VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INTELIGENTNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INTELLIGENT SYSTEMS

SROVNÁVACÍ VÝPOČTY ALGEBRAICKÝCH ROVNIC

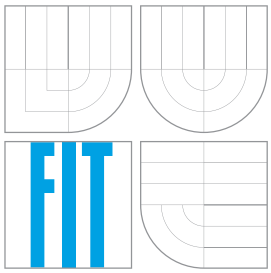
BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR

GABRIELA NEČASOVÁ

BRNO 2012

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INTELIGENTNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INTELLIGENT SYSTEMS

SROVNÁVACÍ VÝPOČTY ALGEBRAICKÝCH ROVNIC

ALGEBRAIC EQUATIONS COMPARISONS

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — GABRIELA NEČASOVÁ
AUTHOR

VEDOUCÍ PRÁCE — Ing. VÁCLAV ŠÁTEK, Ph.D.
SUPERVISOR

BRNO 2012

Abstrakt

Práce se zabývá tématem srovnávacích výpočtů algebraických rovnic. Práce nejprve popisuje srovnání celkového počtu operací u přímých a iteračních metod, zároveň na konkrétních příkladech demonstruje metody a vysvětluje použití přímých a iteračních metod. V další části práce se zaměřuji na možné metody převodu soustav lineárních algebraických rovnic (SLAR) na soustavy diferenciálních rovnic (SDR). V závěru je popsán způsob práce s programy TKSL/C, Matlab a Maple. V rámci bakalářské práce bylo navrženo grafické uživatelské rozhraní pro program TKSL/C sloužící k pohodlné komunikaci s programem. Grafické uživatelské rozhraní bylo otestováno na konkrétních úlohách demonstrujících převod SLAR na SDR.

Abstract

The thesis deals with the topic of comparative calculation of algebraic equations. First it describes the comparison of the overall number of operations at direct and iteration methods, as well as gives concrete examples of the methods and explains solutions of direct and iteration methods. Another part focuses on possible methods of converting systems of linear algebraic equations to the system of differential equations. The end of the thesis describes method of working with TKSL/C, Matlab and Maple. In this thesis, there was designed graphical user interface serving for comfortable communication with TKSL/C programme. Graphical user interface was tested on concrete tasks demonstrating the conversion of system of linear algebraic equations to the system of differential equations.

Klíčová slova

Řešení soustav lineárních algebraických rovnic, přímé metody, iterační metody, výpočetní náročnost, tuhé systémy, diferenciální rovnice TKSL/C, Matlab, Maple.

Keywords

Solving linear algebraic equations, direct methods, iteration methods, computational complexity, stiff systems, differential equations, TKSL/C, Matlab, Maple.

Citace

Gabriela Nečasová: Srovnávací výpočty algebraických rovnic, bakalářská práce, Brno, FIT VUT v Brně, 2012

Srovnávací výpočty algebraických rovnic

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracovala samostatně, pod vedením pana Ing. Václava Šátka, Ph.D.
Uvedla jsem všechny literární prameny a publikace, ze kterých jsem čerpala.

.....
Gabriela Nečasová
11. května 2012

Poděkování

Velice děkuji panu Ing. Václavovi Šátkovi, Ph.D. za poskytnuté rady a podnětné připomínky při řešení této bakalářské práce.

Obsah

Seznam obrázků

Seznam tabulek

Kapitola 1

Úvod

V dnešní době rostou požadavky řešit stále rozsáhlejší úlohy. Náročné výpočty jsou zapotřebí v různých vědeckých oborech a pro řešení multidisciplinárních problémů. Využívají se k řešení složitých výpočetních úloh, jako jsou například fyzikální modelování, modelování chemických či biologických procesů, vyhodnocování velkého množství naměřených dat.

Tato práce se zabývá problematikou výpočetní náročnosti algebraických rovnic. Vysvětluje použití přímých a iteračních metod, zároveň seznamuje s jejich výpočetní náročností. Při výpočtu soustav lineárních algebraických rovnic, ať už přímými nebo iteračními metodami, se můžeme setkat s nestabilitou metod. Možné řešení, jak se s tímto problémem vypořádat, je převést soustavu lineárních algebraických rovnic (SLAR) na soustavu diferenciálních rovnic (SDR), čímž je vždy zajištěna stabilita řešení soustavy rovnic. K převodu byl využit program TKSL/C UI, který vznikl v rámci bakalářské práce jako grafické uživatelské rozhraní k programu TKSL/C. Samotný výpočet probíhá pomocí programu TKSL/C, který je volán z aplikace TKSLC/UI.

Cílem práce je podat ucelený přehled metod řešení algebraických rovnic, seznámit se principem výpočtu algebraických rovnic pomocí rovnic diferenciálních a analyzovat vznik tuhých (stiff) systémů.

Práce je rozdělena do šesti kapitol. Druhá kapitola je věnována přímým metodám, jako jsou Cramerovo pravidlo, Gaussova eliminační metoda, Gauss-Jordanova eliminační metoda a inverzní matice. V druhé kapitole se zaměřujeme na iterační metody. Je zde ukázána Jacobiho a Gauss-Seidelova metoda. Třetí kapitola je věnována diferenciálním rovnicím. Stručně nastiňuje možné metody řešení diferenciálních rovnic, především se však zaměřuje na převod SLAR na SDR pomocí transformačního algoritmu či elementárního převodu. Kapitola nás také seznamuje s programem TKSL/C a se způsobem zápisu SDR v něm. Závěr kapitoly je věnován tuhým systémům a jejich analýze. Pátá kapitola srovnává programy TKSL/C, Matlab a Maple. Také byla v rámci této kapitoly provedena srovnání výpočetní náročnosti řešení SLAR a SDR pomocí programů TKSL/C a Matlab. Šestá a poslední kapitola popisuje způsob práce s programem TKSL/C UI. Práce obsahuje přílohy.

Kapitola 2

Přímé metody

Pomocí přímých metod se dostaneme k řešení soustavy lineárních algebraických rovnic po konečném počtu kroků. Více informací o tomto tématu jsem v této kapitole čerpala z [4, 10, 11, 3, 6]. Pro matice, kterými se budeme v tomto textu zabývat, předpokládáme, že jsou regulární a jejich řádky jsou lineárně nezávislé. Nalezené řešení by bylo přesné, pokud bychom se v průběhu výpočtu nedopouštěli zaokrouhlovacích chyb. Úkolem je řešit soustavu n linerních rovnic o n neznámých.

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n &= b_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n &= b_2 \\ &\vdots \\ a_{n1}x_1 + a_{n2}x_2 + \cdots + a_{nn}x_n &= b_n \end{aligned} \tag{2.1}$$

Soustavu lze zapsat také ve vektorovém tvaru.

$$\mathbf{A} \cdot \vec{x} = \vec{b} \tag{2.2}$$

Pro ověření řešitelnosti soustavy lineárních algebraických rovnic lze využít Frobeninovu větu.

Věta 2.1. *Nehomogenní soustava lineárních algebraických rovnic má řešení pouze tehdy, když hodnost matice soustavy $h(A)$ je rovna hodnosti rozšířené matice soustavy $h(A|\vec{b})$. Pokud je $h(A)$ rovno počtu neznámých, potom má soustava jedno řešení. Pokud je $h(A)$ menší než počet neznámých, řešení je nekonečně mnoho. Pokud je $h(A)$ větší než počet neznámých, soutava nemá řešení.*

Důkaz věty 2.1 je uveden v [5].

2.1 Cramerovo pravidlo

Pokud je matice soustavy regulární, tj. její determinant je nenulový, pak řešení soustavy lze vypočítat jako

$$x_1 = \frac{D_1}{D},\ x_2 = \frac{D_2}{D}, \ldots, x_n = \frac{D_n}{D}, \tag{2.3}$$

kde D je determinant soustavy $\mathbf{A}$ a D_k, $k = 1, \ldots, n$ jsou determinanty matic, které vzniknou z matice $\mathbf{A}$ nahrazením k-tého sloupce matice vektorem pravých stran $\mathbf{b}$. Při použití Cramerova pravidla je tedy zapotřebí sestavit $n+1$ determinantů řádu n. Nyní se pokusíme spočítat počet operací, který je k výpočtu potřeba. Počty operací, které budeme zjišťovat, rozdělíme do dvou skupin:

- počet součinů a podílů,
- počet součtů a rozdílů.

