.1 spolu s jejím singulárním rozkladem $\mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$.

Určíme $\mathbf{d} = \mathbf{U}^T\mathbf{b}$. Protože singulární čísla matice $\mathbf{A}$ jsou $\sigma_1 = 35,127$, $\sigma_2 = 2,465$, $\sigma_3 = 0$ má soustava (11.3.2) tvar

$$\begin{aligned} 35,127 \cdot z_1 &= 11,075, \\ 2,465 \cdot z_2 &= -1,560 \\ 0 \cdot z_3 &= 0. \end{aligned}$$

Tato soustava je řešitelná pro libovolné z_3, tj. $\mathbf{z} = (0,315; -0,633; z_3)^T$. Pro $z_3 = 0$ vypočteme $\mathbf{x}_0 = \mathbf{V}\mathbf{z}_0 = (-0,500; 0; 0,500)^T$. Dosazováním se můžeme přesvědčit, že vektory $\mathbf{x} = (-1,001; 1,002; -0,001)^T$ (pro $z_3 = \\ = -1,226$), $\mathbf{x} = (-1,1,0)$ jsou také řešením.

Abychom postihli všechna řešení, napíšeme $\mathbf{z}$ ve tvaru

$$\mathbf{z} = (0,315; -0,633; z_3)^T = (0,315; -0,633; 0)^T + z_3(0,0,1)^T = \mathbf{z}_0 + z_3\tilde{\mathbf{z}}.$$

Odtud

$$\mathbf{x} = \mathbf{V}\mathbf{z} = \mathbf{V}\mathbf{z}_0 + \mathbf{z}_3\mathbf{V}\tilde{\mathbf{z}} = (-\frac{1}{2}, 0, \frac{1}{2})^T + \mathbf{z}_3(0,408; -0,816; 0,408)^T =$$

$$= \frac{1}{2}(-1,0,1)^T + \alpha(1,-2,1)^T$$

(položili jsme $0,408 z_3 = \alpha$) Volíme-li $\mathbf{b} = (4,5,5,5,5)^T$, bude

$$\mathbf{d} = \mathbf{U}^T\mathbf{b} = (10,720; -0,871; -0,541; -0,193; -0,265)^T.$$

Protože poslední tři složky vektoru $\mathbf{d}$ nejsou nulové, soustava s touto pravou stranou není řešitelná[16]).

[16]) Vektor $\mathbf{b}$ nepatří do prostoru vektorů $\mathbf{y} = \mathbf{A}\mathbf{x}$.

11.4 Neřešitelné soustavy.

Necht' matice $\mathbf{A}$ je typu (m,n), $m \geq n$, a $\mathbf{b}$ je m-rozměrný vektor. Chceme najit n-rozměrný vektor $\mathbf{x}$ takový, aby vektor $\mathbf{Ax}$ byl co "nejblíže" k vektoru $\mathbf{b}$.

Soustava $\mathbf{Ax} = \mathbf{b}$ obecně nemusí být splněna "přesně". Řešením uvedené úlohy budeme rozumět takový vektor $\mathbf{x}_p$, pro který je reziduový vektor $\mathbf{r} = \mathbf{b} - \mathbf{Ax}$ minimální v euklidovské normě, tj. pro libovolný n-rozměrny vektor $\mathbf{z}$ platí

$$\|\mathbf{b} - \mathbf{Ax}_p\|_{\mathbf{E}} \leq \|\mathbf{b} - \mathbf{Az}\|_{\mathbf{E}}. \tag{11.4.1}$$

Takto definovanému řešení se říká *zobecněné řešení* nebo *řešení ve smyslu metody nejmenších čtverců.*

11.4.1 Příklad.

Stanovme zobecněné řešení soustavy $\mathbf{Ax} = \mathbf{b}$, kde

$$\mathbf{A} = \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix}, \quad \mathbf{b} = \begin{pmatrix} 1 \\ 2 \end{pmatrix}, \quad \mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}.$$

Stanovíme minimum funkce

$$f(x_1, x_2) = \|\mathbf{r}\|_{\mathbf{E}}^2 = \|\mathbf{b} - \mathbf{Ax}\|_{\mathbf{E}}^2 = (1 - x_1 - x_2)^2 + (2 - x_1 - x_2)^2.$$

Z podmínek $\partial f/\partial x_1 = 0$, $\partial f/\partial x_2 = 0$ dostaneme

$$-2(1 - x_1 - x_2) - 2(2 - x_1 - x_2) = 0,$$

tj.

$$x_1 + x_2 = \frac{3}{2}.$$

Zobecněním řešením dané soustavy je takový vektor $\mathbf{x}_p = (x_1, x_2)^T$, jehož složky jsou vázány podmínkou $x_1 + x_2 = \frac{3}{2}$. Proto

$$\mathbf{x}_p = \left(x_1, -x_1 + \frac{3}{2}\right)^T = \left(0, \frac{3}{2}\right)^T + x_1 (1, -1)^T; \quad \mathbf{Ax}_p = \left(\frac{3}{2}, \frac{3}{2}\right)^T.$$

Soustava má nekonečně mnoho zobecněných řešení $\mathbf{x}_p$, ale vektor $\mathbf{Ax}_p$ je jediný. Tento vektor je ortogonálním průmětem vektoru $\mathbf{b}$ do prostoru vektorů $\mathbf{Ax}$ a je tedy nejblíže k vektoru $\mathbf{b}$ ve smyslu euklidovské normy.

11.4.2 Normální rovnice.

Zobecněné řešení $\mathbf{x}_p$ definované vztahem (11.4.1) je řešením tzv. *normální soustavy* (viz[4])

$$(\mathbf{A}^T\mathbf{A})\mathbf{x} = \mathbf{A}^T\mathbf{b}. \tag{11.4.2}$$

Matice $\mathbf{A}^T\mathbf{A}$ je symetrická typu (n,n). Je-li matice $\mathbf{A}^T\mathbf{A}$ regulární, potom soustavu (11.4.2) lze řešit Choleského algoritmem. Dá se dokázat (viz opět [4]), že regularita matice $\mathbf{A}^T\mathbf{A}$ zaručuje jednoznačnost zobecněného řešení.

Normální rovnice pro soustavu z příkl. 11.4.1 je

$$\begin{pmatrix}2&2\\2&2\end{pmatrix}\begin{pmatrix}x_1\\x_2\end{pmatrix} = \begin{pmatrix}3\\3\end{pmatrix},\ \text{nebot'}\quad \mathbf{A}^T\mathbf{A} = \begin{pmatrix}2&2\\2&2\end{pmatrix},\quad \mathbf{A}^T\mathbf{b} = \begin{pmatrix}3\\3\end{pmatrix}$$

což vede k podmínce $x_1 + x_2 = \frac{3}{2}$, kterou jsme v uvedeném příkladu dostali minimalizováním kvadrstické formy $f(x_1,x_2) = \|\mathbf{r}\|^2_{\mathbf{E}}$.

Jinou metodou určení zobecněného řešení nám dá singulární rozklad matice $\mathbf{A}$. Protože $\mathbf{U}$ a $\mathbf{V}$ jsou ortogonální matice, lze psát (ortogonální transformace nemění euklidovskou normu vektoru)

$$\|\mathbf{r}\| = \|\mathbf{b}-\mathbf{A}\mathbf{x}\| = \left\|\mathbf{U}^T(\mathbf{b}-\mathbf{A}\mathbf{V}\mathbf{V}^T\mathbf{x})\right\| = \left\|\mathbf{U}^T\mathbf{b}-\mathbf{U}^T\mathbf{A}\mathbf{V}\mathbf{V}^T\mathbf{x}\right\| = \|\mathbf{d}-\mathbf{\Sigma}\mathbf{z}\|,$$

$$\mathbf{d} = \mathbf{U}^T,\quad \mathbf{z} = \mathbf{V}^T\mathbf{x}.$$

Odtud je patrné, že $\|\mathbf{r}\|$ bude minimální, když

$$z_j = \frac{d_j}{\sigma_j} \qquad \text{pro}\quad \sigma_j \neq 0, \tag{11.4.3}$$

$$z_j \quad \text{libovolné} \qquad \text{pro}\quad \sigma_j = 0.$$

Takže $r = h(\mathbf{A})$ rovnic v diagonálním tvaru lze řešit přesně. Ze zbývajících rovnic dostáváme, že $\|\mathbf{r}\| = \sum\limits_j d_j^2$, kde sčítáme přes všchna j, pro která $\sigma_j = 0$ nebo pro než $j > n$. Zpětnou substitucí $\mathbf{x} = \mathbf{V}\mathbf{z}$ dostaneme řešení původní úlohy.

Takto určené zobecněné řešení není ovšem jediné, pokud $h(\mathbf{A}) < n$. Z těchto všech řešení vybíráme nejčastěji to, která má

$$z_j = 0 \quad \text{pro}\quad \sigma_j = 0.$$

Důvodem, proč k určení zobecněného řešení užíváme pracného singulárního rozkladu, je skutečnost, že jde o numericky stabilní algoritmus pro větší n i v těch případech, kdy matice $\mathbf{A}$ nemá plnou hodnost, což právě v jednodušších (a rychlejších) algoritmech vede ke značným potížím.

11.4.3 Příklad.

V příkladu 11.3.1 jsme konstatovali, že soustava $\mathbf{Ax} = \mathbf{b}$, kde $\mathbf{b} = (4,5,5,5)^T$ a $\mathbf{A}$ je dána v příkl. 11.1.1 není řešitelná. Ukážeme si nyní, že existuje zobecněné řešení této soustavy.

Protože

$$\mathbf{\Sigma} = \begin{pmatrix} 35,127 & 0 & 0 \\ 0 & 2,465 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \quad \mathbf{d} = \mathbf{U}^T\mathbf{b} = \begin{pmatrix} 10,720 \\ -0,871 \\ -0,541 \\ -0,193 \\ -0,265 \end{pmatrix},$$

můžeme první dvě rovnice soustavy $\mathbf{\Sigma z} = \mathbf{d}$, tj.

$$\begin{aligned} 35,127 z_1 &= 10,720 \\ 2,465 z_2 &= -0,871, \end{aligned}$$

vyřešit přesně

$$\begin{aligned} z_1 &= 0,305 \\ z_2 &= -0,353. \end{aligned}$$

Neznámou z_3 můžeme podle (11.4.3) zvolit libovolně; proto položíme $z_3 = 0$, abychom dostali řešení $\mathbf{z}$ s nejmenší normou.

"Nepoužité" pravé strany určí $\|\mathbf{r}\|^2 = 0,541^2 + 0,193^2 + 0,265^2 = 0,400$. Zpětnou substituci dostáváme zobecněné řešení $\mathbf{x} = \mathbf{Vz} = (-0,253; 0,067; 0,386)^T$. Pto tento vektor je $\mathbf{Ax} = (4,395; 4,126; 4,795; 4,995; 5,195)^T$ nejblíže vektoru $\mathbf{b}$ ve smyslu euklidovské normy.

11.4.4 Příklad.

Chceme pomocí normální soustavy řešit pčeurčenou soustavu

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ -1 & 1 & 0 \\ 0 & -1 & 1 \\ -1 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 1 \\ 2 \\ 3 \\ 1 \\ 2 \\ 1 \end{pmatrix}.$$

Normální soustava má tvar

$$\begin{pmatrix} 3 & -1 & -1 \\ -1 & 3 & -1 \\ -1 & -1 & 3 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} -1 \\ 1 \\ 6 \end{pmatrix}.$$

Gaussovou eliminací dostaneme jediné řešení $\mathbf{x}_p = \left(\frac{5}{4}, \frac{7}{4}, 3\right)^T$, které je zobecněným řešením dané přeurčené soustavy. Snadno prověříme, že rezidouvý vektor $\mathbf{r} = \frac{1}{4}(-1, 1, 0, 2, 3, -3)^T$ je ortogonální ke sloupcům matice $\mathbf{A}$, tj. platí $\mathbf{A}^T(\mathbf{b} - \mathbf{A}\mathbf{x}_p) = \mathbf{0}$, a tedy platí (11.4.2).

11.5 Pseudoinverzní matice.

Matici $\mathbf{X}$ typu (n, m) nazýváme *Penrosovou-Moorevou pseudoinverzní maticí* k matici $\mathbf{A}$ typu (m, n), platí-li následující čtyři podmínky:

(1) $\mathbf{AXA} = \mathbf{A}$,
(2) $\mathbf{XAX} = \mathbf{X}$,
(3) $\mathbf{AX}$ je symetrická matice,
(4) $\mathbf{XA}$ je symetrická matice.

Lze dokázat, že taková matice $\mathbf{X}$ vždy existuje a je jediná. Le-li matice $\mathbf{A}$ čtvrcová a regulární, potom $\mathbf{X} = \mathbf{A}^{-1}$. Je-li matice $\mathbf{A}$ obdélníková a $h(\mathbf{A}) = n$, potom $\mathbf{X} = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T$. Jestliže $h(\mathbf{A}) < n$ nemáme k dispozici žádnou jednoduchou reprezentaci pseudoiverzní matice. V tomto případě určujeme pseudoinverzní matici - kterou v dalším značíme $\mathbf{A}^P$ - pomocí singulárního rozkladu.

Definujeme nejdříve *pseudoinverzní číslo*:

$$\sigma^P = \begin{cases} 1/\sigma, & \text{když} \quad \sigma \neq 0, \\ 0, & \text{když} \quad \sigma = 0 \end{cases} .$$

Je-li nyní

$$\mathbf{\Sigma} = \begin{pmatrix} \sigma_1 & 0 & \cdots & 0 \\ 0 & \sigma_2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ \vdots & & \ddots & \sigma_n \\ \vdots & & & 0 \\ \vdots & & & \vdots \\ 0 & \cdots & \cdots & 0 \end{pmatrix},$$

potom definujeme matici Σ^P typu (n, m) pomocí pseudoinverzních čísel σ^P:

$$\mathbf{\Sigma}^P = \begin{pmatrix} \sigma_1^P & 0 & \cdots & \cdots & \cdots & \cdots & 0 \\ 0 & \sigma_2^P & \ddots & & & & \vdots \\ \vdots & \ddots & \ddots & \ddots & & & \vdots \\ 0 & \cdots & 0 & \sigma_n^P & 0 & \cdots & 0 \end{pmatrix}.$$

Poznamenejme, že $\mathbf{\Sigma\Sigma}^P = \mathbf{I}_r$, kde $\mathbf{I}_r$ má r jedniček na diagonále a ostatní prvky má nulové. Je-li tedy dán singulární rozklad matice $\mathbf{A}$, tj. $\mathbf{A} = \mathbf{U\Sigma V}^T$, potom definujeme pseudoinverzní matici vztahem

$$\mathbf{A}^P = \mathbf{V\Sigma}^P\mathbf{U}^T.$$

Lehce se přesvědčíme, že matice $\mathbf{A}^P$ splňuje podmínky z úvodu tohoto odstavce.

Vektor $\mathbf{x} = \mathbf{A}^P\mathbf{b}$ je řešením soustavy $\mathbf{Ax} = \mathbf{b}$ za předpokladu, že vektory $\mathbf{Ax}$ a $\mathbf{b}$ patří do téhož podprostoru. Pokud tato podmínka splněna není, potom $\mathbf{x} = \mathbf{A}^P\mathbf{b}$ má nejmenší normu ve všech vektorů $\mathbf{x}$ minimalizujících $\|\mathbf{b} - \mathbf{Ax}\|_{\mathbf{E}}$. Z numerického hlediska je užitečné zavést pojem *efektivní pseudoinverzní matice* $\mathbf{A}^P_\varepsilon$.