Definice 2.1. *Pokud je řád deteminantu menší nebo roven třem, lze aplikovat Sarrusovo pravidlo. Nechť*

$$A = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix}$$

potom

$$\det(A) = a_{11}a_{22}a_{33} + a_{12}a_{23}a_{31} + a_{13}a_{21}a_{32} - a_{13}a_{22}a_{31} - a_{12}a_{21}a_{33} - a_{11}a_{23}a_{32}\,.$$

Příklad 2.1. *Vyřešte níže uvedenou soustavu dvou lineárních rovnic o dvou neznámých pomocí Cramerova pravidla.*

$$\begin{aligned} 3x + 2y &= 2 \\ 5x + 4y &= 6 \end{aligned}$$

Determinant soustavy je

$$D = \begin{vmatrix} 3 & 2 \\ 5 & 4 \end{vmatrix} = 3 \cdot 4 - 2 \cdot 5 = 2\,.$$

Determinanty vzniklé záměnou k-tého sloupce matice vektorem pravých stran $\mathbf{b}$ jsou uvedeny níže.

$$D_x = \begin{vmatrix} 3 & 2 \\ 5 & 6 \end{vmatrix} = 3 \cdot 6 - 2 \cdot 5 = 8$$

$$D_y = \begin{vmatrix} 2 & 2 \\ 6 & 4 \end{vmatrix} = 2 \cdot 4 - 2 \cdot 6 = -4$$

Řešení je

$$x = \frac{D_x}{D} = \frac{8}{2} = 4, \qquad y = \frac{D_y}{D} = \frac{-4}{2} = -2\,.$$

□

Vidíme, že pro každý determinant bylo potřeba

- 2 součiny,
- 1 rozdíl.

Těchto determinantů je $n+1$. Je tedy 6 operací násobení a 3 operace odečítání. K násobení ještě ale musíme započítat 2 operace dělení pro získání vysledků. Těchto podílů je vždy právě n.

Celkem tedy pro soustavu 2×2 máme

- 8 součinů/podílů,
- 3 rozdíly.

Nyní vezmeme soustavu tří lineárních rovnic o třech neznámých.

Příklad 2.2. *Vyřešte následující soustavu rovnic s využitím Cramerova pravidla.*

$$\begin{aligned} 3x + y + z &= 3 \\ 2x + 2y + 5z &= -1 \\ x - 3y - 4z &= 2 \end{aligned}$$

Determinant soustavy je

$$D = \begin{vmatrix} 3 & 1 & 1 \\ 2 & 2 & 5 \\ 1 & -3 & -4 \end{vmatrix} = 3\cdot 2\cdot(-4) + 1\cdot 5\cdot 1 + 1\cdot 2\cdot(-3) - 1\cdot 2\cdot(-4) - 3\cdot 5\cdot(-3) - 1\cdot 2\cdot 1 = 26\,.$$

Determinanty vzniklé záměnou k-tého sloupce matice vektorem pravých stran **b** jsou uvedeny níže.

$$D_x = \begin{vmatrix} 3 & 1 & 1 \\ -1 & 2 & 5 \\ 2 & -3 & -4 \end{vmatrix} = 3\cdot 2\cdot(-4) + 1\cdot 5\cdot 2 + 1\cdot(-1)\cdot(-3) - 1\cdot(-1)\cdot(-4) - 3\cdot 5\cdot(-3) - 1\cdot 2\cdot 2 = 26$$

$$D_y = \begin{vmatrix} 3 & 3 & 1 \\ 2 & -1 & 5 \\ 1 & 2 & -4 \end{vmatrix} = 3\cdot(-1)\cdot(-4) + 3\cdot 5\cdot 1 + 1\cdot 2\cdot 2 - 3\cdot 2\cdot(-4) - 3\cdot 5\cdot 2 - 1\cdot(-1)\cdot 1 = 26$$

$$D_z = \begin{vmatrix} 3 & 1 & 3 \\ 2 & 2 & -1 \\ 1 & -3 & 2 \end{vmatrix} = 3\cdot 2\cdot 2 + 1\cdot(-1)\cdot 1 + 3\cdot 2\cdot(-3) - 1\cdot 2\cdot 2 - 3\cdot(-1)\cdot(-3) - 3\cdot 2\cdot 1 = -26$$

Řešení soustavy rovnic je

$$x = \frac{D_x}{D} = \frac{26}{26} = 1, \qquad y = \frac{D_y}{D} = \frac{26}{26} = 1, \qquad z = \frac{D_z}{D} = \frac{-26}{26} = -1.$$

□

Vidíme, že pro každý determinant bylo potřeba

- 12 součinů,
- 5 součtů/rozdílů.

Těchto determinantů je $n + 1$. Je tedy 48 operací násobení a 20 operací sčítání/odečítání. Opět musíme započítat operace dělení pro získání vysledků. Těchto podílů je vždy právě n.

Celkový počet operací je

- 52 součinů/podílů,
- 20 součtů/rozdílů.

Pokud je ovšem řád determinantu větší než 3, je nutné pro použití Cramerova pravidla aplikovat nejprve Laplaceův rozvoj determinantu.

Věta 2.2. *Pro čtvercovou matici* $\mathbf{A}$ *řádu* n *platí:* $\det \mathbf{A} = \sum_{j=1}^{n} (-1)^{(i+j)} a_{ij} \cdot \det \mathbf{A}_{ij}$ *je rozvoj determinantu podle* i*-tého řádku,* $\det \mathbf{A} = \sum_{i=1}^{n} (-1)^{(i+j)} a_{ij} \cdot \det \mathbf{A}_{ij}$ *je rozvoj determinantu podle* j*-tého sloupce, kde matice* $\mathbf{A}_{ij}$ *vznikne z matice* $\mathbf{A}$ *vynecháním* i*-tého řádku a* j*-tého sloupce.*

Důkaz věty 2.2 je uveden v [5].

Příklad 2.3. *Proveďte Laplaceův rozvoj determinantu řádu 4 podle 1. řádku.*

$$\begin{aligned} 2w + 3x + 4y + 5z &= 7 \\ 4w + 7x + 9y + 8z &= 9 \\ 2w + x + 5y + 3z &= 8 \\ 7w + 9x + 4y + 2z &= 5 \end{aligned}$$

$$D = \begin{vmatrix} 2 & 3 & 4 & 5 \\ 4 & 7 & 9 & 8 \\ 2 & 1 & 5 & 3 \\ 7 & 9 & 4 & 2 \end{vmatrix} = 2 \cdot (-1)^{(1+1)} \cdot \begin{vmatrix} 7 & 9 & 8 \\ 1 & 5 & 3 \\ 9 & 4 & 2 \end{vmatrix} + 3 \cdot (-1)^{(1+2)} \cdot \begin{vmatrix} 4 & 9 & 8 \\ 2 & 5 & 3 \\ 7 & 4 & 2 \end{vmatrix} +$$

$$+4 \cdot (-1)^{(1+3)} \cdot \begin{vmatrix} 4 & 7 & 8 \\ 2 & 1 & 3 \\ 7 & 9 & 2 \end{vmatrix} + 5 \cdot (-1)^{(1+4)} \cdot \begin{vmatrix} 4 & 7 & 9 \\ 2 & 1 & 5 \\ 7 & 9 & 4 \end{vmatrix}$$

□

Při použití Cramerova pravidla musíme sestavit $n + 1$ determinantů řádu n. Pokud máme soustavu rovnic o čtyřech neznámých, tak potom budeme vytvářet celkem 4 determinanty řádu 3. Vytvářené determinanty budou následující:

- determinant soustavy,
- subdeterminanty.

Subdeterminanty jsou determinanty matic, které vzniknou z původní matice soustavy nahrazením k-tého sloupce matice vektorem pravých stran $\mathbf{b}$. Ten můžeme umístit na 3 různé pozice, vzniknou nám tedy celkem 3 determinanty.

Je-li daný determinant vyššího řádu než 3, potom musíme pro každý determinant sestavit n determinantů řádu $n - 1$. Konkrétně pro determinant čtvrtého řádu to znamená, že musíme sestavit 5 determinantů čtvrtého řádu a pro každý z těchto 5 determinantů je nutné sestavit 4 determinanty třetího řádu.

2.1.1 Počet operací násobení a dělení

Nyní spočteme počet operací násobení/dělení potřebný pro 1 Laplaceův rozvoj determinantu čtvrtého řádu.

- Počet násobení před determinanty řádu $n-1$ v Laplaceově rozvoji je 8 (těchto součinů je vždy právě $2n$).
- Počet násobení pro výpočet determinantu třetího řádu je 12 (viz příklad 2.2).
- Počet násobení pro výpočet 4 determinantů třetího řádu je $12 \cdot 4 = 48$.

Počet operací dělení je vždy právě n, kde n představuje řád soustavy. Celkový počet operací násobení pro 1 Laplaceův rozvoj determinantu čtvrtého řádu je tedy $48+8 = 56$. Toto číslo prohlásíme za konstantu, protože ji budeme dále využívat. Budeme pracovat s determinanty vyšších řádů než 3, což znamená, že daný determinant budeme pomocí Laplaceova rozvoje rozkládat tak dlouho, dokud se nedopracujeme k determinantům třetího řádu, které již umíme vyčíslit pomocí Sarrusova pravidla 2.1.

Počet operací násobení/dělení pro determinant čtvrtého řádu je

$$56 \cdot (n+1) + n \,, \text{kde } n{=}4.$$

Vytváříme 5 determinantů čtvrtého řádu a započítáme podíly potřebné pro zisk výsledku.