Definuje se *efektivní pseudoinverzní číslo* (ε je zadaná přesnost):

v ostatních případech

$$\sigma^P = \begin{cases} 1/\sigma, & \text{když} \quad \sigma > \varepsilon, \\ 0, & \text{v ostatních případech,} \end{cases}$$

a odtud

$$\mathbf{A}^P_\varepsilon = \mathbf{V\Sigma}^P_\varepsilon\mathbf{U}^T.$$

Matice $\mathbf{X} = \mathbf{A}^P_\varepsilon$ splňuje podmínky:

(1) $\|\mathbf{AXA} - \mathbf{A}\| < \varepsilon$,
(2) $\mathbf{XAX} = \mathbf{X}$,
(3) $\mathbf{AX}$ je symetrická matice,
(4) $\mathbf{XA}$ je symetrická matice.

Taková matice $\mathbf{X}$ ovšem nemusí být jediná.

11.6 Poznámka.

Singulární rozklad matice nachází uplatněníá i při řešení klasických úloh lineární algebry. K podrobnějším poučení odkazujeme čtenáře na [7a], kde kromě odvození algoritmu je též uveden program ve Fortranu.

II. Řešení nelineárních rovnic

12 Formulace problému

12.1

V této kapitole popíšeme některé numerické metody následujících úloh:

a) Je dána funkce $f : \mathbf{R} \to \mathbf{R}$, definovaná na intervalu $\langle a, b\rangle$. Chceme stanovit reálné číslo $\alpha \in \langle a, b\rangle$ (pokud existuje), pro které platí

$$f(\alpha) = 0$$

Takovému číslu α říkáme *kořen rovnice* $f(x) = 0$.

b) Je dáno zobrazení $\mathbf{F} : \mathbf{R}^n \to \mathbf{R}^n$, definované v oblasti $\Omega \subset \mathbf{R}^n$. Chceme stanovit n-tici $\alpha = (\alpha_1, \alpha_2, ..., \alpha_n) \in \Omega$ reálných čísel α_i takovou, že platí

$$\mathbf{F}(\alpha) = \mathbf{0}.$$

Protože zobrazení $\mathbf{F}$ bývá obvykle reprezentováno n funkcemi o n proměnných $F_1(x_1, x_2, ..., x_n), F_2(x_1, x_2, ..., x_n), ..., F_n(x_1, x_2, ..., x_n)$ říkáme, že n-tici $\alpha = (\alpha_1, ..., \alpha_n)$ je *řešením soustavy* (obecně *nelineárních rovnic*)

$$\begin{aligned} F_1(x_1, x_2, ..., x_n) &= 0, \\ F_2(x_1, x_2, ..., x_n) &= 0, \\ &\cdots\cdots\cdots\cdots \\ F_n(x_1, x_2, ..., x_n) &= 0. \end{aligned}$$

c) Chceme stanovit v š e c h n y kořeny rovnice

$$P_n(x) = 0,$$

kde P_n je polynom stupně n.

Kořeny polynomu P_n stupně n [tj. kořeny algebraické rovnice $P_n(x) = 0$] rozumíme nejen reálná, ale i imaginární čísla α taková, že $P_n(\alpha) = 0$.

Je známo, že polynom stupně n má právě n komplexních kořenů, počítáme-li každý kořen tolikrát, kolik je jeho násobnost. Má tedy úloha o hledání (všech) kořenů polynomu P_n oproti úloze a) svou specifiku (víme předem, kolik je kořenů, pracujeme s komplexními čísly) a budeme se jí zabývat odděleně.

12.2

Numerické metody, kterými se budeme nyní zabývat, jsou založeny na iteračních principech (viz odst. 1.4). Budeme hledat odpovědi na dvě základní otázky:

(i) Konvergují dané iterace k hledanému kořenu?

(ii) Jestliže ano, jak rychle?

Jestliže nemáme předběžné informace o poloze kořenů, víme pouze, že v určitém intervalu a, b kořen leží, užijeme k výpočtu takové iterační metody, jejíž konvergence nezávisí na volbě počáteční aproximace. Tyto tzv. *vždy konvergentní metody* mají většinou tu nevýhodu, že konvergují pomalu, a hodí se tedy především k určení takové aproximace kořene, která může být použita jako počáteční aproximace pro nějakou rychleji konvergující metodu. Konvergence takové "lepší" metody už ,úže dost silně závisetnatom, jak dobrá je tato počáteční aproximace, a na vlastnostech funkce f v okolí kořene. Je tedy rozumné rozdělit metody řešení nelineárních rovnic na dva typy:

a) startovací metody (vždy konverdentní metody);

b) zpřesňující metody. Tímto rozdělením ovšem nechceme zdůraznit, že startovací metoda konverguje pomalu vždy a naopak, že zpřesňující metoda vždy rychle konverguje.

Poznamenejme, že v celé této kapitole předpokládáme (i když to často nebudeme zdůraznovat), že funkce $f : \mathbf{R} \to \mathbf{R}$ je v uvažovaném intervalu $\langle a, b\rangle$ spojitá.

12.3 Pojem rychlosti konvergence.

Budeme říkat, že posloupnost x_k konverguje k číslu *alpha s rychlosti řádu* r, jestliže pro $k \to \infty$

$$|x_{k+1}-\alpha| = C|x_k-\alpha|^r+o(|x_k-\alpha|^r), \quad C \neq 0 \tag{12.3.1}$$

Takto definované *rychlosti konvergence* se někdy říká *asymptotická*, nebot' platnost vztahu (12.3.1) se uvažuje pro velká k. Symbol $o(g)$ byl zaveden v odst. 3.2.1.

V odstavci 7.5 vylo ukázáno, že $\|\mathbf{x}_{k+1} - \mathbf{x}_t\| \leqq \|\mathbf{H}\| \left\|\mathbf{x}^{(k-1)} - \mathbf{x}_t\right\|$, a tedy iterační metody pro řešení soustav lineárních rovnic uvedené v článku 7 konvergují z hlediska definice (12.3.1) s rychlostí řádu $r = 1$.

Definice rychlosti konvergence, uvedená v odst. 7.6.1, se tedy liší od definice uvedené zde.

13 Startovní metody

13.1 Grafická metoda.

Z pečlivého nakreslení grafu funkce $y = f(x)$ můžeme provést lokalizaci reálných kořenů rovnice $f(x) = 0$, tj. určení intervalů, ve kterých kořeny rovnice určitě leží. Protože kořeny rovnice $f(x) = 0$ jsou x-ové souřadnice průsečíků grafu funkce $y = f(x)$ s osou x (obr. 5), můžeme si odtud udělat dosti konkretní představu o jejich poloze.

Někdy je výhodnější rovnice $f(x) = 0$ psát ve tvaru $f_1(x) = f_2(x)$; kořeny pak určíme jako x-ové souřadnice průsečíků grafů funkcí $y = f_1(x)$ a $y = f_2(x)$ (obr. 6).

Grafická metoda nám též může poskytnout informaci, zda reálný kořen dané rovnice v uvažovaném intervalu vůbec neexistuje.

Pro složitější typy funkcí můžeme lokalizaci reálných kořenů provést tabelováním funkce $y = f(x)$.

13.2 Metoda bisekce.

13.2.1

Předpokládáme, že

(i) reálná funkce f je spojitá pro $x \in I_0 = \langle a_0, b_0 \rangle$,

(ii) $f(a_0)f(b_0) < 0$ [funkční hodnoty v koncových bodech intervalu I_0 mají opačná znaménka, tj. $\operatorname{sgn} f(a_0) = \operatorname{sgn}(b_0)$].

Tyto předpolády nám zaručují, že existuje alespoň jedno číslo $\alpha \in I_0$ takové, že $f(\alpha) = 0$.

Sestrojíme posloupnost intervalů

$$I_0 \supset I_1 \supset I_2 \supset \ldots \supset I_{k-1} \supset I_k \supset \ldots, \quad I_k = \langle a_k, b_k \rangle,$$

takto (obr. 7): Je-li $f(a_{k-1})f(b_{k-1}) < 0$, potom $I_k = \langle a_k, b_k \rangle$ bude ten z intervalů $\langle a_{k-1}, s_k \rangle, \langle s_k, b_{k-1} \rangle$, v jehož koncových bodech má funkce f opačná znaménka; s_k je střed intervalu $I_{k-1} = \langle a_{k-1}, b_{k-1} \rangle$, tj.

$$s_k = \frac{1}{2}(a_{k-1} + b_{k-1}).$$

Budou-li naše výpočty přesné, bude kořen α ležet v každém z intervalu $I_k = \langle a_k, b_k \rangle$ a posloupnosti $\{a_k\}$, $\{b_k\}$ koncových bodů těchto intervalů vždy konverguji ke kořenu α. Po n krocích bude interval I_n mít délku

$$b_n - a_n = \frac{1}{2}(b_{n-1} - a_{n-1}) = \frac{1}{2^2}(b_{N-2} - a_{n-2}) = \ldots = \frac{1}{2^n}(b_0 - a_0). \tag{13.2.1}$$

Pro *odhad chyby* proto platí (nebot' $\alpha \in I_n$)

$$|a_n - \alpha| \leqq \frac{b_0 - a_0}{2^n}, \quad \text{resp.} \quad |b_n - \alpha| \leqq \frac{b_0 - a_0}{2^n}. \tag{13.2.2}$$

Metoda bisekce konverguje pomalu. V každém kroku se přesnost zlepšuje pouze o jednu dvojkovou číslici. Protože $10^{-1} \approx 2^{-3,3}$, zlepší se přesnost o jednu decimální číslici po $3,3$ krocích. Je ovšem zřejmé, že rychlost konvergence metody je zcela nezávislá na funkcí f.

13.2.2 Algoritmus metody bisekce.

1. Volíme číslo ε a interval $I - 0 = \langle a_0, b_0 \rangle$.

2. Stanovíme střed intervalu $I_0 : s_1 = \frac{1}{2}(a_0 + b_0)$.

3. Když $f(s_1) = 0$, pak $\alpha = s_1$ a algoritmus končí.

4. Pokud $f(s_1) \neq 0$, pak

$$I_1 = \langle a_1, b_1 \rangle = \begin{cases} \langle a_0, s_1 \rangle, & \text{když} \quad \operatorname{sgn} f(s_1) = \operatorname{sgn} f(b_0), \\ \langle s_1, b_0 \rangle, & \text{když} \quad \operatorname{sgn} f(s_1) = \operatorname{sgn} f(a_0), \end{cases}$$

5. Pokud $|b_1 - a_1| \geqq \varepsilon$, přeznačíme $a_1 \to a_0$, $b_1 \to b_0$ a pokračujeme od bodu 2; v opáčném případě výpočet ukončíme a a_1 (resp. b_1) je aproximace kořene s chybou menší než ε.

Tab. 10

13.2.3 Příklad.

Stanovme aproximaci kořene rovnice

$$f(x) = \left(\frac{x}{2}\right)^2 - \sin x = 0.$$

Volme $\varepsilon = 0,005$.

Z grafu funkcí $y = f(x/2)^2$, $y = \sin x$ odhadneme $I_0 = \langle 1,5; 2 \rangle$. Skutečně $f(1,5) < 0$, $f(2) > 0$ a v I_0 je kořen rovnice $f(x) = 0$. Výsledky výpočtů [v $M(10,6)$] podle předchozího algoritmu zapisujeme do tabulky (viz tab. 10).

Podle (13.2.2) je

$$|a_4 - \alpha| < \frac{1}{2^4} \cdot 0,5 < 0,05 \quad (\text{stejně pro} \quad |b_4 - \alpha|).$$

Později (odst. 13.3.1) si ukážeme, že b_4 je lepší aproximací kořene α.

13.2.4

Výhodou metody bisekce je kromě její jednoduchosti i ten fakt, že se dá předem určit [podle vztahu (13.2.2)] počet kroků, potřebných k dosažení požadované přesnosti. Nevýhodou je pomalá konvergence - proto je vhodné kombinovat ji s některou rychleji konvergující metodou. Kromě toho metodu bisekce nemůžeme užít k určení komplexního kořene - např. u polynomů.

13.3 Metoda prosté iterace.

Rovnici $f(x) = 0$ přepíšeme na tvar (bývá většinou několik možností)

$$x = \varphi(x) \tag{13.3.1}$$

a předpokládáme, že existuje interval $I_0 = \langle a_0, b_0 \rangle$ patřící definičnímu oboru a oboru spojitosti jak funkce f, tak funkce φ takový, který obsahuje společný kořen α obou uvedených rovnic (omezujeme se na reálné funkce ϕ a reálné kořeny) a pro který je splněna podmínka $\varphi(I_0) \subset I_0$.

Pomocí (13.3.1) sestrojíme posloupnost postupných aproximací $x_1, x_2, x_3, ...$ kořene α podle následujícího algoritmu:

1. Zvolíme číslo $\delta > 0$ a počáteční aproximaci $x_0 \in I_0$.
2. Další aproximaci určíme z iterační formule

$$x_k = \varphi(x_{k-1}), \quad k = 1, 2, 3, ... \tag{13.3.2}$$

3. Bude-li $|x_k - x_{k-1}| \geqq \delta$, pokračujeme podle bodu 2; v opačném případě výpočet zastavíme a x_k považujeme za aproximaci kořene α. Tento algoritmus lze psát (vyjma bodu 3) naším obvyklým způsobem

Vstup: $\delta, x_0, \varphi, (n)$.
Pro $i = 1, 2, ..., (n)$:
$\lfloor x_k = \varphi(x_{k-1})$.
Výstup: x_n.

Parametr n obvykle na vstupu nezadáváme, ale proces výpočtu zastavujeme zastavovací podmínkou $|x_n - x_{n-1}| < \delta$.

K realizaci uvedeného algoritmu musíme mít zaručeno, že posloupnost $\{x_k\}$ určená rekurentním vztahem (13.2.2) konverguje ke kořenu $\alpha \in I_0$. K této otázce se ještě vrátíme v odst. 13.3.5 a ukážeme si, jak sranovit odhad chyby, tj. číslo $\varepsilon = \varepsilon(\delta)$ pro které

$$|x_k - \alpha| < \varepsilon. \tag{13.3.3}$$

13.3.1 Příklad.

Hledejme kořen rovnice $f_x) \equiv (x/2)^2 - \sin x = 0$ v intervalu $I_0 = \langle 1,5; 2\rangle$)z příkl. 13.2.2 už víme, že v tomto intervalu leží alespoň jeden kořen dané rovnice). Za počáteční aproximaci x_0 můžeme volit libovolný bod intervalu I_0 (v tab. 11 jsou voleny koncové body).

Tab. 11

Uvedenou rovnici lze přepsat na tvar

$$x = 2\sqrt{\sin x} \quad \text{nebo} \quad x = \arcsin(\frac{x}{2})^2;$$

budeme počítat [v $M(10,6)$] podle iterační formule

$$x_k = 2\sqrt{\sin x_{k-1}} \quad \text{nebo} \quad x = \arcsin(\frac{x_{k-1}}{2})^2$$

a výpočet zastavíme, bude-li $|x_k - x_{k-1}| < 10^{-3} = \delta$. Lehce se přesvědčíme, že interval $I_0 = \langle 1,5; 2\rangle$ patří do definičního oboru funkce $\varphi(x) = s\sqrt{\sin x}$ i funkce $\varphi(x) = \arcsin(x/2)^2$. Z počtu iterací usuzujeme na rychlost konvergence. Je patrná závislost na volbě x_0. Druhá iterační formule konverguje nejrychleji (poslední sloupec tabulky), ovšem ke kořenu $\alpha = 0$, který neleží v intervalu I_0.

13.3.2 Poznámka.

Z předchozího výkladu by čtenář mohl získat dojem, že metoda prosté iterace je lepší než metoda bisekce. To může být pravda pouze pro některé (jednoduché) typy rovnic.

Potížr mohou nastat při výběru funkce φ a určení intervalu I_0 - existence kořene v I_0 totiž ještě nezaručuje konvergenci iteračního procesu (13.3.2).