Počet operací násobení/dělení pro determinant pátého řádu je

$$[n \cdot 56 + 2n] \cdot (n+1) + n \,, \text{kde } n{=}5.$$

Vytváříme 6 determinantů (člen $(n+1)$) pátého řádu (všechny členy obsažené v hranaté závorce). Pro 1 determinant pátého řádu je potřeba vytvořit 5 determinantů čtvrtého řádu (člen $n \cdot 56$). Pro každý determinant čtvrtého řádu je zapotřebí sestavit 4 determinanty třetího řádu, čemuž odpovídá konstanta 56. Na závěr započítáme podíly potřebné pro zisk výsledku (člen n). Všimněme si, že nám zde přibyl člen $2n$, který vyjadřuje počet násobení v Laplaceově rozvoji pro pátý řád determinantu. Tento člen jsme museli započítat, protože již není součástí konstanty 56.

Počet operací násobení/dělení pro determinant šestého řádu je

$$n[2n + 2(n-1) + (n-1)56] \cdot (n+1) + n \,, \text{kde } n{=}6.$$

Vytváříme 7 determinantů šestého řádu a započítáme podíly potřebné pro zisk výsledku. Pro přehlednost rozebereme, co znamenají jednotlivé členy ve vzorci:

- n vyjadřuje, že vytváříme 6 determinantů,
- $2n$ udává počet součinů v Laplaceově rozvoji pro determinant řádu 6,
- $2(n-1)$ představuje počet součinů v Laplaceově rozvoji pro determinant řádu 5,
- $(n-1)56$ udává, že je zde zahrnuto 5 determinantů řádu 4 a 4 determinanty řádu 3
- $(n+1)$ vyjadřuje, že všechny tyto operace bude potřeba provést $(n+1)$ krát, protože $(n+1)$ je počet determinantů, které musíme vytvořit při použití Cramerova pravidla,
- n je počet podílů k dosažení výsledků.

Od vyšších řádů determinantů k nižším se dostáváme přes postupné Laplaceovy rozvoje, ale započítáváme zde pouze operace násobení před determinanty řádu $n-1$. Hlavní myšlenka je ta, že skutečně počítáme až ty determinanty, které jsou třetího řádu. Jejich vyčíslením dostaneme hodnoty determinantu čtvrtého řádu a tyto hodnoty determinantů čtvrtého řádu jsou využity pro výpočet determinantů pátého řádu, atd.

Takto analogicky postupujeme dále – tedy od nižších řádů determinantů k vyšším. Nyní uvedeme další vztahy pro výpočet počtu operací násobení/dělení.

Počet operací násobení/dělení pro determinant sedmého řádu je

$$n[2n + 2(n-1) + 2(n-2) + (n-2)56] \cdot (n+1) + n\,, \text{kde } n{=}7.$$

Počet operací násobení/dělení pro determinant osmého řádu je

$$n[2n + 2(n-1) + 2(n-2) + 2(n-3) + (n-3)56] \cdot (n+1) + n\,, \text{kde } n{=}8.$$

Počet operací násobení/dělení pro determinant devátého řádu je

$$n[2n + 2(n-1) + 2(n-2) + 2(n-3) + 2(n-4) + (n-4)56] \cdot (n+1) + n\,, \text{kde } n{=}9.$$

Počet operací násobení/dělení pro determinant desátého řádu je

$$n[2n + 2(n-1) + 2(n-2) + 2(n-3) + 2(n-4) + 2(n-5) + (n-5)56]\cdot$$

$$\cdot(n+1) + n\,, \text{kde } n{=}10.$$

Lze si všimnout, že s narůstajícím řádem determinantu ve vztazích přibývají členy $2(n-x)$, kde $x = 0, \ldots, n-5$, což jsou členy vyjadřující již zmíněné násobení v daném Laplaceově rozvoji před determinanty řádu $n-1$.

Obecný vztah pro vytvoření vzorce pro řád $n > 5$ ukazuje vztah (2.4).

$$n[2n + \cdots + 2(n-x) + (n-x)56] \cdot (n+1) + n, \text{ pro } x = 0, \ldots, n-5 \tag{2.4}$$

2.1.2 Počet operací sčítání a odčítání

Princip výpočtu je podobný jako u operací násobení a dělení. Opět zavádíme konstantu, tentokrát pro počet sčítání a odčítání pro 1 Laplaceův rozvoj determinantu **čtvrtého řádu**.

- Počet sčítání/odčítání před determinanty řádu $n-1$ v Laplaceově rozvoji je 3 (těchto součinů je vždy právě $n-1$).
- Počet sčítání/odčítání pro výpočet determinantu třetího řádu je 5, počet sčítání/odčítání pro výpočet 4 determinantů třetího řádu je $4 \cdot 5 = 20$.

Celkem tedy

$$20 + 3 = 23$$

operací sčítání/odčítání. Toto číslo prohlásíme opět za konstantu.

Počet operací sčítání/odčítání pro determinant čtvrtého řádu je

$$23 \cdot (n+1)\,, \text{kde } n{=}4.$$

Vytváříme 5 determinantů čtvrtého řádu.

Počet operací sčítání/odčítání pro determinant pátého řádu je

$$\{[(n-1)+n\cdot 23]\cdot(n+1)\}\,,\text{kde } n{=}5.$$

Vytváříme 6 determinantů pátého řádu. Všimněme si, že nám zde přibyl člen $(n-1)$, který vyjadřuje počet násobení v Laplaceově rozvoji před determinantem čtvrtého řádu. Tento člen bylo nutné započítat, protože již není součástí konstanty 23.

Počet operací sčítání/odčítání pro determinant šestého řádu je

$$\{(n-1)+n\cdot[(n-2)+(n-1)\cdot 23]\}\cdot(n+1)\,,\text{kde } n{=}6.$$

Vytváříme 7 determinantů šestého řádu.

Nyní podrobněji vysvětlíme jednotlivé členy ve vztahu:

- $(n-1)$ je počet součtů v Laplaceově rozvoji,
- n vyjadřuje, že vytváříme 6 determinantů řádu 5,
- $(n-2)$ udává počet součtů v Laplaceově rozvoji pro determinant řádu 5,
- $(n-1)23$ značí, že je zde zahrnuto 5 determinantů řádu 4 a 4 determinanty řádu 3,
- $(n+1)$ znamená, že všechny výše uvedené výpočty bude nutné provést $(n+1)$ krát, protože $(n+1)$ je počet determinantů, které musíme vytvořit při použití Cramerova pravidla.

Počet operací sčítání/odčítání pro determinant sedmého řádu je

$$\{(n-1)+n\cdot[(n-2)+(n-1)\cdot[(n-3)+(n-2)\cdot 23]]\}\cdot(n+1)\,,\text{kde } n{=}7.$$

Počet operací sčítání/odčítání pro determinant osmého řádu je

$$\{(n-1)+n\cdot[(n-2)+(n-1)\cdot[(n-3)+(n-2)\cdot[(n-4)+(n-3)\cdot 23]]]\}\cdot(n+1)\,,\text{kde } n{=}8.$$

Počet operací sčítání/odčítání pro determinant devátého řádu je

$$\{(n-1)+n\cdot[(n-2)+(n-1)\cdot[(n-3)+(n-2)\cdot[(n-4)+(n-3)\cdot[(n-5)+(n-4)\cdot 23]]]]\}\cdot(n+1)\,,$$

$$\text{kde } n{=}9.$$

Počet operací sčítání/odčítání pro determinant desátého řádu je

$$\{(n-1)+n\cdot[(n-2)+(n-1)\cdot[(n-3)+(n-2)\cdot[(n-4)+(n-3)\cdot[(n-5)+(n-4)\cdot$$

$$\cdot[(n-6)+(n-5)\cdot 23]]]]\}\cdot(n+1)\,,\text{kde } n{=}10.$$

Lze si všimnout, že s narůstajícím řádem determinantu ve vztazích přibývají členy $(n-x)$. S každým krokem přibývá více a více závorek, protože vždy vycházíme z předchozího kroku, který využijeme k vyčíslení počtu operací pro aktuální řád. Obecný tvar výpočtu můžeme popsat vztahem (2.5).

$$\{(n-1)+n\cdot[slozenaZavorka]\}\cdot(n+1), \tag{2.5}$$

kde *slozenaZavorka* je obsah složené závorky z předešlého kroku snížený o 1. Tímto mechanismem dochází k „množení závorek“. Počet operací při použití Cramerova pravidla znázorňuje tabulka 2.1.

Tabulka 2.1: Počty operací při řešení soustavy rovnic Cramerovým pravidlem

matice (řádky x sloupce)	součiny/podíly	součty/rozdíly
2x2	8	3
3x3	52	20
4x4	284	115
5x5	1745	714
6x6	12690	5033
7x7	17703	40312
8x8	23912	362871
9x9	31509	3628790
10x10	40710	39916789
...	...	...

2.2 Gaussova eliminační metoda

Další běžně používanou metodou pro řešení soustav lineárních algebraických rovnic je Gaussova eliminační metoda, která je podrobně rozvedena v [2]. Cílem této metody je převést rozšířenou matici soustavy na horní trojúhelníkový tvar pomocí elementárních řádkových úprav, což jsou:

1. libovolná záměna pořadí řádků,
2. vynásobení libovolného řádku nenulovým číslem,
3. přičtení lineární kombinace několika řádků k jednomu řádku.

Algoritmus Gaussovy eliminační metody 2.1 je uveden níže.