13.3.3 Postačující podmínky konvergence.

Předpokládáme, že funkce φ je na nějakém intervalu $I \in \langle a,b\rangle$ spojitá a má tyto vlastnosti:

(13.3.3) (a) $\forall x \in I: \quad \varphi(x) \in I$ (*funkce φ zobrazuje I do sebe*),

(b) $\exists q \in \langle 0,1): \quad |\varphi(x) - \varphi(y)| \leqq q|x-y| \quad \forall x,y \in I$.

Potom

1. *v intervalu I existuje právě jeden (reálný) kořen α rovnice $x = \varphi(x)$*,

2. it posloupnost $\{x_k\}_{k=1}^{\infty}$ *určená formulí* $x_k = \varphi(x_{k-1})$ *konverguje pro každé* $x_0 \in I$ *a je* $\lim\limits_{k\to\infty} x_k = \alpha$.

D ů k a z. Z předpokládu (a) plynou nerovnosti $\varphi(a) \geqq a$, $\varphi(b) \leqq b$ [případy $\varphi(a) = b$, $\varphi(b) = a$ nastat nemohou], které zaručují existenci kořene α. Jednoznačnost je důsledkem (b). Ze zřejmého vztahu $x_k - \alpha = \varphi(x_{k-1}) - \varphi(\alpha)$ a z (b) dostaneme $|x_k = \alpha| \leqq q|x_{k-1} - \alpha|$ a tedy máme

$$|x_k = \alpha| \leqq q^k |x_0 - \alpha|, \quad x_0 \in I.$$

Protože $\lim\limits_{k\to\infty} q^k = 0$, musí být $\lim\limits_{k\to\infty} |x^k - \alpha| = 0$.

Podmínky (13.3.3) (a), (b) jsou tedy postačujícími podmínkami konvergence metody prosté iterace. Pro deferencovatelnou funkci φ lze podmínku (b) nahradit podmínku [ze které plyne (b)]:

$$\text{(b')} \quad \exists q: \ |\varphi'(x)| \leqq q < 1 \quad \forall x \in I.$$

13.3.4 Příklad.

Vyšetříme možnost použití metody prosté iterace k rovnici $f(x) \equiv x + \sin x - 1 = 0$. Protože $f(0) = -1 < 0$, $f(5\pi/2) = 5\pi/2 - 1 > 0$ a funkce f je spojitá pro všechna $x \in \mathbf{R}$, existuje v intervalu $\langle 0, 5\pi/2\rangle$ alespoň jeden reálný kořen uvedené rovnice.

Převedeme rovnici např. na tvar

$$x = -\sin x + 1$$

a vyšetříme funkci $\varphi(x) - \sin x + 1$. Podmínka (13.3.3) (b') $|\varphi'(x)| = |-\cos x| < 1$ je splněna pro všechna $x \neq k\pi$, $k = 0, 1, 2, \ldots$. Avšak funkce φ zobrazuje pouze interval $\langle 0, \pi/2\rangle$ do sebe - konkrétně $\varphi(\langle 0, \pi/2\rangle) = \langle 0, 1\rangle$. Zvolíme-li $I_0 = \langle 0, 1; \pi/2\rangle$, budou na tomto intervalu splněny obě podmínky (13.3.3) a iterační proces $x_k = -\sin x_{k-1} + 1$ bude konvergovat pro libovolné $x_0 \in I_0$ k jedinému kořenu $\alpha \in I_0$ ($\alpha \approx 0,510\,97$).

13.3.5 Příklad.

Ilustrujme si graficky konstrukci posloupnosti $x_k = \varphi(x_{k-1})$.

Kořen α je průsečík grafů funkcí $y = x$, $y = \varphi(x)$. Zvolíme x_0 a určíme bod $P_0 = [x_0, 0]$. Potom bod P_1 bude mít souřadnice $[x_0, \varphi(x_0)] \equiv \equiv [x_0, x_1]$. Bodem P_1 vedeme rovnoběžku s osou x a určíme její průsečík

P_2 s přímkou $y = x$. Rovnoběžka s osou y vedená bodem P_2 protne křivku $y = \varphi(x)$ v bodě P_3. Opakováním tohoto procesu sestrojujeme postupně body $P_4, P_5, P_6, \ldots$ atd. x-ové souřadnice bodů $P_1, P_3, P_5, \ldots$ atd. určují posloupnost $\{x_k\}$. Na obrázku 8 je znázorněn konvergentní iterační proces a na obrázku 9 divergentní proces.

13.3.6 Odhad chyby metody prosté iterace a rychlost konvergence.

Máme-li konvergentní iterační proces $x_k = \varphi(x_{k-1})$, pak $\alpha = \varphi(\alpha)$, kde $\alpha = \lim_{k\to\infty} x_k$. Odečtením těchto dvou rovností dostáváme

$$x_k - \alpha = \varphi(x_{k-1}) - \varphi(\alpha). \tag{13.3.4}$$

Protože

$$|x_{k-1} - \alpha| = |x_{k-1} - x_k + x_k - \alpha| \leqq |x_{k-1} - x_k| + |x_k - \alpha|,$$

dostáváme z podmínky (13.3.3) (b) odhad

$$|x_{k-1} - \alpha| \leqq q|x_{k-1} - \alpha| \leqq q|x_{k-1} - x_k| + q|x_k - \alpha|,$$

a odtud (protože $i - q > 0$)

$$|x_k - \alpha| \leqq \frac{q}{1-q}|x_k - x_{k-1}|. \tag{13.3.5}$$

Jestliže jsme výpočet zastavili podmínkou $==$, potom pro odhad chyby máme

$$|x_k - \alpha| \leqq \frac{q}{1-q}\delta. \tag{13.3.6}$$

Pro funkci φ z příkl. 13.3.1 je

$$\varphi'(x) = \left|\frac{\cos x}{\sqrt{\sin x}}\right| < \frac{1}{2} = q \quad \text{pro} \quad x \in \langle 1,5; 2\rangle;$$

potom

$$|x_7 - \alpha| \leqq \frac{\frac{1}{2}}{1-\frac{1}{2}} \cdot 10^{-3} = 10^{-3}.$$

Je-li funkce φ dostatečně hladká, můžeme napsat její Taylorův rozvoj v bodě α, a pak pro $x = x_{k-1}$, máme

$$\varphi(x_{k-1}) = \varphi(\alpha) + \varphi'(\alpha)(x_{k-1} - \alpha) + \frac{\varphi''(\alpha)}{2}(x_{k-1} - \alpha)^2 + \frac{\varphi'''(\xi)}{6}(x_{k-1} - \alpha)^3,$$

tj.

$$x_k-\alpha=\varphi'(\alpha)(x_{k-1}-\alpha)+\frac{\varphi''(\alpha)}{2}(x_{k-1}-\alpha)^2+\frac{\varphi'''(\xi)}{6}(x_{k-1}-\alpha)^3, \tag{13.3.7}$$

Je-li $\varphi'(\alpha)\neq 0$, potom odtud plyne

$$x_k-\alpha=\varphi'(\alpha)(x_{k-1}-\alpha)+o((x_{k-1}-\alpha)) \tag{13.3.8}$$

a porovnáním s definicí (12.3.1) vidíme, že rychlost konvergence je řádu 1 (hovoříme také o lineárním iteračním procesu).

Je-li $\varphi'(\alpha)=0$ a $\varphi''(\alpha)\neq 0$, pak

$$x_k-\alpha=\frac{\varphi''(\alpha)}{2}(x_{k-1}-\alpha)^2+o((x_{k-1}-\alpha)^2) \tag{13.3.9}$$

a rychlost konvergence je v tomto případě řádu 2 (kvadratický iterační proces).

O tom, kolik iterací potřebujeme k dosažení zadané přesnosti, také rozhoduje hodnota $\varphi'(\alpha)$, resp. $\varphi''(\alpha)$ [když $\varphi'(\alpha)=0$], jak je patrné ze vztahu (13.3.6), nebot' $q\approx|\varphi'(\alpha)|$.

13.4 Metoda regula falsi.

Uvažujeme opět rovnici $f(x)=0$ a předpokládáme, že funkce f je spojitá v intervalu $I=\langle a,b\rangle$ a že $f(a)f(b)<0$ (tj. že v intervalu I existuje reálný kořen uvedené rovnice).

Vypočteme x-ovou souřadnici průsečíku sečny křivky $y=f(x)$ sestrojené v bodech $A=[a,f(a)]$, $B=[b,f(b)]$ podle vzorce

$$s_1=a-\frac{f(a)}{f(b)-f(a)}(b-a). \tag{13.4.1}$$

Bude-li $\operatorname{sgn} f(s_1)=\operatorname{sgn} f(a)$, potom přeznačíme s_1 na a a počítáme s_2 podle stejného vzorce. Bude-li $\operatorname{sgn} f(s_1)=\operatorname{sgn} f(a)$, přeznačíme s_1 na b a dále počítáme s_2 opět podle (13.4.1). Situace je zřejmá z obr. 10. Proces výpočtu zastavíme např. podmínkou $f(s_k)<\delta$. Metoda regula falsi je vždy konvergentní metodou, tj. sestrojená posloupnost s_k vždy konverguje ke kořenu α, pokud je jeho existence zaručena. Dá se ukázat, že její rychlost je $r=1$. Nedoporučuje se užívat této metody "příliš" blízko kořene.

Protože podle věty o střední hodnotě

$$f(s_k)-f(a)=f'(\xi)(s_k-\alpha).$$

dostáváme pro odhad chyby

$$|s_k - \alpha| \leqq \frac{f(s_k)}{m}, \quad m = \min_{x \in I} |f'(x)|. \tag{13.4.2}$$

Poznamenejme, že tento odhad není závislý na konkrétní metodě.

13.4.1 Příklad.

Metodou regula falsi řešme rovnici $f(x) \equiv (x/2)^2 - \sin x = 0$. Volme $I_0 = \langle 1,5; 2\rangle$. Viz tab. 12. Výpočet jsme zastavili podmínkou $|f(s_k)| < 10^{-5}$.

Tab. 12

13.5 Cvičení.

13.5.1

Grafickou metodou určete intervaly, v nichž existuje jediný reálný kořen následujících rovnic:
a) $x^3 - 3x + 1 = 0$; b) $e^x = x + 2$; c) $e^x = 1 + 1/x$; d) $\sin x = 3x - 2$.
[a) $\langle -2, -1\rangle$; $\langle 0, 1\rangle$; $\langle 1, 2\rangle$; b) $\langle -2,5; -1,5\rangle$; $\langle 0,5; 1,5\rangle$; c) $\langle -1,5; -1\rangle$; $\langle 0,6; 1\rangle$; d) $\langle 0,8; 1,2\rangle$.]

13.5.2

Metodou bisekce nebo metodou regula falsi určete aproximace kořenů rovnic ze cvič. 13.5.1 s chybou $\varepsilon = 10^{-2}$. U metody bisekce určte počet kroků potřebných k dosažení požadované aproximace.

13.5.3

Metodou prosté iterace řešte rovnice ze cvič. 13.5.1. Proveřte postačující podmínky konvergence. Iterační proces zastavte podmínkou $|x - x_{k-1}| < 10^{-3}$. Určete odhad chyby vypočtené aproximace kořene. [a) $0,347$; $1,532$; $-1,879$; b) $1,146$; $-1,841$; c) $0,806$; d) $0,934$.]

13.5.4

Ilustrujte graficky iterační proces $x_k = \varphi(x_{k-1})$, když:
a) $-1 < \varphi'(x) < 0$; b) $\varphi'(x) > 1$; c) $\varphi'(x) < -1$.

13.5.5

Metodou prosté iterace řešte rovnici $x + \ln x = 0$. Posud'te iterační formule

$$x_k = -\ln x_{k-1}; \quad x_k = e^{-x_{k-1}}; \quad x_k = \frac{x_{k-1} + e^{-x_{k-1}}}{2}$$

z hlediska a) konvergence; b) rychlosti konvergence.

13.5.6

Metodou prosté iterace stanovte kladné kořeny rovnice $x \cos x = \sin x - -\pi/2$. Vyjasněte otázku konvergence metody pro různě volené funkce φ. Počítejte např.v $M(10, 6)$. [$\alpha \equiv 1,905\,69$.]

13.5.7

Lze řešit kvadratickou rovnici $x^2 - 2x + 2 = 0$ iterační formulí $x_k = = (x_{k-1}^2)/2$? Jsou splněny postačující podmínky konvergence?

13.5.8

Ukažte pomocí Taylorovy řady, že pro chybu $\varepsilon_k = x_k - \alpha$ metody prosté iterace platí $\varepsilon_{k+1} = \varphi'(\alpha)\varepsilon_k + \frac{1}{2}\varphi''(\alpha)\varepsilon_k^2 + \frac{1}{6}\varphi'''(\alpha)\varepsilon_k^3 + \ldots$. Určete pomocí tohoto vztahu rychlost konvergence metody, je-li α vícenásobným kořenem rovnice $x = \varphi(x)$. [Návod. Stanovte Taylorův rozvoj v bodě α a určete z něho $\varphi(x_k)$.]

14 Zpřeňující metody

Efektivní algoritmy řešení nelineární rovnice mají obvykle dvě části. V prvé části se využívá některé startovací metody a v druhé části se přejde k

nějaké rychleji konvergující metodě, která slouží ke zpřesnění počítané aproximace kořene.

14.1 Newtonova metoda.

Necht' jednoduchý reálný kořen α rovnice $f(x) = 0$ leží v intervalu I. Zvolíme $x_0 \in I$ a vyjádříme funkci f ve tvaru (Taylorův rozvoj v bodě x_0)

$$f(x) = f(x_0)+f'(x_0)(x-x_0)+\frac{1}{2}f''(\xi_0)(x-x_0)^2, \tag{14.1.1}$$

přičemž předpokládáme, že v intervalu I existují derivace f', f''.

Rovnici $f(x) = 0$ nahradíme (aproximujeme) lineární rovnicí [první dva členy rozvoje (14.1.1)]

$$f(x_0)+f'(x_0)(x-x_0) = 0 \tag{14.1.2}$$

a stanovíme její kořen

$$x_1 = x_0-\frac{f(x_0)}{f'(x_0)}. \tag{14.1.3}$$

Nyní nahradíme rovnici $f(x) = 0$ lineární rovnicí

$$f(x_1) + f'(x_1)(x - x_1) = 0,$$

která opět vyplývá z Taylorova rozvoje typu (14.1.1), ovšem v bodě x_1. Kořenem této lineární rovnice je číslo

$$x_2 = x_1 - \frac{f(x_1)}{f'(x_1)}.$$

Opakovaná náhrada rovnice $f(x) = 0$ lineárními rovnicemi typu

$$f(x_k) + f'(x_k)(x - x_k) = 0$$

je základní myšlenkou *Newtonovy metody*. Proto se také Newtonově metodě často říká *metody linearizace*. Kořeny těchto lineárních rovnic tvoři posloupnost, která ke určená rekurentní formulí

$$x_{k+1} = x_k + h_k, \quad h_k = -\frac{f(x_k)}{f'(x_k)}. \tag{14.1.4}$$

je to iterační formule (*Newtonova iterační formule*), nebot' požadujeme, aby $\lim_{k\to\infty} f(x_k) = f(\alpha) = 0$. pro spojitou funkci f to znamena, že kořen α musí být limitou posloupnosti $\{x_k\}$.

Iterační proces (14.1.4) zastavujeme podmínkou $|h_k| < \delta$, kde δ (požadovaná přesnost) zadáváme na vstupu. K zastavení procesu můžeme ovšem použít i podmínku $|f(x_k)| < \delta$.