Algoritmus 2.1. *1. Přímý chod – zde je cílem danou rozšířenou matici soustavy převést na horní trojúhelníkový tvar. Postupujeme následovně:*

- *upravíme matici soustavy tak, aby prvek $a_{11} \neq 0$,*
- *odečteme vhodný násobek prvního řádku od ostatních tak, abychom získali nulové členy pod prvkem a_{kk}, kde $k = 1, \ldots, n$,*
- *takto postupujeme dále tak dlouho, dokud nezískáme všechny nulové pozice pod hlavní diagonálou matice.*

Cílem je získat horní trojúhelníkový tvar.

$$A = \left(\begin{array}{cccc|c} a_{11} & a_{12} & \ldots & a_{1n} & b_1 \\ 0 & a_{22} & \ldots & a_{2n} & b_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \ldots & a_{nn} & b_n \end{array}\right)$$

2. Zpětný chod – zde postupně vypočítáváme hodnoty řešení. Postupujeme v obráceném pořadí, získáváme tedy nejprve hodnotu x_n a nakonec hodnotu x_1. Po skončení Gaussovy eliminační metody máme soustavu ve výše uvedeném tvaru.

Výpočet neznámé můžeme vyjádřit následujícím vztahem, který se nazývá zpětnou substitucí, viz (2.6).

$$x_i = \frac{1}{d_{ii}} \cdot (e_i - d_{i,i+1}x_{i+1} - d_{i,i+2}x_{i+2} - \ldots - d_{i,n}x_n) \text{ pro } i = 1, \ldots, n. \tag{2.6}$$

2.2.1 Počet operací násobení a dělení

Nejprve si uvědomíme, že rozšířená matice soustavy má vždy rozměry $n \times (n+1)$, protože započítáváme i vektor pravých stran. Pokusíme se spočítat, kolik operací násobení a dělení je potřeba pro první iteraci Gaussovy eliminační metody. Pro ilustraci vezměme příklad A.1, ve kterém vyřešíme jednoduchou soustavu 3 lineárních rovnic o 3 neznámých.

Nebudeme totiž násobit všech $(n+1)$ řádků rozšířené matice soustavy, protože pokud vezmeme v úvahu například první řádek matice, ten vynásobíme číslem $-(a_{21}/a_{11})$ a poté ho celý přičítáme k druhému řádku matice. Druhou možností je řádek násobit číslem (a_{21}/a_{11}) a poté od něj odečíst druhý řádek. Je ale jasné, že koeficient (a_{21}/a_{11}) byl zvolen tak, abychom na místě (a_{21}) získali nulu. Proto je tedy na úpravu druhého řádku potřeba

- $n+1$ operací násobení/dělení.

Pokud přihlédneme k našemu příkladu, znamenalo by to, že bychom členy 1, 2 a 3 z prvního řádku matice násobili koeficientem $-(a_{21}/a_{11}) = -(2/1) = -2$. Počet operací odpovídá výše uvedenému vztahu. Takto musíme upravit všech $(n-1)$ řádků pod prvním řádkem rozšířené matice soustavy. Celkem tedy celá první iterace požaduje

- $(n+1)\cdot(n-1) = n^2 - 1$ operací násobení/dělení.

Podívejme se nyní, jak to bude vypadat v druhé iteraci. Nyní se již nemusíme zabývat prvky v prvním řádku a prvním sloupci. Počet operací je

- $n\cdot(n-2) = (n-1)^2 - 1$ operací násobení/dělení.

Celkový počet operací násobení/dělení pro soustavu řádu n vyjadřuje vztah (2.7).

$$\sum_{i=1}^{n}(i^2-1) \tag{2.7}$$

Počet operací násobení/dělení pro zpětnou substituci, jejíž výpočet získáme dle vztahu (2.6) je

$$\sum_{i=1}^{n}(n-i+2) = \sum_{i=1}^{n}(i+1)\,. \tag{2.8}$$

Pro objasnění uvedeme příklad 2.4.

Příklad 2.4. *Získání řešení soustavy rovnic z příkladu A.1 zpětnou substitucí dle vztahu (2.6).*

$$A = \left(\begin{array}{ccc|c} 1 & 2 & 3 & 4 \\ 0 & -1 & -1 & -1 \\ 0 & 0 & -9 & -15 \end{array}\right)$$

$$\begin{aligned} z &= -\frac{1}{9}\cdot -15 \\ y &= -\frac{1}{1}\cdot(-1-(-1)\cdot x_3) \\ x &= \frac{1}{1}\cdot(4-2\cdot x_2-3\cdot x_3) \end{aligned}$$

Řešení soustavy je

$$x = \frac{1}{3},\ y = -\frac{2}{3},\ z = \frac{5}{3}\,.$$

□

Vidíme, že pro $i = 1$ (tedy pro x_1) potřebujeme 4 operace (3 součiny a 1 podíl), pro $i = 2$ (tedy pro x_2) 3 operace (2 součiny a 1 podíl) a konečně pro $i = 3$ (tedy pro x_3) 2 operace (1 součin a 1 podíl). Pokud vyjádříme počet operací násobení/dělení v přímém i zpětném chodu Gaussovy eliminace dostaneme výsledný vztah (2.9).

$$\sum_{i=1}^{n}(i^2 - 1) + \sum_{i=1}^{n}(i + 1) \tag{2.9}$$

2.2.2 Počet operací sčítání a odčítání

Princip je podobný jako při zjišťování počtu operací násobení/dělení. Pro názornost můžeme použít příklad A.1. Zde bylo řečeno, že první řádek násobíme koeficientem 2 a nyní od tohoto řádku budeme chtít odečíst druhý řádek, čímž získáme nulu na pozici a_{21}. Tedy odečítáme pouze následující členy.

$$2 \cdot 2 - 3 = 1$$
$$2 \cdot 3 - 5 = 1$$
$$2 \cdot 4 - 7 = 1$$

Na úpravu 2. řádku matice (1. iterace) je potřeba

- n sčítání/odčítání.

Tímto způsobem musím upravit všech zbývajících $(n - 1)$ řádků pod prvním řádkem rozšířené matice soustavy. Celá druhá iterace Gaussovy eliminace požaduje

- $n \cdot (n - 1) = n^2 - n$ sčítání/odčítání.

Ve třetí iteraci Gaussovy eliminace se již nemusíme zabývat prvky v prvním řádku a prvním sloupci, třetí iterace tedy požaduje

- $(n - 2) \cdot (n - 1) = (n - 1)^2 - (n - 1)$ sčítání/odčítání.

Vztah (2.10) znázorňuje počet operací sšítání/odčítání pro přímý chod Gaussovy eliminace.

$$\sum_{i=1}^{n}(i^2 - i) \tag{2.10}$$

Počet operací pro zpětný chod můžeme vyčíst z příkladu 2.4. Pro $i = 1$ (tedy pro x_1) potřebujeme 2 operace sčítání/odčitání, pro $i = 2$ (tedy pro x_2) 1 operaci a pro $i = 3$ (tedy pro x_3) 0 operací sčítání/odčítání. Počet operací pro zpětný chod je uveden ve vztahu (2.11).

$$\sum_{i=1}^{n}(n - i) = \sum_{0}^{n-1} i \tag{2.11}$$

Celkový počet operací sčítání/odčítání pro přímý i zpětný chod Gaussovy eliminace ukazuje vztah (2.12).

$$\sum_{i=1}^{n}(i^2 - i) + \sum_{0}^{n-1} i \tag{2.12}$$

Počty operací jsou shrnuty do tabulky 2.2.

Tabulka 2.2: Počty operací při použití Gaussovy eliminační metody

matice (řádky x sloupce)	součiny/podíly	součty/rozdíly
2x2	6	3
3x3	17	11
4x4	36	26
5x5	65	50
6x6	106	85
7x7	161	133
8x8	232	196
9x9	321	276
10x10	430	375
...	...	...

2.3 Gaussova eliminace s částečným výběrem hlavního prvku

Tato metoda je modifikací Gaussovy eliminační metody, která slouží ke zmenšení zaokrouhlovacích chyb. Je-li absolutní hodnota některého z dělitelů $a_{ii}^{(i-1)}$ malá ve srovnání s absolutní hodnotou prvků $a_{ki}^{(i-1)}, k > i$, potom může hrozit nebezpečí velkých zaokrouhlovacích chyb. Zaokrouhlovací chyba v absolutní hodnotě malého čísla způsobí velkou chybu v jeho převrácené hodnotě a tedy i v číslech, jimiž násobíme řádky při eliminaci. Abychom se vyhnuli dělení čísly, která jsou malá vzhledem k ostatním veličinám, provádíme výběr hlavního prvku.

Algoritmus 2.2.