V geometrické interpretaci formule (14.1.4) je bod $[x_{k+1}, 0]$ průsečíkem tečny sestrojené v bodě $[x_k, f(x_k)]$ ke křivce $y = f(x)$ s osou x. Proto také hovoříme o *metodě tečen.*

Algoritmus Newtonovy metody odpovídá algoritmu metody prosté iterace pro funkci

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

14.1.1 Příklad.

Stanovme aproximaci kořene $\alpha \in \langle 1,5; 2\rangle$ rovnice $f(x) \equiv (x/2)^2 - \sin x = 0$ Newtonovou metodou. Výpočet v $M(10,6)$ zastavme, bude-li $|h_k| = |x_{k+1} - x_k| < 10^{-5}$. Volíme $x_0 - 1,5$ (Viz tab. 13.) Protože $|h_4| < 10^{-5}$, potom $\alpha \approx 1,933\,75$. Na počítači s $M(10,6)$ lepší výsledek nedostaneme. V $M(10,10)$ lze stanovit $\alpha \approx 1,933\,753\,764$.

Tab. 13

14.1.2 Konvergence Newtonovy metody.

Vyšetříme naní podmínky konvergence poslounosti $\{x_k\}$ určené formulí (14.1.4). Konverguje-li tato posloupnost, pak $\lim k \to \infty h_k = 0$, a tedy $\lim k \to \infty f(x_k) = 0$. Pro spojitou funkci f to znamená, že limita poslounosti $\{x_k\}$ je kořenem rovnice $f(x) = 0$.

Z Taylorova rozvoje (14.1.1) pro $x = \alpha$ vyplývá [využíváme skutečnost, že $f(x) = 0$]

$$\alpha - x_0 = \frac{f(x_0)}{f'(x_0)} = \frac{1}{2}\frac{f''(\xi_0)}{f'(x_0)}(\alpha - x_0)^2.$$

Dosazením do vzorce (14.1.3) dostaneme

$$\alpha - x_1 = \frac{1}{2}\frac{f''(\xi_0)}{f'(x_0)}(\alpha - x_0)^2. \tag{14.1.5}$$

Označíme-li $\alpha - x_k = \varepsilon_k$, $k = 0, 1, ...$ (chyba k-té aproximace kořene α), můžeme vztah (14.1.5) psát ve tvaru

$$\varepsilon_1 = \frac{1}{2}\frac{f''(\xi_0)}{f'(x_0)}\varepsilon_0^2. \tag{14.1.6}$$

Opakováním uvedeného postupu odvodíme vztah

$$\varepsilon_{k+1} = \frac{1}{2}\frac{f''(\xi_k)}{f'(x_k)}\varepsilon_k^2. \tag{14.1.7}$$

Předpokládejme nyní, že v intervalu I (obsahujícím kořen α) platí

$$\frac{1}{2}\left|\frac{f''(\xi)}{f'(\eta)}\right| \leqq C, \quad \xi, \eta \in I, \tag{14.1.8}$$

kde C je nějaké číslo. Potom ze vztahu (14.1.7) vyplývá, že

$$|\varepsilon_{k+1}| \leqq C|\varepsilon_k|^2, \quad \text{tj.} \quad |C\varepsilon_{k+1}| \leqq |C\varepsilon_k|^2.$$

Indukcí lze dokázat, že platí odhad chyby

$$|\varepsilon_{k+1}| \leqq \frac{1}{C}|C\varepsilon_0|^{2k}. \tag{14.1.9}$$

Vybereme-li počáteční aproximaci x_0 tak, aby platilo

$$|C(\alpha - x_0)| < 1, \tag{14.1.10}$$

bude posloupnost ε_k konvergovat k nule a bude $x_k \in I$ pro všechny k. *Postačující podmínka konvergence Newtonovy metody* je tedy vyjádřena vztahem (14.1.10). Protože obvykle nemáme žádné informace o čísle C, musíme volit x_0 dostatečně blízko kořene α, aby tato podmínka byla splněna.

Často se uživá následující snadno ověřitelné *kritérium konvergence: Je-li* $f'(x) \neq 0$, f'' *nemění znaménko vintervalu* $I = \langle a, b\rangle$, *platí* $f(a)f(b) < 0$ *a současně*

$$\left|\frac{f(a)}{f'(a)}\right| < b - a, \quad \left|\frac{f(b)}{f'(b)}\right| < b - a,$$

potom Newtonova metoda konverguje prto libovolné $x_0 \in I$. Zřejmost uvedeného kritéria je patrná z obr. 11.

Porovnáme-li vztah (14.1.7) se vztahem (12.3.1), vidíme, že Newtonova metoda má rychlost konverguje řádu $r = 2$.

Protože konstanta C ve tvaru (14.1.10) se obvykle těžko určuje, uživá se praktického pravidla pro odhad přesnosti, které vyplývá ze vztahu (14.1.7):

Je-li

$$|\alpha - x_k| < 10^{-d},$$

potom zhruba platí

$$|\alpha - x_{k+1}| < 10^{-2d},$$

tj. pro větší k se každou iterací zdvojnásobuje počet platných desetinných míst aproximace kořene.

14.2 Interpolační metody.

14.2.1 Metoda sečen.

Předpokládáme, že $x_k \neq x_{k-1}$ jsou dobré aproximace jednoduchého kořene α rovnice $f(x) = 0$ (obr. 12). Funkci f nahradíme lineární funkcí g:

$$g(x) = \frac{f(x_k) - f(x_{k-1})}{x_k - x_{k-1}}(x - x_k) + f(x_k)$$

(g je lineárním interpolačním polynomem určeým hodnotami $[x_{k-1}, f(x_{k-1})]$, $[x_k, f(x_k)]$ a místo rovnice $f(x) = 0$ řešíme rovnici $g(x) = 0$. Kořen x_{k+1} této rovnice je tedy určen formulí

$$x_{k+1} = x_k + \tau_k, \quad \tau_k = -f(x_k)\frac{x_k - x_{k-1}}{f(x_k) - f(x_{k-1})} \tag{14.2.1}$$

a považujeme jej za aproximaci kořene α rovnice $f(x) = 0$.

Dostali jsme tak dvoukrokovou iterační formulí, tzn. k zahájení výpočtu potřebujeme dvě počáteční aproximace x_0, x_1, ale na rozdíl od Newtonovy metody počítáme na každém kroku pouze jednu novou funkční hodnotu funkce f (úspora času!). Další její výhodou je, že ji lze užít i k funkcím f, které nejsou diferencovatelné stačí, aby byly spojité).

V geometrické interpretaci je graf lineární funkce g sečnou grafu funkce f - odtud název metody.

Formuli (14.2.1) lze také získat přimo z Newtonovy formule (14.1.4), nahradíme-li derivaci $f'(x_k)$ diferenčním podílem

$$\frac{f(x_k) - f(x_{k-1})}{x_k - x_{k-1}}.$$

Jestliže předpokládáme platnost vztahu (14.1.8) i zde, dostaneme úvahami analogickými jako u Newtonovy metody, že platí

$$|\varepsilon_{k+1}| \approx C|\varepsilon_k||\varepsilon_{k-1}|. \tag{14.2.2}$$

Odtud se dá dokázat (viz [4], [12]), že rychlost konvergence metody sečen je řádu $r = \frac{1}{2}(1+\sqrt{5}) \approx 1,618$ (pro $k \ll 1$), tedy poněkud menší než u Newtonovy metody.

Pokud x_0, x_1 nebudou dobré aproximace kořene α, nemusí metoda sečen vůbec konvergovat. Proto je nutné kombinovat ji s některou startovací metodou - např. s metodou regula falsi, která je konstruována na stejném principu.

14.2.2 Mullerova metoda.

Předpokládáme, že máme tři (dobré) aproximace x_0, x_1, x_2 kořene α rovnice $f(x) = 0$. Další (lepší) aproximaci kořene α stanovíme jako ten kořen interpolačního polynomu 2. stupně (parabola) sestrojeného z tabulky $[x_0, f(x_0)]$, $[x_1, f(x_1)]$, $[x_2, f(x_2)]$, který je blíže k x_2. Řešíme tedy kvadratickou rovnici

$$f(x_0)\frac{(x-x_1)(x-x_2)}{(x_0-x_1)(x_0-x_2)} + f(x_1)\frac{(x-x_0)(x-x_2)}{(x_1-x_0)(x_1-x_2)} + \\ + f(x_2)\frac{(x-x_0)(x-x_1)}{(x_2-x_0)(x_2-x_1)} = 0$$

Po ne zcela triviálních úpravách dostaneme tříbodovou iterační formuli

$$x_3 = x_2 - \frac{2f(x_2)}{\omega \pm \sqrt{\omega^2 - 4f(x_2)B}}, \tag{14.2.3}$$

kde

$$\omega = A + (x_2 - x_1)B;$$

$$A = \frac{f(x_2)-f(x_1)}{x_2-x_1}; \quad B = \frac{\frac{f(x_2)-f(x_1)}{x_2-x_1} - \frac{f(x_1)-f(x_0)}{x_1-x_0}}{x_2-x_0}.$$

Znaménko u odmocniny vybíráme tak, aby jmenovatel v absolutní hodnotě byl co největší.

Vztah mezi chybami uvažovaných aproximací má tvar

$$|\varepsilon_3| \approx |\varepsilon_2||\varepsilon_1||\varepsilon_0|. \tag{14.2.4}$$

V kombinaci např. s metodou bisekce poskytuje Mullerova metoda velmi efektivní algoritmus řešení nelineární rovnice.

14.3 Násobné kořeny.

Kořen α rovnice $f(x) = 0$ má násobnost s, jestliže

$$0 \neq g(\alpha) < \infty, \quad \text{kde} \quad g(x) = \frac{f(x)}{(x-\alpha)^s}.$$

Tvrzení o rychlosti konvergence Newtonovy metody a metody sečen pro násobné kořeny neplatí. Např. řád rychlosti konvergence Newtonovy metody v blizkosti dvojnásobného kořene je stejný jako u metody bisekce.

Modifikovaná iterační formule

$$x_{k+1} = x_k + sh_k, \quad h_k = -\frac{f(x_k)}{f'(x_k)} \tag{14.3.1}$$

dává opět kvadratický iterační proces, avšak předpokládá apriorní znalost násobnosti s kořene α.

Jinou možnost poskytuje následující úvaha. Je-li α s-násobný kořen rovnice $f(x) = 0$, potom tento kořen je $s-1$-násobným kořenem ronvice $f'(x) = 0$, a tedy jednoduchým kořenem rovnice $u(x) = 0$, kde $u(x) = f(x)/f'(x)$ (zdůvodněte!).

Jsetliže u Newtonovy metody se při výskytu násobného kořene iterační procs zpomaluje, je tomu u metody prosté iterace právě naopak.

14.4 Cvičení.

14.4.1

Stanovte aproximace prvních dvou prvních dvou kladných kořenů rovnice $\tan x = x$ metodou a) Newtonovou; b) sečen. Výpočet zastavte, bude-li $|x_k - x_{k-1}| < 10^{-6}$. Řešte tuto rovnici současně metodou prosté iterace. Posud'te problematiku konvergence metody. Počáteční aproximaci určete grafickou metodou.

14.4.2

Stanovte menší kladný kořen rovnic (s přesností 10^{-5}). Užijte metodu prosté iterace a stanovte odhad chyby. a) $e^x - 2(x-1)^2 = 0$; b) $e^{-x} - (x-1)^2 = 0$; c) $x^2 - \cos \pi x = 0$. [a) $0,213\,31$; b) $1,477\,68$; c) $0,438\,33$.]

14.4.3

Necht' X_k je interval se středem x_k. Užijte intervalové iterační formule (intervalové Newtonovy formule) $X_{k+1} = x_k - f(X_k)/f'(X_k)$ k určení intervalu délky 10^{-5}, v nemž leží kořen rovnice $e^x - 3x = 0$, vite-li, že interval $X_0 = \langle 0, 1\rangle$ kořen obsahuje.

14.4.4

Steffensenovou metodou: $x_{k+1} = x_k - f(x_k)/g(x_k)$, $g(x_k) = [f(x_k + f(x_k)) - f(x_k)]/f(x_k)$ stanovte reálný kořen rovnice $x^3 + 4x - 6 = 0$. [1, 134 729 .]

14.4.5

Je dán kruh K_1 o poloměru 1. Stanovte poloměr ϱ kruhu K_2 se středem A na obvodu kruhu K_1, tak aby společná část kruhů K_1 a K_2 měla obsah rovný polovině obsahu kruhu K_1. [Návod. Úhel $\measuredangle PAQ$, kde P, Q jsou průsečíky kružnic K_1 a K_2, označte x a obsah společné části kruhů K_1 a K_2 vyjádřete pomocí x a ϱ. Dostanete rovnici $\pi/2 = x(1 + \cos x) + (\pi - x) - \sin x$. Po úpravě $x \cos x = \sin x - \pi/2$ (viz cvič. 13.5.6). Tuto rovnici řešte Newtonovou metodou s přesností 10^{-7} (počáteční aproximaci určete odhadem ze smyslu úlohy).]

[$x \approx 1,905\,696$; $\varrho = \sqrt{2(1 + \cos x)} \approx 1,158\,728$.]

14.4.6

Jsou-li $x_{k-2}, xk - 1, x_k$ tři po sobě jdoucí aproximace jednoduchého kořene rovnice $x = \phi(x)$ získané konvergentní metodou prosté iterace, ukažte na příkladě, že posloupnost $\{y_k\}$ určená formulí (Aitkenova extrapolační formule)

$$y_k = x_k - \frac{(x_k - x_{k-1})^2}{x_k - 2x_{k-1} + x_{k-2}}$$

konverguje rychleji.

15 Řešení algebraických rovnic

Úloha stanovit kořeny algebraické rovnice

$$P(z) \equiv a_0 z^n + a_1 z^{n-1} + ... + a_{n-1} z + a_n = 0$$

se v numerické praxi vyskytuje natolik často, že to ospravedlňuje samostatný odstavec věnovaný této problematice. K řešení této úlohy lze ovšem použít metod, o lterých jsme hovořili v předcházejících odstavcích. Nicméně existuje ještě celá řada speciálních metod, z nichž se o některých stručně zmíníme. Podrobnější informace najde čtenář např. v [5], [12], [13]. Podle základní věty algebry má uvedená algebraická rovnice právě n kořenů $\alpha_1, \alpha_2, ..., \alpha_n$ (každý kořen počítáme tolikrát, kolik je jeho násobnost) a polynom P lze psát ve tvaru

$$P(z) = a_0(z - \alpha_1)(z - \alpha_2)...(z - \alpha_n).$$

Jestliže koeficienty $a_0, a_1, ..., a_n$ jsou reálné, potom eventuální komplexní kořeny se vyskytují v komplexně sdružených dvojicích. Všechny kořeny algebraické rovnice leží v mezikruží

$$r < |z| < R,$$

kde

$$R = 1 + \frac{1}{|a_0|} \max_{k=1,2,...,n} |a_k|;$$

$$\frac{1}{r} = 1 + \frac{1}{|a_n|} \max_{k=0,1,...,n-1} |a_k|.$$

Důkaz tohoto tvrzení najde čtenář např. v [5], kde jsou uvedeny i přesnější odhady polohy kořenů (ovšem už pouze pro reálné kořeny).

Při řešení algebraických rovnic se často ukazuje účelným využít znalost jednoho nebo více kořenů (resp. jejich aproximací) ke snížení stupně uvažovaného polynomu. V další fázi pak hledáme kořen (kořeny) rovnice nižšího stupně. Této problematice jsou věnovány následující dva odstavce.

15.1 Snížení stupně polynomu.

15.1.1 Dělení lineárním činitelem.

Je-li α_1 kořen rovnice $P(z) = 0$, kde P je polynom stupně n, potom další kořeny rovnice $P(z) = 0$, kde Q je polynom stupně $n-1$ vzniklý dělením polynomu P lineárním činitelem $z-\alpha_1$, a tedy platí [viz též (1.4.2)]

$$P(z) = (z - \alpha_1)Q(z)$$

Určíme-li kořen α_2 řešením rovnice $Q(z) = 0$, můžeme opět dělit činitelem $z-\alpha_2$, nyní ovšem polynom Q; dostaneme polynom R stupně $n-2$ takový, že

$$Q(z) = (z - \alpha_2)R(z).$$

V tomto procesu dělení můžeme pokračovat, až stanovíme všechny kořeny α_i, $i = 1, 2, ..., n$.

K určení koeficientů polynomů Q, R atd. používáme opakovaného Hornerova algoritmu (viz odst. 1.4.4).

Díky zaokrouhlovacím chybám a také díky tomu, že místo α_1 budeme mít k dispozici pouze nějakou aproximaci z_1 kořene α_1, budou při výpočtu koeficienty redukovaných polynomů, počínaje polynomem Q, zatíženy chybami a dobrá aproximace kořene redukovaného polynomu nemusí být stejně dobrou aproximací kořene původního polynomu. V učebnici [2] se uvádí, že dělení lineárními činiteli $z - z_k$ je numericky bezpečný proces, pokud dělíme postupně činiteli $z - z_1$, $z - z_2$ atd., kde

$$|z_1| \leqq |z_2| \leqq ... \leqq |z_{n-1}|.$$

V praxi toto pořadí podle rostoucí absolutní hodnoty kořenů často nedodržíme, proto se doporučuje aproximace z_2, z_3 atd. zlepšit iterováním přes původní polynom P. Pokud totiž dělíme v opačném pořadí (tj. začneme od kořene s největší absolutní hodnotou), můžeme dostat zcela pochybené výsledky.

15.1.2 Dělení kvadratickým činitelem.

Z algebry také víme (viz též odst. 1.4.3), že dělením polynomu

$$P(z) = a_0z^n + a_1z^{n-1} + ... + a_{n-1}z + a_n \tag{15.1.1}$$

kvadratickým polynomem

$$d(z) = z^2 - pz - q, \tag{15.1.2}$$

dostaneme podíl (polynom stupně $n-2$)

$$Q(z) = b_0z^{n-2} + b_1z^{n-3} + ... + b_{n-2}$$

a zbytek (lineární polynom)

$$\phi(z) = b_{n-1}z + b_n$$

a platí

$$P(z) = d(z)Q(z)+\phi(z), \tag{15.1.3}$$

K daným polynomům P, d jsou Q, ϕ uurčeny jednoznačně a lze je určit ze vztahu (15.1.3) porovnáním koeficientů u stejných mocnin z. Lehce se proto odvodí algoritmus výpočtu koeficientů polynomů Q a ϕ (analogie Hornerova algoritmu):

$$\begin{aligned}
&\text{Vstup}: \quad n, p, q, a_0, a_1, ..., a_n.\\
&\qquad b_0 = a_0.\\
&\qquad b_{-1} = 0.\\
&\qquad \text{Pro} \quad k = 1, 2, ..., n-1:\\
&\qquad\qquad b_k = a_k + pb_{k-1} + qb_{k-2}.\\
&\qquad b_n = a_n + qb_{n-2}.\\
&\text{Výstup}: \quad b_1, b_2, ..., b_{n-1}, b_n.
\end{aligned} \tag{15.1.4}$$

Vezmeme-li za polynom d součin kořenových činitelů $z-\alpha$, $z-\bar{\alpha}$, kde $\alpha = x - \mathrm{i}\,y$, $\bar{\alpha} = x - \mathrm{i}\,y$ jsou komplexně sdružené kořeny polynomu P s reálnými koeficienty, tj.

$$d(z) = (z-\alpha)(z-\bar{\alpha}) = z^2-2xz+x^2+y^2, \tag{15.1.5}$$

potom polynom ϕ ve vztahu (15.1.3) bude nulový.

K určení dalších kořenů polynomu P stačí řešit rovnici $Q(z) = 0$.

Známe-li pouze aproximace kořenů $\alpha, \bar{\alpha}$, potom dělením polynomu P trojčlenem, příslušným k těmto aproximacím, nedostaneme polynom ϕ nulový. Máme zde tedy obdobnou situaci jako v odst. 15.1.1.

V odstavci 15.5 bude popsána metoda postupného zpřesňování koeficientů kvadratického trojčlenu odpovídajícího aproximacím kořenů, tak abychom dostali kvadratický trojčlen určující součin kořenových činitelů přesných kořenů.

15.1.3 Příklad.

Číslo $\alpha = 0,330\,73 - 0,825\,12\mathrm{i}$ je aproximací kořene rovnice

$$z^3 - 9z^2 + 5z - 6 = 0$$

s odhadem chyby $3 \cdot 10^{-2}$. Dělením kvadratickým činitelem

$$(z-\alpha)(z-\bar{\alpha}) = z_2 - 0,661\,46z + 0,790\,20 \quad (p = 0,661\,46,\ \ q = -0,790\,20)$$

podle (15.1.3) dostáváme ve vztahu (15.1.2)

$$z^3 - 9z^2 + 5z - 6 = (z_2 - 0,661\,46z + 0,790\,20)(b_0 z + b_1) + b_2 z + b_3,$$

kde

$$\begin{aligned} b_0 &= a_0 = 1, \quad b_{-1} = 0, \\ b_1 &= a_1 + pb_0 + qb_{-1} = -8,338\,54, \\ b_2 &= a_2 + pb_1 + qb_0 = -1,305\,81, \\ b_3 &= a_3 + qb_1 = 0,589\,114, \end{aligned}$$

Aproximaci zbývajícího (reálného) kořene určíme z rovnice $B(z) \equiv b_0 z + b_1 = 0$. Bude to číslo $-b_1/b_0 = 8,338\,54$. Dá se ukázat, že odhad chyby této aproximace je $1,6 \cdot 10^{-1}$ (tedy horší než u kořene α). Značná odchylka od nuly u koeficientů b_2, b_3 svědčí o tom, že aproximace kořene α (a tedy i $\bar{\alpha}$) je dosti špatná.

15.2 Graeffova metoda.

15.2.1 Speciální případ.

Pro kořeny α_1, α_2 kvadratické rovnice

$$a_0 x^2 + a_1 x + a_2 = 0$$

platí vztahy

$$\alpha_1 + \alpha_2 = -\frac{a_1}{a_0}, \quad \alpha_1 \alpha_2 = \frac{a_2}{a_0}, \quad a_0 \neq o. \tag{15.2.1}$$

Jestliže absolutní hodnoty kořenů se od sobe dosti liší - např. $|\alpha_2/\alpha_1 \ll 1|$, potom v prvním ze vztahů (15.2.1), upraveném na tvar

$$\alpha_1 \left(1 + \frac{\alpha_2}{\alpha_1}\right) = -\frac{a_1}{a_0},$$

můžeme druhý sčítanec v závorce zanedbat a dostaneme aproximaci kořene

$$\alpha_1 \approx -\frac{a_1}{a_0}.$$

Po dosazení této aproximace do druhé rovnosti v (15.2.1) máme

$$\alpha_2 \approx -\frac{a_2}{a_1}.$$

Umocníme-li vztahy (15.2.1), dostaneme

$$\alpha_1^2 + \alpha_2^2 = \frac{a_1^2 - 2a_0a_2}{a_0^2}, \quad \alpha_1^2\alpha_2^2 = \frac{a_2^2}{a_0^2}.$$

Můžeme tedy sestavit kvadratickou rovnici

$$b_0y^2 + b_1y + b_2 = 0,$$

v níž

$$\begin{aligned} b_0 &= a_0^2, \\ b_1 &= -(a_1^2 - 2a_0a_2), \\ b_2 &= a_2^2, \end{aligned} \tag{15.2.2}$$

a pro jejíž kořeny β_1, β_2 platí

$$\beta_1 = \alpha_1^2, \quad \beta_2 = \alpha_2^2.$$

Analogicky můžeme určit aproximaci těchto kořenů z přibližných vztahů

$$\beta_1 \approx -\frac{b_1}{b_0}, \quad \beta_2 \approx -\frac{b_2}{b_1}$$

a také

$$|\alpha_1| \approx \sqrt{\left|\frac{b_1}{b_0}\right|}, \quad |\alpha_2| \approx \sqrt{\left|\frac{b_2}{b_1}\right|}. \tag{15.2.3}$$

Při určování těchto aproximací zanedbáváme ve vztahu

$$\alpha_1^2\left(1 + \frac{\alpha_2^2}{\alpha_1^2}\right) = -\frac{b_1}{b_0}$$

člen α_2^2/α_1^2, který je menší než $|\alpha_2/\alpha_1|$ (pokud $|\alpha_2/\alpha_1| < 1$), proto aproximace kořenů α_1, α_2 určené ve vztahu (15.2.3) budou lepší než aproximace určené pomocí koeficientů původní kvadratické rovnice.

O znaménku rozhodneme dosazením do původní kvadratické rovnice.

V uvedeném umocňování koeficientů rovnice můžeme ovšem pokračovat a dostávat tak stále lepší aproximace kořenů α_1, α_2.

Zobecnění předchozího postupu na polynom n-tého stupně je obsahem následujícího odstavce.

15.2.2 Případ reálných kořenů.

Chceme najít aproximace kořenů $\alpha_1, \alpha_2, ..., \alpha_n$ polynomu n-tého stupně

$$P(z) = a_0 z^n + a_1 z^{n-1} + ... + a_{n-1} z + a_n,$$

přičemž předpokládáme, že kořeny jsou reálné a máme je očíslovány podle klesající absolutní hodnoty, tj.

$$|\alpha_1| > |\alpha_2| > ... > |\alpha_{n-1}| > |\alpha_n|.$$

Definujeme posloupnost polynomů

$$P^{(0)}(z) \approx P(z), P^{(1)}(z), P^{(2)}(z), ..., P^{(k-1)}(z), P^{(k)}(z), ...$$

$$P^{(k)}(z) = a_0^{(k)} z^n + a_1^{(k)} z^{n-1} + ... + a_{n-1}^{(k)} z + a_n^{(k)}$$

postřednictvím vztahů pro koeficienty ($a_i^{(0)} = a_i,\ i = 0, 1, 2, ..., n$):

$$\begin{aligned}
a_0^{(k)} &= (a_0^{(k-1)})^2, \\
-a_1^{(k)} &= (a_1^{(k-1)})^2 - 2a_0^{(k-1)} a_2^{(k-1)}, \\
a_2^{(k)} &= (a_2^{(k-1)})^2 - 2a_1^{(k-1)} a_3^{(k-1)} + 2a_0^{(k-1)} a_4^{(k-1)}, \\
-a_3^{(k)} &= (a_3^{(k-1)})^2 - 2a_2^{(k-1)} a_4^{(k-1)} + 2a_1^{(k-1)} a_5^{(k-1)} - 2a_0^{(k-1)} a_0^{(k-1)}, \\
&\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \\
(-1)^{n-1} a_{n-1}^{(k)} &= (a_{n-1}^{(k-1)})^2 - 2a_{n-2}^{(k-1)} a_n^{(k-1)}, \\
(-1)^{n-1} a_n^{(k)} &= (a_n^{(k-1)})^2.
\end{aligned} \tag{15.2.4}$$

Zcela analogicky jako v (15.2.3) se dá odvodit (viz [12]), že

$$|a_j = \sqrt[2^k]{\left|\frac{a_j^{(k)}}{a_{j-1}^{(k)}}\right|}|, \quad j = 1, 2, ..., n. \tag{15.2.5}$$

Vztahy (15.2.4) jsou odvozeny z požadavku, aby kořeny polynomu $P^{(k)}$ byly druhými mocninami kořenů polynomu $P^{(k-1)}$.

Pomocí vztahu (15.2.5) vypúočteme absolutní hodnoty kořenů a_j. Pokud ani jedno ze dvou čísel $|a_j|$, $-|a_j|$ nedává uspokojivě malou hodnotu polynomu P, signalizuje to nesplnění předpokládu metody, tj. v rovnici $P(z) = 0$ se vyskytuji kořeny se stejnou hodnotou, včetně kořenů komplexních.

Při ručním výpočtu si do řádek pod sebe zapisujeme koeficienty polynomů P_0, P_1, P_2 atd. podle vztahů (15.2.4).

Protože pro rostoucí k koeficienty $a_j^{(k)}$ obvykle rychle rostou, doporučuje se provést občasnou normalizaci koeficientů, abychom se vyvarovali problému s přeplněním počítače.

15.2.3 Příklad

Užijme Graeffovy metody k určení kořenů rovnice

$$z^3 - 0z^2 + 5z - 6 = 0.$$

Počítejme v $M(10,5)$ (viz tab. 14). Podle (15.2.5) máme

$$|\alpha_1| \approx \sqrt[8]{\frac{2,710\,9 \cdot 10^7}{1}} \approx 8,494\,5;$$

dosazením do rovnice zjistíme, že $\alpha_1 \approx 8,494\,5$ a že ostatní podíly nám neposkytnou dobré výsledky. Skutečnost, že se ve sloupci $a_2^{(k)}$ střídají znaménka (nemusí se strídat pravidelně) signalizuje výskyt komplexních kořenů. Hornerovým algoritmem zjistíme (viz též odst. 15.1.1), že

$$z^3 - 9z^2 + 5z - 6 \approx (z - 8,494\,5)(z^2 - 0,505\,50z + 0,706\,03),$$

a tedy kořeny $\alpha_{2,3} \approx 0,252\,75 \pm 0,801\,34\mathrm{i}$ jsme vypočítali řešením kvadratické rovnice.

Tab. 14

15.2.4 Případ komplexních kořenů.

Maji-li některé kořeny algebraické rovnice stejnou absolutní hodnotu, nastanou při užití Graeffovy metody jisté komplikace, které však nejsou nezvládnutelné.

Omezíme se na případ, kdy dva kořeny α_i, α_{i+1} mají stejnou absolutní hodnotu . Předpokládáme tedy, že platí

$$|\alpha_1| > |\alpha_2| > ... > |\alpha_{i-1}| > |\alpha_i| = |\alpha_{i+1}| > |\alpha_{i+2}| > ... > |\alpha_n|$$

a že kořeny se stejnou absolutní hodnotou jsou komplexní, tj.

$$\begin{aligned} \alpha_i &= r(\cos\alpha + \mathrm{i}\,\sin\alpha), \\ \alpha_{i+1} &= r(\cos\alpha - \mathrm{i}\,\sin\alpha), \quad |\alpha_i| = |\alpha_{i+1}| = r. \end{aligned}$$

Obecnější případ je popsán v [12] nebo v [5]. Výskyt těchto komplexních kořenů je v posloupnosti koeficientů $a_i^{(0)}, a_i^{(1)}, a_i^{(2)}, ..., a_i^{(k)}$ ziskaných pomocí vztahů (15.2.4) signalizován nepravidelnostmi ve znaménku (viz sloupce $a_2^{(k)}$ v příkl. 15.2.3).

Absolutní hodnotu r komplexních kořenů α_i, α_{i+1} určíme ze vztahu

$$r^2 \approx \sqrt[2^k]{\left|\frac{a_{i+1}^{(k)}}{a_{i-1}^{(k)}}\right|}.^{17}) \tag{15.2.6}$$

Protože

$$\alpha_i + \alpha_{i+1} = 2r\cos\alpha,$$

Potom z vlastnosti kořenů rovnice $P(z) = 0$

$$\alpha_1 + \alpha_2 + ... + \alpha_{i-1} + a_i + a_{i+1} + ... + a_n = -\frac{a_1^{(0)}}{a_0^{(0)}}$$

máme

$$2r\cos\alpha = -\frac{a_1^{(0)}}{a_0^{(0)}} - \alpha_1 - \alpha_2 - ... - \alpha_{i-1} - \alpha_{i+2} - ... - \alpha_n. \tag{15.2.7}$$

Aproximace kořenů α_i, α_{i+1} určíme řešením kvadratické rovnice

$$\xi^2 - 2r(\cos\alpha)\xi + r^2 = 0. \tag{15.2.8}$$

Jestliže znaménkové nepravidelnosti se projeví ve více sloupcích sestavených z koeficientů $a_i^{(k)}$ $k = 1, 2, ..., n$ na místě úvaha o vhodnosti Graeffovy metody k dané rovnici.