1. *Hledáme rovnici, jejíž koeficient v absolutní hodnotě u x_1 je největší. Nalezenou rovnici zaměníme s první rovnicí a pomocí jejich násobků eliminujeme x_1 z ostatních rovnic.*

2. *Hledáme rovnici, jejíž koeficient v absolutní hodnotě u x_2 je největší. Nalezenou rovnici zaměníme s druhou rovnicí a pomocí jejich násobků eliminujeme x_2 z ostatních rovnic.*

3. *Obecně: v k-tém kroku eliminace hledáme mezi zbývajícími $n - k + 1$ rovnicemi tu, která má v absolutní hodnotě největší koeficient, vyměníme ji s k-tou rovnicí a pak pomocí ní eliminujeme.*

2.3.1 Počet operací porovnání

Při počítání počtu operací budeme vycházet z již známých vztahů pro Gaussovu eliminaci, viz vztahy (2.9) a (2.12). Navíc musíme uvažovat operace porovnávání pro nalezení hlavního prvku (pivota) a také operace prohození rovnic. Mějme matici o rozměrech $m \times n$. Potom počet operací porovnání je v rámci prvního sloupce matice $m - 1$, v rámci druhého sloupce $m - 2 \ldots, m - n$ v rámci n-tého sloupce. Vztah pro počet operací porovnání je uveden ve vztahu (2.13).

$$\sum_{i=1}^{n-1}(n-i) \tag{2.13}$$

2.3.2 Počet operací prohození rovnic

Počet operací vyplývá přímo z algoritmu 2.2. Počet operací prohození bude odpovídat počtu rovnic v soustavě sníženému o 1, kde počet rovnic je označen jako n.

$$n - 1 \tag{2.14}$$

Celkový počet operací pro Gaussovu eliminaci s částečným výběrem hlavního prvku znázorňuje vztah (2.15).

$$\sum_{i=1}^{n}(i^2 - 1) + \sum_{i=1}^{n}(i + 1) + \sum_{i=1}^{n}(i^2 - i) + \sum_{i=1}^{n-1} i + \sum_{i=1}^{n-1}(n - i) + (n - 1) \tag{2.15}$$

Pro lepší orientaci si shrneme význam jednotlivých členů (sčítanců) ve výše uvedeném vztahu, postupujeme zleva:

- počet operací násobení/dělení pro přímý chod Gaussovy eliminační metody,
- počet operací násobení/dělení pro zpětný chod Gaussovy eliminační metody,
- počet operací sčítání/odčítání pro přímý chod Gaussovy eliminační metody,
- počet operací sčítání/odčítání pro zpětný chod Gaussovy eliminační metody,
- počet operací porovnání,
- počet operací prohození rovnic.

Opět pro lepší přehlednost uvádíme počet operací v tabulce 2.3. Pro úplnost uvedeme příklad A.2, zadání je stejné jako u příkladu A.1 pro Gaussovu eliminační metodu.

Tabulka 2.3: Počty operací při použití Gausovy eliminační metody a částečným výběrem hlavního prvku

matice (řádky x sloupce)	součiny/podíly	součty/rozdíly	Porovnání	Prohození
2x2	6	3	1	1
3x3	17	11	3	2
4x4	36	26	6	3
5x5	65	50	10	4
6x6	106	85	15	5
7x7	161	133	21	6
8x8	232	196	28	7
9x9	321	276	36	8
10x10	430	375	45	9
...	...	...	...	...

2.4 Gaussova eliminace s úplným výběrem hlavního prvku

Při této modifikaci Gaussovy eliminace volíme v k-tém kroku za hlavní prvek ten, který je největší v absolutní hodnotě v submatici tvořené vynecháním prvních $k - 1$ řádků a sloupců v dané matici soustavy. Je snahou, abychom v k-tém kroku na pozici a_{kk} dostali

prvek s největší absolutní hodnotou. Proto tedy nepřehazujeme pouze řádky, ale i sloupce. Poznamenejme, že je potřeba dávat pozor na to, že změnou pořadí sloupců se změní pořadí výsledků a je tedy vhodné si změnu pořadí sloupců v průběhu výpočtu vhodně poznačit.

Tato metoda má ovšem vyšší časovou náročnost, protože je nutné vyhledávat největší prvek v celé submatici. Proto se spíše používá Gaussova eliminace s částečným výběrem hlavního prvku.

Při určování počtu operací budeme postupovat stejně jako u Gaussovy eliminace s částečným výběrem hlavního prvku. Opět je nutné uvažovat operace porovnávání a prohazování pořadí rovnic.

2.4.1 Počet operací porovnání

Algoritmus 2.3.

1. *Inicializujeme proměnnou* `max` *na hodnotu prvního prvku z matice soustavy.*
2. *Procházíme matici prvek po prvku a pokud narazíme na prvek, který má větší hodnotu než proměnná* `max`*, proměnnou* `max` *přepíšeme hodnotou nově nalezeného prvku.*
3. *Takto postupujeme tak dlouho, dokud neprojdeme celou matici. Na konci průchodu bude v proměnné* `max` *největší prvek.*

Příklad 2.5. *Je dána následující matice soustavy.*

$$A = \left(\begin{array}{ccc|c} 1 & 2 & 3 & 4 \\ 2 & 3 & 5 & 7 \\ 5 & 6 & 2 & 1 \end{array}\right)$$

Operace porovnání uvedeme společně s indexy prvků tabulky 2.4. Zde jsou znázorněny operace porovnání při první iteraci výpočtu. Je zřejmé, že prvek na pozici a_{32} je největší a má hodnotu 6.

Třetí řádek, kde se nachází největší prvek, prohodíme s prvním řádkem a také prohodíme první sloupcec s druhým sloupcem, abychom dostali největší prvek na pozici a_{kk}, tedy a_{11}. Matice soustavy má nyní následující tvar.

$$A = \left(\begin{array}{ccc|c} 6 & 5 & 2 & 1 \\ 3 & 2 & 5 & 7 \\ 2 & 1 & 3 & 4 \end{array}\right)$$

Pozor na to, že nyní jsou v prvním sloupci hodnoty neznámé z a v druhém sloupci hodnoty neznámé x. Při druhé iteraci výpočtu bychom vynechali první řádek a první sloupec, hledali bychom největší prvek v této submatici.

$$A = \left(\begin{array}{cc|c} 2 & 5 & 7 \\ 1 & 3 & 4 \end{array}\right)$$

Největší prvek se nachází na pozici a_{23} v původní matici a má hodnotu 5. Opět prohodíme první sloupec s druhým sloupcem a dostáváme níže uvedený tvar.

$$A = \left(\begin{array}{cc|c} 5 & 2 & 7 \\ 3 & 1 & 4 \end{array}\right)$$

Počty operací porovnání pro 2. iteraci jsou v tabulce 2.5. Další iterace by již byla zbytečná, protože vynecháním prvního sloupce a řádku z matice o rozměrech 2×2 bychom dostali pouze jednoprvkovou matici.

Tabulka 2.4: Počty operací při použití Gaussovy eliminační metody s úplným výběrem hlavního prvku při rozměru matice 3×3

krok	čísla k porovnání	indexy prvků	proměnná `max`
1	$\|1\| < \|2\|$	$a_{11} < a_{12}$	2
2	$\|2\| < \|3\|$	$a_{12} < a_{13}$	3
3	$\|3\| > \|2\|$	$a_{13} < a_{21}$	3
4	$\|3\| = \|3\|$	$a_{13} < a_{22}$	3
5	$\|3\| < \|5\|$	$a_{13} < a_{23}$	5
6	$\|5\| = \|5\|$	$a_{23} < a_{31}$	5
7	$\|5\| < \|6\|$	$a_{23} < a_{32}$	6
8	$\|6\| > \|2\|$	$a_{32} < a_{33}$	6

Tabulka 2.5: Počty operací při použití Gaussovy eliminační metody s úplným výběrem hlavního prvku při rozměru matice 2×2

krok	čísla k porovnání	indexy prvků	proměnná `max`
1	$\|2\| < \|5\|$	$a_{11} < a_{12}$	5
2	$\|5\| > \|1\|$	$a_{12} < a_{21}$	5
3	$\|5\| > \|3\|$	$a_{12} < a_{22}$	5

□

Všimněme si, že pro matici o rozměrech 3×3 je počet operací porovnání 8 a pro matici o rozměrech 2×2 je počet operací porovnání 3. Pro celkový počet operací matice o rozměrech $n \times n$ lze odvodit vztah (2.16).

$$\sum_{i=0}^{n-2} (n-i)^2 - 1 \tag{2.16}$$

2.4.2 Počet operací prohození

V průběhu první iteraci lze provést až 2 operace prohození, protože prohazujeme nejen řádky, ale i sloupce matice. Prohazování provádíme do řádu matice $n = 2$, dále již nemá smysl z důvodu, který je uveden výše. Proto je počet operací prohození

$$2 \cdot (n-1)\,. \tag{2.17}$$

Na závěr uvedeme kompletní příklad A.3.

2.5 Gauss-Jordanova eliminace

U této metody převádíme rozšířenou matici soustavy na jednotkovou matici, tj. tvar, kde nad i pod hlavní diagonálou matice budou samé nuly. Toho dosáhneme pomocí elementárních řádkových úprav, které byly zmíněné již v podkapitole 2.2. Více informací nalezneme v [2]. Algoritmus je obdobný jako u Gaussovy eliminační metody, ovšem s tím rozdílem, že zde již neprovádíme zpětnou substituci.