15.2.5 Příklad.

Chceme stanovit aproximace všech kořenů rovnice

$$z^4 - 4z^3 + 3z^2 + 2z - 6 = 0.$$

Počítáme v $M(10, 10)$ a výsledky zapíšeme do tabulky (tab. 15). Anomálii vykazuje sloupec $a_2^{(k)}$. Očekáváme proto, že kořeny α_2, α_3 budou komplexní

Tab. 15

[17]) Také pro kořeny s různými absolutními hodnotami podle (15.2.5) platí

$$|\alpha_j||\alpha_{j+1}| \approx \sqrt[2^k]{\left|\frac{a_j^{(k)}}{a_{j-1}^{(k)}}\right|}\sqrt[2^k]{\left|\frac{a_{j+1}^{(k)}}{a_j^{(k)}}\right|} = \sqrt[2^k]{\left|\frac{a_{j+1}^{(k)}}{a_{j-1}^{(k)}}\right|}.$$

Podle (15.2.5) nejdříve vypočteme

$$|\alpha_i| \approx \sqrt[8]{\left|\frac{a_1^{(3)}}{a_0^{(3)}}\right|} = \sqrt[8]{\frac{6\,594}{1}} \approx 3,001\,882\,007.$$

Protože $P^{(0)}(3,001\,882\,007) \approx 0,037\,7$, $P^{(0)}(-3,001\,882\,007) \approx 204,44$, usoudíme, že

$$\alpha_1 \approx 3,001\,882\,007.$$

Analogicky

$$|\alpha_4| \approx \sqrt[8]{\left|\frac{a_4^{(3)}}{a_3^{(3)}}\right|} = \sqrt[8]{\frac{1\,679\,616}{1\,889\,824}} \approx 0,985\,368\,285\,8.$$

Protože

$$P^{(0)}(0,985\,368\,285\,8) \approx -4,000\,6, \quad P^{(0)}(-0,985\,368\,285\,8) \approx -0,288,$$

usoudíme, že

$$\alpha_4 \approx -0,985\,368\,285\,8.$$

Zbývající podíly typu (15.2.5) nám nedají dobré výsledky, proto užijeme (15.2.6):

$$r^2 \approx \sqrt[8]{\left|\frac{a_3^{(3)}}{a_1^{(3)}}\right|} = \sqrt[8]{\frac{1\,889\,824}{6\,594}} \approx 2,028\,425\,455 = |\alpha_2\alpha_3|.$$

Dále podle (15.2.7) dostaneme

$$2r\cos\alpha = 4 - \alpha_1 - \alpha_4 = 1,983\,486\,278.$$

Kořeny

$$\alpha_2 = r(\cos\alpha + \mathrm{i}\,\sin\alpha),$$
$$\alpha_3 = r(\cos\alpha - \mathrm{i}\,\sin\alpha)$$

určíme řešením kvadratické rovnice (15.2.8)

$$\xi^2 - 1,983\,486\,278\xi + 2,028\,425\,455 = 0,$$

tj.

$$\alpha_{2,3} \approx 0,991\,743\,139\,0 \pm 1,022\,189\,317\mathrm{i}.$$

Poznamenejme, že přesné hodnoty kořenů jsou: 3, $1+\mathrm{i}$, $1-\mathrm{i}$, 1.

15.3 Laguerrova metoda.

Tato metody je zvlášt' učinná pro řešení algebraických rovnic, jejichž všechny kořeny jsou reálné a jednoduché. Pro takové rovnice metoda rychle konverguje a to nezávisle na (reálné) počáteční aproximaci x_0. Proto lze tuto metodu zařadit mezi startovaci i zpřesňující metody. Pro vícenásobné kořeny se konvergence zpomalí v okolí násobného kořene. pro stanovení komplexních kořenů se této metody dá také užít, i když se vtomto případě o konvergenci mnoho nevi ([12]). Zkušenosti ukazuji, že pradvěpodobnost výskytu divergentního proces je velive malá.

Laguerrova metoda je určena iterační formulí

$$x_{k+1} = x_k - \frac{nP(x_k)}{P(x_k) \pm \sqrt{H(x_k)}}, \tag{15.3.1}$$

kde n je stupeň polynomu a

$$H(x_k) = (n-1)[(n-1)(P'(x_k))^2 - nP(x_k)P''(x_k)].$$

Jsou-li všechny kořeny reálné, dá se ukázat, že H je stále nezáporná funkce; znaménko u odmocniny vybereme rovné $\operatorname{sgn} P(x_k)$. U komplexních kořenů vybereme znyménko tak, aby jmenovatel měl co největší absolutní hodnotu.

Jsou-li reálné kořeny seřyzeny tak, že

$$\alpha_1 \leqq \alpha_2 \leqq \alpha_3 \leqq ... \leqq \alpha_n,$$

pak když bude $x_0 \in (\alpha_{j-1}, \alpha_j)$, $j = 1, 2, ..., n$ konverguje proces (15.3.1) k jednomu z kořenů α_{j-1}, α_j; když bude $x_o < \alpha_1$, potom (15.3.1) konverguje k α_1; je-li $x_0 > \alpha_n$, potom konverguje k α_n.

15.4 Newtonova metody a metoda sečen pro polynomy.

Uved'me si zde pouze algoritmy, nebot' výklad těchto metod byl proveden v odstavcích 14.1, 14.2.1.

Uvedeme algoritmus Newtonovy metody, jehož základem je opakovaný Hornerův algoritmus (viz odst. 1.4.4) pro reálné z_0:

(15.4.1)

Vstup : $n, \delta, z_0, a_0, a_1, ..., a_n.$

$b_0 = a_0.$

$c_0 = a_0.$

Pro $k = 0, 1, 2, ..., (m)$:

  Pro $j = 1, 2, ..., n$:

    $b_j = z_k b_{j-1} + a_j.$

  Pro $s = 1, 2, ..., n-1$:

    $c_s = z_k c_{s-1} + b_s.$

  $h_k = -\dfrac{b_n}{c_{n-1}}.$

  $z_{k+1} = z_k + h_k.$

Výstup : $z_m.$

Číslo m neurčujeme předem, ale výpočet ukončíme, bude-li splněna podmínka $|h_k| = |z_{k+1} - z_k| < \delta$.

Je-li aproximace $z_0 = x_0 + \mathrm{i}\, y_0$ kořene rovnice $P(z) = 0$ s reálnými koeficienty komplexní (získaná např. Graeffovou metodou), lze algoritmus (15.4.1) upravit tak, abychom počítali pouze s reálnými čísly.

Označíme proto

$$\begin{aligned} z_k &= x_k + \mathrm{i}\, y_k, \qquad k = 0, 1, 2, ..., \\ b_j &= \beta_j + \mathrm{i}\, \delta_j, \qquad j = 1, 2, ..., n, \\ c_s &= \gamma_s + \mathrm{i}\, \eta_s, \quad s = 1, 2, ..., n-1. \end{aligned}$$

Rozepíšeme-li všechny operace v algoritmu (15.4.1) v komplexní aritmetice na základě předchozího označení, dostaneme:

(15.4.2)

$$
\begin{aligned}
&\text{Vstup :} \quad n, \delta, x_0, y_0, a_0, a_1, ..., a_n.\\
&\qquad \beta_0 = a_0, \quad \gamma_0 = a_0, \quad \delta_0 < 0, \quad \eta_0 = 0.\\
&\qquad \text{Pro} \quad k = 0, 1, 2, ..., (m) :\\
&\qquad\qquad \text{Pro} \quad j = 1, 2, ..., n :\\
&\qquad\qquad\qquad \beta_j = a_j + x_k\beta_{j-1} - y_k\delta_{j-1},\\
&\qquad\qquad\qquad \delta_j = y_k\beta_{j-1} + x_k\delta_{j-1}.\\
&\qquad\qquad \text{Pro} \quad s = 1, 2, ..., n-1 :\\
&\qquad\qquad\qquad \gamma_s = \beta_s + x_k\gamma_{s-1} - y_k\eta_{s-1},\\
&\qquad\qquad\qquad \eta_s = \delta_j + y_k\gamma_{j-1} + x_k\eta_{j-1},\\
&\qquad\qquad x_{k+1} = x_k - \frac{\beta_n\gamma_{n-1} + \delta_n\eta_{n-1}}{\gamma_{n-1}^2 + \eta_{n-1}^2},\\
&\qquad\qquad y_{k+1} = y_k - \frac{\delta_n\gamma_{n-1} + \beta_n\eta_{n-1}}{\gamma_{n-1}^2 - \eta_{n-1}^2},\\
&\text{Výstup :} \quad x_m, y_m.
\end{aligned}
$$

Proces zas¨tavíme, bude-li

$$|z_{k+1} - z_k| = \sqrt{(x_{k+1} - x_k)^2 + (y_{k+1} - y_k)^2} < \delta.$$

Z uvedeného algoritmu je patrně, že k určení komplexního kořene musíme volit komplexní počáteční aproximací.

Algoritmus metody sečen se od (15.4.1) bude lišit tím, že v tomto případě je

$$h_k = -\frac{b_n(z_k - z_{k-1})}{b_n - \tilde{b}_n}, \tag{15.4.3}$$

kde $\tilde{b}_n = P(z_{k-1})$ (vypočítává se v předcházejícím kroku algoritmu) a ve vstupních datech musí být ještě z_1 .

15.5 Bairstowova metoda.

Předpokládáme, že

$$d(z) = z^2 - pz - q = (z - z_1)(z - z_2)$$

je aproximace kvadratického trojčlenu

$$d^*(z) = z^2 - p^*z - q^* = (z - \alpha)(z - \bar{\alpha}),$$

kde $\alpha, \bar{\alpha}$ jsou komplexně sdružené kořeny rovnice s reálnými koeficienty

$$P(z) \equiv a_0 z^n + a_1 z^{n-1} + ... + a_{n-1} z + a_n = 0, \quad n \geqq 4.$$

Čísla z_1, z_2 proto aproximují kořeny $\alpha, \bar{\alpha}$

Dělením polynomu P trojčlenem d [algoritmus (15.1.4)] dostaneme polynom stupně $n-2$ (viz odst. 15.1.2)

$$Q(z) = b_0 z^{n-2} + b_1 z^{n-3} + ... + b_{n-2}$$

a zbytek

$$\phi(z) = b_{n-1} z + b_n,$$

přičemž platí

$$P(z) = d(z)Q(z) + \phi(z). \tag{15.5.1}$$

Koeficienty b_{n-1}, b_n lineárního polynomu d^* závisejí na parametrech p, q což zapíšeme ve tvaru

$$\begin{aligned} b_{n-1} &= f(p,q), \\ b_n &= g(p,q). \end{aligned} \tag{15.5.2}$$

Protože polynom P je dělitelný polynomem d^*, budou koeficienty b_{n-1}, b_n pro $p = p^*$, $q = q^*$ nulové, tj.

$$\begin{aligned} f(p^*, q^*) &= 0, \\ g(p^*, q^*) &= 0. \end{aligned}$$

Tedy čísla p^*, q^* můžeme považovat za řešení soustavy (nelineárních) rovnic

$$\begin{aligned} f(p,q) &= 0, \\ g(p,q) &= 0. \end{aligned} \tag{15.5.3}$$

Tuto soustavu převedeme Newtonovu metodu (viz odst. 16.3) na posloupnost lineárních rovnic typu

$$\begin{pmatrix} \frac{\partial f(p,q)}{\partial p} & \frac{\partial f(p,q)}{\partial q} \\ \frac{\partial g(p,q)}{\partial p} & \frac{\partial g(p,q)}{\partial q} \end{pmatrix} \begin{pmatrix} \triangle p \\ \triangle q \end{pmatrix} = \begin{pmatrix} -f(p,q) \\ -g(p,q) \end{pmatrix}, \tag{15.5.4}$$

kde $\triangle p, \triangle q$ jsou opravy parametrů p, q.

Abychom mohli stanovit derivace funkcí f, g v soustavě (15.5.4), budeme polynom Q určený vztahem (15.5.1) dělit tak, aby

$$Q(z) = d(z)R(z) + \xi(z), \tag{15.5.5}$$

kde

$$R(z) = c_0 z^{n-4} + c_1 z^{n-5} + ... + c_{n-5} z + c_{n-4},$$
$$\psi(z) = c_{n-3} z + c_{n-2}.$$

Dosazením do vztahu (15.5.1) dostáváme

$$P(z) = d(z)[d(z)R(z)+\psi(z)]+\phi(z) = d^2(z)R(z)+d(z)(c_{n-3}z+c_{n-2})+zf+g$$

Derivujeme tuto rovnost [$P(z)$ na p, q nezávisí] podle p a q

$$0 = -2d(z)zR(z) - z(c_{n-3}z + c_{n-2}) + d(z)\frac{\partial}{\partial q}(c_{n-3}z + c_{n-2}) + z\frac{\partial f}{\partial p} + \frac{\partial g}{\partial p},$$
$$0 = -2d(z)R(z) - (c_{n-3}z + c_{n-2}) + d(z)\frac{\partial}{\partial q}(c_{n-3}z + c_{n-2}) + z\frac{\partial f}{\partial q} + \frac{\partial g}{\partial q}.$$

Protože $d(z_i) = 0\,,\ i = 1, 2\,,$ máme odtud (po dosazení $z_i^2 = pz_i + q\,$)

$$z_i\left(\frac{\partial f}{\partial p} - c_{n-2} - pc_{n-3}\right) + \left(\frac{\partial g}{\partial q} - qc_{n-3}\right) = 0,$$

$$z_i\left(\frac{\partial f}{\partial p} - c_{n-3}\right) + \left(\frac{\partial g}{\partial q} - c_{n-2}\right) = 0.$$

Pro $z_1 \neq z_2$ lte tuto vztahy současně splnit pouze, když výrazy v závorkách jsou nulové. Soustavu (15.5.4) proto můžeme psát ve tvaru

$$\begin{pmatrix} pc_{n-3} + c_{n-2} & c_{n-3} \\ qc_{n-3} & c_{n-2} \end{pmatrix} \begin{pmatrix} \triangle p \\ \triangle q \end{pmatrix} = \begin{pmatrix} -b_{n-1} \\ -b_n \end{pmatrix}. \tag{15.5.6}$$

Tedy

$$\triangle p = \frac{c_{n-3}b_n - c_{n-2}b_{n-1}}{N}, \quad \triangle q = \frac{c_{n-3}W - c_{n-2}b_n}{N},$$

kde jsme označili

$$W = qb_{n-1} - pb_n, \quad M = pc_{n-2} - qc_{n-3}, \quad N = c_{n-3}M - c_{n-2}^2.$$

15.5.1 Algoritmus Bairstowovy metody.

Vstup : $n\,(\geqq 4), p, q, a_0, a_1, ..., a_n, \delta.$

$b_0 = c_0 = a_0.$

$b_{-1} = c_{-1} = 0.$

Pro $k = 1, 2, ..., n-1$:

$b_k = a_k + pb_{k-1} + qb_{k-2}.$

$b_n = a_n + qb_{n-2}.$

Pro $j = 1, 2, ..., n-3$:

$c_j = b_j + pc_{j-1} + qc_{k-2}.$

$c_{n-2} = b_{n-2} + qc_{n-4}.$

$M = pc_{n-2} - qc_{n-3}.$

$W = qb_{n-1} - pb_n.$

$N = c_{n-2}^2 + Mc_{n-3}.$

$\triangle p = (c_{n-3}b_n - c_{n-2}b_{n-1})/N.$

$\triangle q = (c_{n-3}W - c_{n-2}b_n)/N.$

Výstup : $p + \triangle p, q + \triangle q.$

Ve výpočtu podle uvedeného algoritmu můžeme pokračovat přeznačením $p + \triangle p \to p\,,\ q + \triangle q \to q\,.$ Výpočet zastavíme např. podmínkou: $\max(|\triangle p|, |\triangle q|) < \delta\,.$

15.5.2 Příklad.

V příkladu 15.2.5 jsme zjistili, že kvadratický trojčlen $z^2 - 1,983\,486\,278z + 2,028\,425\,455$ je aproximací trojčlenu odpovídajícího komplexním kořenům $(1 + \mathrm{i}, 1 - \mathrm{i})$ rovnice

$$z^3 - 4z^3 + 3z^2 + 2z - 6 = 0.$$

Užijeme algoritmu z odst. 15.5.1 k opravě parametru $p, q\,.$ Vstupní parametry:

$$n = 4; \quad p = 1,983\,486\,278; \quad q = -2,028\,425\,455;$$

$$a_0 = 1, \quad a_1 = -4, \quad a_2 = 3, \quad a_3 = 2, \quad a_4 = -6.$$

Dále

$$
\begin{aligned}
&b_0 = c_0 = 1, \quad b_{-1} = c_{-1} = 0,\\
&b_1 = a_1 + pb_0 + qb_{-1} = -2,016\,513\,722,\\
&b_2 = a_2 + pb_1 + qb_0 = -3,028\,152\,751,\\
&b_3 = a_3 + pb_2 + qb_1 = 0,084\,048\,335\,00,\\
&b_4 = a_4 + qb_2 = 0,142\,382\,121\,0,\\
&c_1 = b_1 + pc_0 + qc_{-1} = -0,033\,027\,444\,00,\\
&c_2 = b_2 + qc_0 = -5,056\,578\,206,\\
&M = pc_2 - qc_1 = -10,096\,647\,18,\\
&W = qb_3 - pb_4 = -0,452\,898\,765\,3,\\
&N = c_2^2 + Mc_1 = 25,902\,449\,59,\\
&\triangle p = (c_1b_4 - c_2b_3)/N = 1,658\,914\,516 \cdot 10^{-2},\\
&\triangle q = (c_1W - c_2b_4)/N = 2,837\,277\,67 \cdot 10^{-2}.
\end{aligned}
$$

Opravený kvadratický trojčlen má tvar

$$z^2 - 2,000\,075\,423z + 2,000\,052\,679.$$

15.6 Podmíněnost

V souvislosti s řešením úlohy najít kořen rovnice $f(x) = 0$ na intervalu $\langle a, b\rangle$ nás zajímá citlivost vypočteného kořene na změně funkce f (vstupní údaj). Jde nám tedy o posouzení podmíněnosti dané úlohy.

Změníme-li funkci f o $\triangle f$, bude číslo $\alpha + \triangle\alpha$ kořenem této změněné rovnice [α je kořen rovnice $f(x) = 0$], tj.

$$f(\alpha+\triangle\alpha)+\triangle f(\alpha+\triangle\alpha) = 0. \tag{15.6.1}$$

Z Taylorova rozvoje máme $f(\alpha + \triangle\alpha) = f(\alpha) + f'(\alpha) \triangle \alpha + O((\triangle\alpha)^2)$, $\triangle f(\alpha + \triangle\alpha) = \triangle f(\alpha) + O((\triangle\alpha)^2)$ a rovnici (15.6.1) nahradíme rovnicí

$$f(\alpha)+f'(\alpha)\triangle\alpha+\triangle f(\alpha) = O((\triangle\alpha)^2) \tag{15.6.2}$$

Odtud

$$\triangle\alpha \approx -\frac{\triangle f(\alpha)}{f'(\alpha)}.$$

Z velikosti $|f'(\alpha)|$ usuzujeme napodmíněnost. Při malém $|f'(\alpha)|$ může malá změna ve vstupních datec funkce f způsobit velkou změnu ve vypočteném kořenu. ==== Na obr. 13 a 14 vidíme průběhy funkce $f = f(x)$, resp.

$f + \triangle f$, pro něž je uvažovaná úlphy špatně, resp. extrémně špatně podmíněná. Je-li f polynom, pak v této souvislosti hovoříme o *dobře* či *špatně podmíněných polynomech*. Ve [12] je uváděn následující příklad: Řešíme-li místo rovnice

$$(z-1)(z-2)...(z-19)(z-20) = 0$$

rovnici

$$(z-1)(z-2)...(z-19)(z-20) - 2^{-23}z^{19} = 0$$

(změnili jsme pouze koeficient u z^{19} o $2^{-23} \approx 10^{-7}$), dostaneme (napsané číslice jsou platné): $1,0; 2,0; ..., 8,0; 8,9; 10,1 \pm 0,6\mathrm{i}; 14,0 \pm 2,5\mathrm{i}; 16,7 \pm$ $\pm 2,8\mathrm{i}; 19,5 \pm 1,9\mathrm{i}; 20,8$.

Užití dvojnásobné nebo vícenásobné aritmetiky při řešení algebraických rovnic vyšších stupňů je většinou nezbytné k dosažení přesnějších výsledků.

Při řešení takových rovnic můžeme narazit na další těžkosti. Např. může dojít při výpočtu funkčních hodnot k přeplnění počítače.

15.7 Cvičení.

15.7.1

Graeffovou metodou řešte rovnici $x^4 + x^3 - 10x^2 - 34x - 26 = 0$.

[$\alpha_1 \approx 4,014\,709\,485; \alpha_{2,3} \approx 1,936\,384\,71 \pm 2,772\,454\,98\mathrm{i}; \alpha_4 \approx -1,141\,940\,065$.]

15.7.2

Stanovte kořeny rovnice $x^4 - 6,79x^3 + 2,995x^2 - 0,043\,69x + 0,000\,089\,25 = 0$. Počítejte v $M(10,4)$. [Návod. Určete největší kořen, snižte stupeň rovnice a zjistěte, jak se kořeny redukovaného polynomu liší od kořenů původního polynomu.]

[$6,326; 0,457\,3; 0,012\,58; 0,002\,453$; redukovaný polynom $x^3 - 0,472\,0x^2 +$ $+ 0,009\,128x + 0,014\,05$ má kořeny $-0,143\,7; 0,307\,9 \pm 0,503\,30\mathrm{i}$.]

15.7.3

Vypočtěte dvě iterace Laguerrovy metody k určení aproximace komplexního kořene rovnice $z^3 - 9z^2 + 5z - 6 = 0$. Volte $z_0 = 0$ a počítejte v $M(10,5)$.

[$z_2 = 0,330\,73 - 0,825\,12\mathrm{i}$.]

15.7.4

Vypočtěte všechny kořeny rovnice $z^3 - 9z^2 + 5z - 6 = 0$.
$[8,494\,5; 0,252\,73 \pm 0,801\,53\mathrm{i}\,.]$

16 Soustavy nelineárních rovnic

16.1 Úvodní poznámky.

Soustavu n nelineárních rovnic pro n neznámých

$$\begin{aligned} F_1(x_1, x_2, ..., x_n) &= 0, \\ F_2(x_1, x_2, ..., x_n) &= 0, \\ &\cdots\cdots\cdots\cdots \\ F_n(x_1, x_2, ..., x_n) &= 0, \end{aligned} \tag{16.1.1}$$

můžeme psát vevektorové symbolice jako

$$\mathbf{F}(\mathbf{x}) = \mathbf{0}, \tag{16.1.2}$$

kde je označeno $\mathbf{F} = (F_1, F_2, ..., F_n)^T$; $\mathbf{x} = (x_1, x_2, ..., x_n)^T$. Chceme najít *řešení* této *soustavy*, tj. n-tici reálných čísel (vektor) $(x_1^*, x_2^*, ..., x_n^*) = \mathbf{x}^*$ takovou, že $\mathbf{F}(\mathbf{x}^*) = \mathbf{0}$.

Dříve, než uvedeme některé metody řešení rovnice $\mathbf{F}(\mathbf{x}) = \mathbf{0}$, je třeba si uvědomit, že tato nemusí být vždy v oboru reálných čísel řešitelná. Například soustava

$$\begin{aligned} x^2 - y + a &= 0, \\ -x + y^2 + a &= 0 \end{aligned} \tag{16.1.3}$$

pro $a = 1$ nemá řešení, pro $a = \frac{1}{4}$ má jediné řešení $(0,5; 0,5)^T$, pro $a = 1$ má dvě řešení $(0,0)^T$, $(1,1)^T$ a pto $a = -1$ má čtyři řešení $(-1,0)^T$, $(0,1)^T$, $((1+\sqrt{5})/2)^T$, $((1-\sqrt{5})/2)^T$. Doporučujeme čtenáři, aby si nakreslil grafy křívek reprezentovaných rovnicemi (16.1.3).

Na rozdíl od soustav lineárních rovnic se přímé metody u nelineárních soustav dají užít pouze pro některé speciální soustavy a jen tehdy, když počet neznámých není příliš velký. Proto budeme věnovat pozornost iteračním metodám. Podáme zde ovšem pouze základní informaci. Podrobněji se čtenář může s těmito problémy seznámit např. v [1], [5] a především v knize [7].

16.2 Metoda prosté iterace.

Rovnici $\mathbf{F}(\mathbf{x}) = \mathbf{0}$ přepíšeme (srov. s ost. 13.3) vhodným způsobem na tvar $\mathbf{x} = \mathbf{\Phi}(\mathbf{x})$ tak, aby $\mathbf{x}^*$ byl společným řešením obou těchto rovnic. Předpokládáme samozřejmě, že $\mathbf{F}$ je spojité zobrazení.

Zvolíme $\mathbf{x}^{(0)}$ a počítáme aproximace $\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, ...$ vektoru $\mathbf{x}^*$ z iterační formule

$$\mathbf{x}^{(k)} = \mathbf{\Phi}(\mathbf{x}^{(k-1)}), \quad k = 1, 2, 3, \dots . \tag{16.2.1}$$

pokud v Ω_0 platí [srov. s podmínkami (13.3.3)]

$$\begin{aligned} &(a) \quad \forall \mathbf{x} \in \Omega_0 : \quad \mathbf{\Phi}(\mathbf{x}) \in \Omega_0, \\ &(b) \quad \exists q \in \langle 0, 1) : \quad \|\mathbf{\Phi}(\mathbf{x}) - \mathbf{\Phi}(\mathbf{y})\| \leqq q \, \|\mathbf{x} - \mathbf{y}\| \quad \forall \mathbf{x}, \mathbf{y} \in \Omega_0, \end{aligned} \tag{16.2.2}$$

potom posloupnost $\left\{\mathbf{x}^{(0)}\right\}$ určená rekurentním vztahem (16.2.1) konverguje k $\mathbf{x}^* \in \Omega_0$ pro libovolnou počáteční aproximaci $\mathbf{x}^{(0)} \in \Omega_0$ a platí odhad pro chybu metody prosté iterace

$$\left\|\mathbf{x}^{(k)} - \mathbf{x}^*\right\| \leqq \frac{q}{1-q} \left\|\mathbf{x}^{(k)} - \mathbf{x}^{(k-1)}\right\| . \tag{16.2.3}$$

Symbolem $\|\cdot\|$ značíme vektorovou normu zavedenou v odst. 5.2.

Protože $\mathbf{\Phi}(\mathbf{x}) = (\phi_1(\mathbf{x}), \phi_2(\mathbf{x}), ..., \phi_n(\mathbf{x}))^T$, lze pro diferencovatelné funkce $\phi_i(\mathbf{x})$, $i = 1, 2, ..., n$, podmínku (16.2.2) (b) nahradit podmínkou

$$(b') \quad \exists q : \quad \|\mathbf{\Phi}'(\mathbf{x})\| \leqq q < 1 \quad \forall \mathbf{x} \in \Omega_0,$$

kde $\mathbf{\Phi}'(\mathbf{x})$ je matice s prvky $\partial\phi_i(x_1, x_2, ..., x_n)/\partial x_j$. V podmínce (b') maticová norma musí odpovídat příslušné vektorové normě.

Výpočty podle (16.2.1) jsou jednoduché. Ovšem je mnohem obtížnější najít takovou rovnici $\mathbf{x} = \mathbf{\Phi}(\mathbf{x})$, která by byla ekvivalentní s výchozí rovnici $\mathbf{F}(\mathbf{x}) = \mathbf{0}$ (alespoň v nějakém Ω_0) a současně taková, aby zajistila konvergenci procesu (16.2.1).

16.2.1 Příklad.

Řešme soustavu $F_i(x, y) \equiv 2x^2 - xy - 5x + 1 = 0$, $F_i(x, y) \equiv x + 3\ln x - y^2 = 0$.

Danou soustavu přepíšeme na tvar

$$\begin{aligned} x &= \sqrt{[x(5+y) - 1]/2} \equiv \phi_1(x, y), \\ y &= \sqrt{x + 3\ln x} \equiv \phi_2(x, y). \end{aligned}$$

Tyto dvě soustavy jsou ekvivalentní v okolí bodu $(x_0, y_0)^T = (3,8; 2,8)^T$, který najdeme jako průsečík křivek daných soustavou. Srovnáním s formulí (16.2.1) je:

$$\mathbf{x}^{(k)} = \begin{pmatrix} x_k \\ y_k \end{pmatrix}, \quad \begin{pmatrix} \phi_1(x_{k-1}, y_{k-1}) \\ \phi_2(x_{k-1}, y_{k-1}) \end{pmatrix} = \mathbf{\Phi}(\mathbf{x}^{(k-1)}).$$

Vypočteme [v $M(10,4)$]:

$$\begin{array}{ll} ll \quad x_1 = \sqrt{0,5}[3,8(5+2,8)-1] = 3,784; & y_1 = \sqrt{3,8} + 3\ln 3,8 = 2,794; \\ x_2 = 3,774, & y_2 = 2,789, \\ x_3 = 3,768, & y_3 = 2,785, \\ \ldots\ldots\ldots\ldots & \ldots\ldots\ldots\ldots \\ x_9 = 3,757, & y_9 = 2,780. \end{array}$$

Iterační proces jsme ukončili podmínkou

$$\left\|\mathbf{x}^{(k)} - \mathbf{x}^{(k-1)}\right\|_{\mathbf{R}} = \max(|x_k - x_{k-1}|, |y_k - y_{k-1}|) < 10^{-3}.$$

Tento proces skutečně konverguje, nebot' v bodě $\mathbf{x}^{(0)}$ (a tedy i v jeho malém okolí obsahujícím ovšem $\mathbf{x}^*$) je

$$\left|\frac{\partial\phi_1(\mathbf{x}^{(0)})}{\partial x}\right| \approx 0,52; \quad \left|\frac{\partial\phi_1(\mathbf{x}^{(0)})}{\partial y}\right| \approx 0,25; \quad \left|\frac{\partial\phi_2(\mathbf{x}^{(0)})}{\partial x}\right| \approx 0,32;$$

$$\left|\frac{\partial\phi_2(\mathbf{x}^{(0)})}{\partial y}\right| \approx 0,$$

a tedy

$$\left\|\mathbf{\Phi}'(\mathbf{x}^{(0)})\right\|_{\mathbf{R}} = \max(0,52 + 0,25; 0,32 + 0) = 0,77 < 1.$$

Z předchozích výpočtů soudíme, že i podmínka (16.2.2) (a) je splněna.