Řešení soustavy vyčteme z výsledné matice. Po skončení Gauss-Jordanovy eliminace bude soustava ve tvaru uvedeném níže.

$$A = \left(\begin{array}{cccc|c} 1 & 0 & \dots & 0 & v_1 \\ 0 & 1 & \dots & 0 & v_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \dots & 1 & v_n \end{array}\right)$$

2.5.1 Počty operací násobení a dělení

V průběhu první iterace bude potřeba

- n násobení/dělení pro násobení prvího řádku číslem $1/a_{11}$,
- n násobení/dělení pro získání nuly na pozici a_{21}.

Tyto operace je nutné provést se všemi $(n-1)$ řádky. Počet násobení/dělení pro první iteraci je tedy

$$n + (n-1)n\,.$$

Druhá iterace vyžaduje

- $n-1$ násobení/dělení pro násobení prvího řádku číslem $\frac{1}{a_{22}}$,
- $n-1$ násobení/dělení pro získání nuly na pozici a_{21}.

Tyto operace provedeme se všemi $(n-1)$ řádky. Počet operací násobení/dělení je

$$(n-1) + (n-1)(n-1)\,.$$

Celkový počet operací násobení/dělení pro Gauss-Jordanovu eliminaci je uveden ve vztahu (2.18).

$$\sum_{i=1}^{n}(i + (n-1)i) \tag{2.18}$$

2.5.2 Počty operací sčítání a odčítání

Pro první iteraci je potřeba

- n sčítání/odčítání pro získání nuly na pozici a_{21}.

Tyto operace provedeme se všemi $(n-1)$ řádky. Celkový počet operací sčítání/odčítání pro první iteraci je

$$(n-1)n\,.$$

V druhé iteraci potřebujeme získat nuly nad i pod prvkem matice a_{22}. Nezabýváme se již prvním řádkem ani sloupcem. Počet sčítání/odčítání je

$$(n-1)(n-1)\,.$$

Použití Gauss-Jordanovy eliminace ukazuje příklad A.4. Celkový počet operací sčítání/odčítání uveden níže, viz vztah (2.19).

$$\sum_{i=1}^{n}(n-1)i \tag{2.19}$$

Celkový počet operací je znázorněn v tabulce 2.6.

Tabulka 2.6: Počty operací při použití Gauss-Jordanovy eliminační metody

matice (řádky x sloupce)	součiny/podíly	součty/rozdíly
2x2	6	3
3x3	18	12
4x4	40	30
5x5	75	60
6x6	126	105
7x7	196	168
8x8	288	252
9x9	405	360
10x10	550	495
...	...	...

2.6 Inverzní matice

Inverzní matici můžeme počítat pouze tehdy, pokud je matice čtvercová a také regulární. Inverzní matici k matici **A** značíme $\mathbf{A^{-1}}$. Pro inverzní matici platí vztahy

$$(A^{-1}) \cdot A^{-1} = A \tag{2.20}$$

a také

$$(A^{-1}) \cdot A = E, \tag{2.21}$$

kde **E** je jednotková matice. Vypočítat inverzní matici můžeme dvěma způsoby:

1. Gauss-Jordanovou eliminační metodou,
2. pomocí determinantů.

Postupně si ukážeme obě zmíněné metody.

2.6.1 Výpočet inverzní matice pomocí Gauss-Jordanovy eliminační metody

Postup výpočtu lze shrnout do těchto kroků, jak ukazuje algoritmus 2.4.

Algoritmus 2.4. *1. Napíšeme matici soustavy, ke které chceme nalézt inverzní matici a vedle ní umístíme matici jednotkovou (stejného řádu jako je matice soustavy).*

2. Pomocí ekvivaletních řádkových úprav se snažíme dostat jednotkovou matici na levou stranu. Na straně pravé se poté nachází výsledná inverzní matice. Každá úprava se provádí na obou maticích současně.

3. Pro nalezení vektoru neznámých proměnných musíme vypočítat $x = A^{-1} \cdot b$, tedy vynásobit získanou inverzní matici A^{-1} s vektorem pravých stran **b**.

4. Správnost výsledku lze ověřit zkouškou tak, že vynásobíme původní matici soustavy s její inverzní maticí, měli bychom dostat jednotkovou matici.

Schématické znázornění postupu výpočtu lze vidět níže.

$$M = \left(\begin{array}{c|c} A & E_n \end{array} \right) = \left(\begin{array}{cccc|cccc} a_{11} & a_{12} & \dots & a_{1n} & 1 & 0 & \dots & 0 \\ 0 & a_{22} & \dots & a_{2n} & 0 & 1 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & a_{mn} & 0 & 0 & \dots & 1 \end{array} \right) \rightarrow$$

$$\rightarrow \left(\begin{array}{c|c} E_n & A \end{array} \right) = \left(\begin{array}{cccc|cccc} 1 & 0 & \dots & 0 & a_{11} & a_{12} & \dots & a_{1n} \\ 0 & 1 & \dots & 0 & 0 & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 1 & 0 & 0 & \dots & a_{mn} \end{array} \right)$$

U Gaussovy eliminační metody jsme potřebovali získat nulové pozice pouze pod hlavní diagonálou. Zde he nutné splnit následují úkoly:

- získat nuly pod i nad hlavní diagonálou,
- získat jedničky na hlavní diagonále.

Počet operací násobení a dělení

Na začátek uvedeme kompletní příklad A.5. Počet operací násobení pro nalezení vektoru neznámých je

$$n \cdot n \,. \tag{2.22}$$

Počet operací sčítání/odčítání je

$$(n-1) \cdot n \,. \tag{2.23}$$

Podíváme se nyní, kolik operací násobení/dělení bude potřeba pro první iteraci Gauss-Jordanovy metody. Rozměry matice jsou $2n \times n$. Z příkladu A.5 z výchozího tvaru soustavy si lze všimnout, že na každém řádku je v nejhorším možném případě $n+1$ členů nenulových, ostatní jsou nulové. Při provádění výpočtu není třeba nulové členy uvažovat.

Konkrétně na začátku máme na pravé straně jednotkovou matici řádu $n = 3$, která v každém řádku obsahuje dvě nuly. Po prvním kroku výpočtu jsme získali nulu na pozici a_{21} a naopak ubyla nula na pozici a_{24}. V dalším kroku jsme získali nulu na pozici a_{31} a zároveň ubyla nula na pozici a_{34}. Nyní máme pod hlavní diagonálou již samé nuly. Při získávání nul nad hlavní diagonálou je situace obdobná, tedy také se nám nuly postupně přesouvají na jiné indexy.

□

Počet operací násobení/dělení je stejný jako pro Gauss-Jordanovu eliminaci, viz (2.18).

$$\sum_{i=1}^{n} (i + (n-1)i) \tag{2.24}$$

Tabulka 2.7: Počty operací při počítání inverzní matice pomocí Gauss Jordanovy eliminace

matice (řádky x sloupce)	součiny/podíly	součty/rozdíly
2x2	14	5
3x3	45	24
4x4	104	66
5x5	200	140
6x6	342	255
7x7	539	420
8x8	800	644
9x9	1134	936
10x10	1550	1305
...	...	...

Počet operací sčítání a odčítání

Jak bylo řečeno výše, lze vycházet ze vztahů pro Gauss-Jordanovu eliminaci, viz vztah (2.19). Počet operací sčítání/odčítání je uveden ve vztahu (2.25).

$$\sum_{i=1}^{n}(n-1)i \tag{2.25}$$

Na závěr opět uvádíme tabulku 2.7 shrnující počet operací.

2.6.2 Výpočet inverzní matice pomocí determinantů

Nechť **A** je čtvercová matice řádu n. Potom k ní existuje inverzní matice A^{-1} a platí vztah (2.26).

$$A^{-1} = \frac{1}{\det A}\cdot(A_{ij})^T = \frac{1}{\det A}\cdot\begin{pmatrix} a_{11} & a_{21} & \dots & a_{n1} \\ a_{12} & a_{22} & \dots & a_{n2} \\ \vdots & \vdots & \ddots & \vdots \\ a_{1n} & a_{2n} & \dots & a_{nn} \end{pmatrix} \approx$$

$$\approx \begin{pmatrix} \frac{|A_{11}|}{|A|} & \frac{|A_{21}|}{|A|} & \dots & \frac{|A_{n1}|}{|A|} \\ \frac{|A_{12}|}{|A|} & \frac{|A_{22}|}{|A|} & \dots & \frac{|A_{n2}|}{|A|} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{|A_{1n}|}{|A|} & \frac{|A_{2n}|}{|A|} & \dots & \frac{|A_{nn}|}{|A|} \end{pmatrix}, \tag{2.26}$$

kde čísla A_{ij} jsou algebraickými doplňky k prvkům $a_{ij} \in \{1,\dots,n\}$ matice A. Algebraickým doplňkem prvku a_{ij} matice $A \equiv (a_{ij})$ nazýváme číslo $A_{ij} = (-1)^{i+j}M_{ij}$, kde M_{ij} je subdeterminant (minor) vzniklý z matice A vynecháním i-tého řádku a j-tého sloupce. Vzorec pro výpočet inverzního prvku je uveden ve vztahu 2.27.

$$a_{ij}^{-1} = \frac{(-1)^{i+j}\cdot|A_{j,i}|}{|A|} \tag{2.27}$$

Výpočet počtu operací rozdělíme do těchto bodů.