Pro odhad chyby podle (16.2.3) platí

$$\left\|\mathbf{x}^{(9)} - \mathbf{x}^*\right\|_{\mathbf{R}} \leqq \frac{0,77}{1-0,77} \cdot 10^{-3} < 3,5 \cdot 10^{-3}.$$

Protože [v $M(10,8)$] je $\mathbf{x}^* = (3,756\,833\,5; 2,779\,849\,1)^T$, pak ve skutečnosti je

$$\left\|\mathbf{x}^{(9)} - \mathbf{x}^*\right\|_{\mathbf{R}} \leqq 1,7 \cdot 10^{-4}.$$

16.3 Newtonova metoda.

Necht' $\mathbf{x}^{(k)} = (x_k, y_k)^T$ aproximace řešení $\mathbf{x}^* = (x^*, y^*)^T$ rovnice

$$\mathbf{F}(\mathbf{x}) = \begin{pmatrix} F_1(x,y) \\ F_2(x,y) \end{pmatrix} = \mathbf{0}. \tag{16.3.1}$$

Předpokládáme, že funkce F_1, F_2 jsou diferencovatelné v okolí $\mathbf{x}^*$ a můžeme je tedy vyjádřit ve tvaru Taylorova rozvoje. Zanedbáme-li členy vyšších řádů, dostaneme

$$F_i(x,y) \approx F_i(x_k, y_k) + \frac{\partial F_i(x_k,y_k)}{\partial x}(x - x_k) + \frac{\partial F_i(x_k,y_k)}{\partial y}(y - y_k), \quad i = 1, 2.$$

Aproximujeme tedy funkce F_i lineární funkcí. Veltorově lze psát

$$\mathbf{F}(\mathbf{x}) \approx \mathbf{F}(\mathbf{x}^{(k)}) + \mathbf{F}'(\mathbf{x}^{(k)})(\mathbf{x} - \mathbf{x}^{(k)}), \tag{16.3.2}$$

kde jsme označili

$$\mathbf{F}'(\mathbf{x}^{(k)}) = \begin{pmatrix} \frac{\partial F_1(x_k,y_k)}{\partial x} & \frac{\partial F_1(x_k,y_k)}{\partial y} \\ \frac{\partial F_2(x_k,y_k)}{\partial x} & \frac{\partial F_2(x_k,y_k)}{\partial y} \end{pmatrix}, \quad \mathbf{x} - \mathbf{x}^{(k)} = \begin{pmatrix} x - x_k \\ y - y_k \end{pmatrix}.$$

Matici $\mathbf{F}'$ nazýváme *Jacobiovu maticí* funkce $\mathbf{F}$.

Místo rovnice $\mathbf{F}(\mathbf{x}) = \mathbf{0}$ budeme [vzhledem k (16.3.2)] řešit lineární rovnici (s neznámou $\mathbf{x} - \mathbf{x}^{(k)}$)

$$\mathbf{F}(\mathbf{x}^{(k)}) + \mathbf{F}'(\mathbf{x}^{(k)})(\mathbf{x} - \mathbf{x}^{(k)}) = \mathbf{0}. \tag{16.3.3}$$

Je-li matice $\mathbf{F}(\mathbf{x}^{(k)})$ regulární, potom rovnice (16.3.3) má jediné řešení, které označíme

$$\mathbf{x}^{(k+1)} - \mathbf{x}^{(k)} = \mathbf{h}^{(k)}, \quad \mathbf{h}^{(k)} = -[\mathbf{F}'(\mathbf{x}^{(k)})]^{-1}\mathbf{F}(\mathbf{x}^{(k)}). \tag{16.3.4}$$

Vektor $\mathbf{x}^{(k+1)}$ budeme považovat za novou (lepší) aproximaci řešení $\mathbf{x}^*$ rovnice $\mathbf{F}(\mathbf{x}) = \mathbf{0}$. Zdůrazněme, že $\mathbf{h}^{(k)}$ určujeme řešením rovnice $\mathbf{F}'(\mathbf{x}^{(k)})\mathbf{h} = -\mathbf{F}(\mathbf{h}^{(k)})$.

I když jsme z důvodů přehlednosti odvodili iterační vztah (16.3.4) pro soustavu dvou rovnic pro dvě neznámé, dovoluje nám meticově-vektorová symbolika převést všechny vztahy i na případ, kdy

$$\mathbf{F}(\mathbf{x}) = (F_1(x_1, x_2, ..., x_n), F_2(x_1, x_2, ..., x_n), ..., F_n(x_1, x_2, ..., x_n))^T.$$

Podmínky konvergence procesu (16.3.4) uvádět nebudeme. Jsou dosti komplikované (viz [1], [5]) a navíc, což je největší nedostatek, nedají se většinou prakticky prověřit. Při výpočtech pomocí formule (16.3.4) musíme dbát toho, aby $\mathbf{x}^{(0)}$ byla dobrá aproximace $\mathbf{x}^*$ a aby matice $\mathbf{F}'$ byla regulární v okolí $\mathbf{x}^*$. Potom konvergence procesu (16.3.4) bude rachlá (řádu 2).

16.3.1 Příklad.

Newtonovou metodou řešme soustavu

$$F_1(x,y) \equiv x^2 + y^2 - x = 0,$$
$$F_2(x,y) \equiv x^2 - y^2 - y = 0.$$

Křivky dané rovnicemi (kružnice a hyperbola) mají právě dva průsečíky. Jeden z nich je zhruba určen vektorem $\mathbf{x}^{(0)} = (0,8; 0,4)^T$. Tuto aproximaci řešení zpřesníme pomocí formule (16.3.4). Protože

$$\mathbf{F}'(\mathbf{x}^{(k)}) = \begin{pmatrix} \frac{\partial F_1(x_k,y_k)}{\partial x} & \frac{\partial F_1(x_k,y_k)}{\partial y} \\ \frac{\partial F_2(x_k,y_k)}{\partial x} & \frac{\partial F_2(x_k,y_k)}{\partial y} \end{pmatrix}, = \begin{pmatrix} ll2x-1 & 2y \\ 2x & -2y-1 \end{pmatrix}.$$

bude rovnice (16.3.3) mít tvar [označíme $\mathbf{x} = \mathbf{x}^{(0)} = \mathbf{h} = (h_1, h_2)^T$]:

$$(2x_0 - 1)h_1 + 2y_0 h_2 = -x_0^2 - y_0^2 + x_0,$$
$$2x_0 h_1 - (2y_0 + 1)h_2 = -x_0^2 + y_0^2 + y_0.$$

Řešením této soustavy (na 3 desetinná místa) bude

$$h_1^{(0)} = -0,027, \quad h_2^{(0)} = 0,020.$$

Odtud pak

$$\mathbf{x}^{(1)} = \mathbf{x}^{(0)} + \mathbf{h}^{(0)} = (0,8; 0,4)^T + (-0,027; 0,020)^T = (0,773; 0,420)^T.$$

Opakováním celého postupu dostáváme (na 7 desetinných míst)

$$\mathbf{x}^{(2)} = \mathbf{x}^{(1)} + \mathbf{h}^{(1)} = (0,771\,846\,1; 0,419\,644\,1)^T.$$

Pokračujeme-li ve výpočtech, dokud není splněna podmínka

$$\left\|\mathbf{x}^{(k-1)} - \mathbf{x}^{(k)}\right\|_{\mathbf{R}} < 10^{-7}.$$

dostaneme nakonec

$$\mathbf{x}^* \approx (0,771\,844\,5; 0,419\,643\,4)^T.$$

16.3.2 Poznámka.

Metodami čl. 16 lze stanovit též komplexní kořeny jedné rovnice $f(z) = 0$, řešíme-li soustavu dvou rovnic

$$u(x,y) = 0,$$
$$v(x,y) = 0,$$

kde $u(x,y) = \operatorname{Re} f(z)$, $v(x,y) = \operatorname{Im} f(z)$ a $x = x + \mathrm{i}\, y$.

16.4 Cvičení.

16.4.1

Řešte soustavu rovnic $y - x^2 = 0\,,\ x - xy + 1 = 0$ a) metodou prosté iterace; b) Newtonovou metodou. V případě a) vašetřete postačující podmínky konvergence metody. Počáteční aproximaci $\mathbf{x}^{(0)} = (x_0, y_0)^T$ stanovte graficky.

16.4.2

Řešte soustavu rovnic $x^2 + xy + y^2 = 3\,,\ \sin x - y^2 = 0$ a) metodou prosté iterace; b) Newtonovou metodou, víte-li, že $(x_0, y_0)^T = (1, 1)^T$ je dobrá počáteční aproximace.

[$(1{,}063\,7; 0{,}935\,0)^T$.]

16.4.3

Stanovte všechna (reálná) řešení soustavy rovnic $x^2 + y^2 = 1\,,\ xy = 0{,}4\,.$ Počáteční aproximace určíte graficky.

[$(0{,}447\,2; 0{,}894\,4)^T, (0{,}894\,4; 0{,}447\,2)^T, (-0{,}447\,2; -0{,}894\,4)^T,$
$(-0{,}894\,4; -0{,}447\,2)^T$.]

16.4.4

Soustavu $F_i(x_1, x_2, ..., x_n) = 0\,,\ i = 1, 2, ..., n\,,$ lze řešit pomocí iterační formule

$$x_i^{(k+1)} = x_i^{(k)} - \frac{F_i(x_1, x_2, ..., x_n)}{\frac{\partial F_i(x_1, x_2, ..., x_n)}{\partial x_i}}.$$

Užijte formule k řešení soustav z předcházejících cvičení. Aplikujte ji také v případě, že $F_i(x_1, x_2, ..., x_n) = a_{i1}x_1 + a_{i2}x_2 + ... + a_{in}x_n - b_n\,,$ tj. když $\mathbf{F}(\mathbf{x}) = \mathbf{A}\mathbf{x} - \mathbf{b}\,.$

k	$\mathbf{A}_k = \mathbf{L}_k\mathbf{L}_k^T = \mathbf{L}_{k-1}^T\mathbf{L}_{k-1}$	$\mathbf{L}_k$
0	$\begin{pmatrix} 2 & -1 & 0 \\ -1 & 2 & -1 \\ 0 & -1 & 1 \end{pmatrix}$	$\begin{pmatrix} 1,414\,2 & & \\ -0,707\,1 & 1,224\,7 & \\ & -0,813\,5 & 0,577\,3 \end{pmatrix}$
1	$\begin{pmatrix} 2,499\,9 & -0,866\,0 & \\ -0,866\,0 & 2,166\,6 & -0,471\,4 \\ & -0,471\,4 & 0,333\,3 \end{pmatrix}$	$\begin{pmatrix} 1,581\,1 & & \\ -0,547\,7 & 1,366\,2 & \\ & -0,345\,0 & 0,462\,9 \end{pmatrix}$
2	$\begin{pmatrix} 2,800\,0 & -0,748\,3 & \\ -0,748\,3 & 1,866\,6 & -0,159\,7 \\ & -0,159,7 & 0,214\,3 \end{pmatrix}$	$\begin{pmatrix} 1,673\,3 & & \\ -0,447\,2 & 1,341\,5 & \\ & -0,119\,0 & 0,447\,3 \end{pmatrix}$
3	$\begin{pmatrix} 2,999\,9 & -0,599\,9 & \\ -0,599\,9 & 1,813\,8 & -0,053\,2 \\ & -0,053\,2 & 0,200\,0 \end{pmatrix}$	$\begin{pmatrix} 1,732\,0 & & \\ -0,346\,3 & 1,301\,5 & \\ & -0,018\,3 & 0,446\,9 \end{pmatrix}$
4	$\begin{pmatrix} 3,119\,7 & -0,450\,7 & \\ -0,450\,7 & 1,694\,2 & \\ & -0,008\,2 & 0,199\,7 \end{pmatrix}$	$\begin{pmatrix} 1,766\,2 & & \\ -0,255\,2 & 1,276\,4 & \\ & -0,002\,8 & 0,446\,9 \end{pmatrix}$
5	$\begin{pmatrix} 3,184\,6 & -0,325\,7 & \\ -0,325\,7 & 1,629\,2 & -0,001\,6 \\ & -0,001\,6 & 0,199\,7 \end{pmatrix}$	$\begin{pmatrix} 1,784\,5 & & \\ -1,182\,5 & 1,263\,3 & \\ & -0,000\,4 & 0,446\,9 \end{pmatrix}$
6	$\begin{pmatrix} 3,217\,7 & -0,230\,5 & \\ -0,230\,5 & 1,595\,9 & -0,000\,1 \\ & -0,000\,1 & 0,199\,7 \end{pmatrix}$	atd.

k	a_k	b_k	s_{k+1}	$f(s_{k+1})$	$b_k - a_k$
0	1,500 00	2,000 00	1,750 0	< 0	0,5
1	1,750 00	2,000 00	1,875 00	< 0	0,25
2	1,875 00	2,000 00	1,937 50	> 0	0,125
3	1,875 00	1,937 50	0,906 25	< 0	0,062 5
4	1,906 25	1,937 50			0,031 25

k	$x_k = 2\sqrt{\sin x_{k-1}}$ $x_0 = 1,500\,00$	$x_k = 2\sqrt{\sin x_{k-1}}$ $x_0 = 2,000\,00$	$x_k = \arcsin(x_{k-1}/2)^2$ $x_0 = 2,000\,00$
1	$1,997\,49$	$1,907\,14$	$1,570\,79$
2	$1,908\,23$	$1,943\,16$	$1,644\,72 \notin I_0$
3	$1,942\,79$	$1,930\,25$	$0,110\,08 \notin I_0$
4	$1,930\,39$	$1,935\,03$	$0,003\,06 \notin I_0$
5	$1,934\,98$	$1,933\,28$	$\lvert x_5\rvert < 10^{-5}$
6	$1,933\,30$	$1,933\,92$	Rovnice $x = \arcsin(x/2)^2$
7	$1,933\,92$		nemá kořen v I_0!

k	s_k	a_k	b_{k+1}	$f(s_k)$	$f_k(a_k)$	$f_k(b_k)$
0	$-$	$1,500\,00$	$2,000\,0$	$-$	< 0	> 0
1	$1,913\,73$	$1,913\,75$	$2,000\,00$	< 0	< 0	> 0
2	$1,933\,05$	$1,933\,05$	$2,000\,00$	< 0	< 0	> 0
3	$1,933\,73$	$1,933\,73$	$2,000\,00$	< 0	< 0	> 0
4	$1,933\,75$					

k	x_k	$f(x_k)$	$f'(x_k)$	$h_k = -\frac{f(x_k)}{f'(x_k)}$	$\varepsilon_k = x_k - \alpha$
0	$1,500\,00$	$-0,434\,995$	$0,679\,263$	$6,403\,930 \cdot 10^{-1}$	$-4,337\,50 \cdot 10^{-1}$
1	$2,140\,39$	$0,303\,197$	$1,609\,48$	$-1,883\,81 \cdot 10^{-1}$	$2,066\,36 \cdot 10^{-1}$
2	$1,952\,01$	$2,437\,19 \cdot 10^{-2}$	$1,348\,05$	$-1,807\,93 \cdot 10^{-2}$	$1,825\,63 \cdot 10^{-2}$
3	$1,933\,93$	$2,329\,91 \cdot 10^{-4}$	$1,322\,17$	$-1,762\,18 \cdot 10^{-4}$	$1,762\,36 \cdot 10^{-4}$
4	$1,933\,75$	$-4,975\,70 \cdot 10^{-6}$	$1,321\,91$	$3,764\,02 \cdot 10^{-6}$	$-3,764 \cdot 10^{-6}$

k	$a_0^{(k)}$	$a_1^{(k)}$	$a_2^{(k)}$	$a_3^{(k)}$	$P^{(k)}(z)$
0	1	-9	5	-6	$P^{(0)}(z)$
1	1	-71	-83	-36	$P^{(1)}(z)$
2	1	$-5\,207$	$1\,777$	$-1\,296$	$P^{(2)}(z)$
3	1	$-2,710\,9 \cdot 10^7$	$-1,033\,9 \cdot 10^7$	$-1,679\,6 \cdot 10^6$	$P^{(3)}(z)$

k	$a_0^{(k)}$	$a_1^{(k)}$	$a_2^{(k)}$	$a_3^{(k)}$	$a_4^{(k)}$	$P^{(k)}(z)$
0	1	-4	3	2	-6	$P^{(0)}(z)$
1	1	-10	13	-40	36	$P^{(1)}(z)$
2	1	-74	-599	-644	$1\,296$	$P^{(2)}(z)$
3	1	$-6,594$	$216\,801$	$-1,889\,824$	$1\,679\,616$	$P^{(3)}(z)$