- Pro determinant soustavy řádu n bude potřeba určitý počet operací, v závislosti na jeho řádu (pro determinanty řádu ≤ 3 používáme Sarrusovo pravidlo, pro determinanty řádu > 3 Laplaceův rozvoj determinantu).
- Ke každému prvku matice soustavy budeme vytvářet subdeterminant. Celkem tedy vždy vytvoříme právě $n \cdot n$ determinantů řádu $n - 1$.
- Dále ve vzorci (2.27) vidíme podíl. Těch bude tolik, kolik je celkem subdeterminantů, tedy $n \cdot n$.

Pro řešení soustavy je nutné provést výpočet podle vztahu $x = A^{-1} \cdot b$, podobně jako u počítání inverzní matice pomocí Gauss-Jordanovy eliminační metody. Počet operací je uveden ve vztahu (2.22).

Počet operací násobení a dělení

Nyní se podíváme podrobněji na počty operací násobení a dělení. Proces zjišťování počtu operací rozdělíme do následujících částí:

1. počet operací násobení a dělení pro determinant soustavy (budeme ho značit $\det A$),
2. počet operací násobení a dělení pro determinanty řádu $n - 1$,
3. počet operací dělení.

Ad 1. Počet operací násobení a dělení pro determinant soustavy

Při vyjádření operací vycházíme z již známých vztahů, které byly zjištěny u Cramerova pravidla (viz kapitola 2.1.1), ovšem v upravené podobě. Počet operací násobení pro determinant o rozměrech 4×4 je 56. Jedná se o konstantu, kterou jsme zavedli již u Cramerova pravidla. Počet operací násobení pro determinant o rozměrech 5×5 je

$$56n + 2n\,.$$

Všimněme si, že oproti vzorcům, které jsme uvedli u Cramerova pravidla neuvažujeme v těchto vzorcích členy za hranatou závorkou, tedy $(n - 1) + n$. To proto, že pro Cramerovo pravidlo jsme vytvářeli $(n + 1)$ determinantů řádnu n, počet podílů pro získání řešení soustavy rovnic byl n. Počty operací násobení/dělení pro determinant soustavy o rozměrech 6×6 je

$$n[2n + 2(n - 1) + (n - 1)56]\,, \text{kde } n{=}6.$$

Počet operací násobení/dělení pro determinant o rozměrech 7×7 je

$$n[2n + 2(n - 1) + 2(n - 2) + (n - 2)56]\,, \text{kde } n{=}7,$$

pro determinant o rozměrech 8×8

$$n[2n + 2(n - 1) + 2(n - 2) + 2(n - 3) + (n - 3)56]\,, \text{kde } n{=}8,$$

pro determinant o rozměrech 9×9

$$n[2n + 2(n - 1) + 2(n - 2) + 2(n - 3) + 2(n - 4) + (n - 4)56]\,, \text{kde } n{=}9,$$

pro determinant o rozměrech 10×10

$$n[2n + 2(n-1) + 2(n-2) + 2(n-3) + 2(n-4) + 2(n-5) + (n-5)56]\,, \text{kde } n{=}10.$$

Obecný vztah pro vytvoření vzorce pro daný řád je uveden ve vztahu (2.28).

$$n \cdot \left[\sum_{i=0}^{n-5} 2 \cdot (n-i) + (n-(n-5)) \cdot 56 \right] \,, \text{ kde } n > 5 \tag{2.28}$$

Ad 2. Počet operací násobení a dělení pro determinanty řádu $n-1$

V případě, že musíme pro výpočet determinantu použít Laplaceův rozvoj, můžeme při počítání determinantů řádu $n-1$ ušetřit n operací sčítání/odčítání i násobení/dělení z celkového množství těchto determinantů, protože právě těchto n determinantů bude již spočteno v rámci Laplaceova rozvoje. Například pro determinant čtvrtého řádu budeme mít v Laplaceově rozvoji 4 determinanty třetího řádu a právě tyto 4 determinanty již nebudeme muset počítat.

Počet determinantů řádu $n-1$, které bude nutné vypočítat je

$$(n \cdot n) - n = n^2 - n\,. \tag{2.29}$$

Při výpočtu determinantu řádu 4 a výše využíváme dříve vypočtených hodnot. Počet operací násobení, který je uveden v předchozím kroku pro determinant soustavy, je využit v aktuálním kroku pro výpočet počtu operací násobení u determinantů řádu $n-1$. Jinými slovy: determinant soustavy z předchozího kroku slouží k vytvoření subdeterminantů v kroku následujícím. Hodnotu získanou v předchozím kroku násobíme výrazem $n^2 - n$.

Ad 3. Počet operací podílů

Podílů je tolik, kolik je determinantů řádu $n-1$, tedy n^2.

Počet operací sčítání a odčítání

Zjišťování počtu operací rozdělíme na 2 sekce:

1. počet operací sčítání/odčítání pro determinant soustavy,
2. počet operací sčítání/odčítání pro subdeterminanty (determinanaty řádu $n-1$).

Ad 1. Počet operací sčítání/odčítání pro determinant soustavy

Podobně jako u operací násobení/dělení i zde využíváme upravené vztahy pro součet/rozdíl odvozené ze vztahů pro Cramerovo pravidlo (viz sekce 2.1.1). Neuvažujeme členy za hranatými závorkami $(n+1)$.

Počet operací sčítání/odčítání pro determinant o rozměrech 4×4 je konstanta 23. Počet operací sčítání/odčítání pro determinant o rozměrech 5×5 je

$$\{(n-1) + n \cdot 23\}\,.$$

Počet operací sčítání/odčítání pro determinant o rozměrech 6×6 je

$$\{(n-1) + n \cdot [(n-2) + (n-1) \cdot 23]\}\,.$$

Počet operací sčítání/odčítání pro determinant o rozměrech 7×7 je

$$\{(n-1)+n\cdot[(n-2)+(n-1)\cdot[(n-3)+(n-2)\cdot 23]]\}\,.$$

Počet operací sčítání/odčítání pro determinant o rozměrech 8×8 je

$$\{(n-1)+n\cdot[(n-2)+(n-1)\cdot[(n-3)+(n-2)\cdot[(n-4)+(n-3)\cdot 23]]]\}\,.$$

Počet operací sčítání/odčítání pro determinant o rozměrech 9×9 je

$$\{(n-1)+n\cdot[(n-2)+(n-1)\cdot[(n-3)+(n-2)\cdot[(n-4)+(n-3)\cdot$$
$$\cdot[(n-5)+(n-4)\cdot 23]]]]\}\,.$$

Počet operací sčítání/odčítání pro determinant o rozměrech 10×10 je

$$\{(n-1)+n\cdot[(n-2)+(n-1)\cdot[(n-3)+(n-2)\cdot[(n-4)+(n-3)\cdot[(n-5)+(n-4)\cdot$$
$$\cdot[(n-6)+(n-5)\cdot 23]]]]]\}\,.$$

Pro tvorbu obecného vztahu počtu operací násobení/dělení pro determinant daného řádu využíváme stejného postupu jako tomu bylo u Cramerova pravidla v sekci 2.1.

$$\{(n-1)+n\cdot[slozenaZavorka]\},$$

kde *slozenaZavorka* je obsah složené závorky z předešlého kroku snížený o 1. Tímto mechanismem dochází k „množení závorek“.

Ad 1. Počet operací sčítání/odčítání pro subdeterminanty

Postup výpočtu je stejný jako u počtu operací pro subdeterminanty pro násobení/dělení (viz sekce 2.6.2). Na závěr uvedeme kompletní příklad A.6. Tabulka 2.8 znázorňuje počty opercí, v tabulce 2.9 potom můžeme vidět, kolik je potřeba operací při získávání výsledků pomocí vztahu $x = A^{-1} \cdot b$.

Tabulka 2.8: Počty operací při počítání inverzní matice pomocí determinantů

matice (řádky x sloupce)	součiny/podíly	součty/rozdíly
2x2	6	1
3x3	39	14
4x4	216	83
5x5	1435	579
6x6	10548	4289
7x7	78365	35237
8x8	126592	322503
9x9	194463	3265847
10x10	287300	36287909
...	...	...

Tabulka 2.9: Počty operací pro získání výsledků při počítání s inverzní maticí

matice (řádky x sloupce)	Počet součinů	Počet součtů
2x2	4	2
3x3	9	6
4x4	16	12
5x5	25	20
6x6	36	30
7x7	49	42
8x8	64	56
9x9	81	72
10x10	100	90
...	...	...

Kapitola 3

Iterační metody

Iterační metody využíváme k řešení lineárních rovnic pomocí postupného přibližování se k výsledku. Volíme počáteční aproximaci přibližného řešení a určitým způsobem ji v každém kroku iterační metody zpřesňujeme. K řešení se ovšem přibližujeme limitně. Je potřeba tedy určit, kdy výpočet ukončíme. Výpočet lze ukončit pokud dosáhneme požadované přesnosti nebo je předem dán počet kroků, který se bude provádět. Výsledkem je přibližné řešení soustavy. Podrobněji se iteračními metodami zabývají publikace [4, 13, 14, 10].

3.1 Jacobiho metoda

Při řešení soustavy pomocí Jacobiho metody postupujeme následujícím způsobem.

3.1.1 Ověření konvergence Jacobiho metody

Je-li matice soustavy ostře sloupcově nebo řádkově dominantní, potom Jacobiho metoda konverguje.

- Řádkově dominantní: v každém řádku matice je absolutní hodnota prvku na hlavní diagonále větší než součet absolutních hodnot všech ostatních prvků v onom řádku.
- Sloupcově dominantní: v každém sloupci matice je absolutní hodnota prvku na diagonále větší než součet absolutních hodnot všech ostatních prvků v onom sloupci.

Pokud podmínka ostrosti není splněna, potom není zaručeno splnění podmínky konvergence. Poznamenejme, že tento způsob ověřování konvergence metody není vhodný pro velké soustavy. V těchto případech je potom lepší stanovit maximální počet kroků metody nebo požadovanou přesnost.

3.1.2 Iterační vztahy

Máme následující soustavu lineárních rovnic zapsanou v obecném tvaru.

$$\begin{aligned} a_{11}x1 + a_{12}x_2 + \cdots + a_{1n}x_n &= b_1 \\ a_{21}x1 + a_{22}x_2 + \cdots + a_{2n}x_n &= b_2 \\ &\vdots \\ a_{n1}x1 + a_{n2}x_2 + \cdots + a_{nn}x_n &= b_n \end{aligned} \tag{3.1}$$

Nyní postupujeme tak, že z první rovnice vyjádříme x_1, z druhé rovnice x_2, až z n-té rovnice vyjádříme x_n. Vztahy (3.2) nazýváme iterační vztahy.

$$\begin{aligned} x_1 &= \frac{1}{a_{11}}(b_1 - a_{12}x_2 - a_{13}x_3 - \ldots - a_{1n}x_n) \\ x_2 &= \frac{1}{a_{22}}(b_2 - a_{21}x_1 - a_{23}x_3 - \ldots - a_{2n}x_n) \\ &\vdots \\ x_n &= \frac{1}{a_{nn}}(b_n - a_{n1}x_1 - a_{n2}x_2 - \ldots - a_{nn-1}x_{n-1}) \end{aligned} \tag{3.2}$$

3.1.3 Volba počáteční apoximace

Volíme libovolnou počáteční aproximaci $x^0 = (x_1^0, x_2^0, \ldots x_n^0)^T$. Zvolené hodnoty počáteční aproximace dosadíme do pravé strany rovnice (3.2). Tímto způsobem dostaneme novou aproximaci řešení $x^1 = (x_1^1, x_2^1, \ldots x_n^1)^T$. Získané hodnoty opět dosadíme do pravé strany rovnice 3.2, atd. Pro názornost uvedeme příklad 3.1.

Příklad 3.1. *Pomocí Jacobiho metody vyřešte následující soustavu rovnic. Předpokládejte přesnost ε=0,01.*

$$\begin{aligned} 10x_1 - 2x_2 + 3x_3 &= 20 \\ 4x_1 - 8x_1 + x_3 &= 25 \\ x_1 + 4x_2 + 16x_3 &= 30 \end{aligned}$$

Nejprve ověříme konvergenci metody. Matice soustavy je

$$A = \left(\begin{array}{ccc|c} 10 & -2 & 3 & 20 \\ 4 & -8 & 1 & 25 \\ 1 & 4 & 16 & 30 \end{array}\right).$$

Zkusíme ověřit, zda je matice ostře řádkově diagonálně dominantní.

$$\begin{aligned} |10| &> |-2| + |3| \\ |-8| &> |4| + |1| \\ |16| &> |1| + |4| \end{aligned}$$

Podmínka konvergence je tedy splněna. Dalším krokem je tvorba iteračních vztahů.

$$\begin{aligned} x_1^{(r+1)} &= \frac{1}{10}(20 + 2x_2^{(r)} - 3x_3^{(r)}) \\ x_2^{(r+1)} &= \frac{-}{1}8(25 - 4x_1^{(r)} - x_3^{(r)}) \\ x_3^{(r+1)} &= \frac{1}{16}(30 + x_1^{(r)} - 4x_2^{(r)}) \end{aligned}$$

Zvolíme počáteční aprosimaci $x = (0, 0, 0)^T$. Aproximace řešení budeme zapisovat pro přehlednost do tabulky 3.1.

Přenost je stanovena na 0,01 a výpočet můžeme po sedmi iteracích ukončit, protože

$$\begin{aligned} |x_1^7 - x_1^6| &< 0,01\,, \\ |x_2^7 - x_2^6| &< 0,01\,, \\ |x_3^7 - x_3^6| &< 0,01. \end{aligned}$$

Tabulka 3.1: Aproximace řešení pomocí Jacobiho metody

r	x_1^r	x_2^r	x_3^r
0	0	0	0
1	2	-3,125	1,875
2	0,8125	-1,8906	2,5313
3	0,8625	-2,4023	2,2969
4	0,8305	-2,4066	2,4217
5	0,7922	-2,4070	2,4247
6	0,7912	-2,4258	2,4277
7	0,7867	-2,4263	2,4320

□

Všimněme si, že záleží na pořadí rovnic. Kdybychom například prohodili druhou a třetí rovnici, podmínka konvergence metody by již nebyla splněna.

Příklad 3.2. *Pomocí Jacobiho metody vyřešte následující soustavu rovnic. Předpokládejte přesnost ε=0,01.*

$$\begin{aligned} 10x_1 - 2x_2 + 3x_3 &= 20 \\ x_1 + 4x_2 + 16x_3 &= 30 \\ 4x_1 - 8x_1 + x_3 &= 25 \end{aligned}$$

Nejdříve ověříme konvergenci metody. Matice soustavy je

$$A = \left(\begin{array}{ccc|c} 10 & -2 & 3 & 20 \\ 1 & 4 & 16 & 30 \\ 4 & -8 & 1 & 25 \end{array}\right).$$

Zkusíme ověřit, zda je matice soustavy ostře řádkově diagonálně dominantní.

$$\begin{aligned} |10| &> |-2| + |3| \\ |4| &< |1| + |16| \\ |1| &< |4| + |-8| \end{aligned}$$

Podmínka konvergence není splněna. Není tedy zaručeno, že Jacobiho metoda bude konvergovat.

□

3.1.4 Počty operací u ověřování konvergence metody

Začneme počty operací, které je potřeba provést ještě před započetím výpočtu. Jedná se o fázi, kdy zjišťujeme podmínky konvergence. Vypočteme počty operací, které je nutné provést při ověřování, zda je matice soustavy ostře diagonálně dominantní.

Pro matici 3×3

$$A = \left(\begin{array}{ccc|c} 10 & -2 & 3 & 20 \\ 4 & -8 & 1 & 25 \\ 1 & 4 & 16 & 30 \end{array}\right)$$

$$
\begin{aligned}
|10| &> |-2| + |3|\,, \\
|-8| &> |4| + |1|\,, \\
|16| &> |1| + |4|\,.
\end{aligned}
$$

Byly tedy potřeba 3 operace porovnání a 3 operace součtu

Pro matici 4×4

$$
A = \left(\begin{array}{cccc|c}
10 & -2 & 3 & 2 & 20 \\
4 & -8 & 1 & 1 & 25 \\
1 & 4 & 16 & 3 & 30 \\
2 & 7 & 3 & 20 & 45
\end{array}\right)
$$

$$
\begin{aligned}
|10| &> |-2| + |3| + |2|\,, \\
|-8| &> |4| + |1| + |1|\,, \\
|16| &> |1| + |4| + |3|\,, \\
|20| &> |2| + |7| + |3|\,.
\end{aligned}
$$

Zde je zapotřebí provést 4 operace porovnání a 8 operací součtů.

Pro matici 5×5

$$
A = \left(\begin{array}{ccccc|c}
10 & -2 & 3 & 2 & 1 & 20 \\
4 & -8 & 1 & 1 & 2 & 25 \\
1 & 4 & 16 & 3 & 5 & 30 \\
2 & 7 & 3 & 20 & 4 & 45 \\
3 & 2 & 7 & 5 & 22 & 48
\end{array}\right)
$$

$$
\begin{aligned}
|10| &> |-2| + |3| + |2| + |1|\,, \\
|-8| &> |4| + |1| + |1| + |2|\,, \\
|16| &> |1| + |4| + |3| + |5|\,, \\
|20| &> |2| + |7| + |3| + |4|\,, \\
|22| &> |3| + |2| + |7| + |5|\,.
\end{aligned}
$$

V případě rozměrů matice 5×5 je nutné provést 5 operací porovnání a 15 operací součtů. Vidíme, že počet operací porovnání bude vždy právě n a to bez ohledu na to, zda ověřujeme sloupcovou či řádkovou diagonální dominantnost, protože matice soustavy je vždy čtvercová.

Počet členů na pravé straně nerovnice je vždy právě $n-1$, protože diagonální člen, se kterým porovnávání provádíme, je na straně levé. Těchto $n-1$ mimodiagonálních členů lze propojit pomocí $n-2$ operací sčítání. Celkově tedy máme $n-2$ operací sčítání pro jeden řádek matice soustavy. Celkový počet řádků matice je n. Výsledný vztah pro počet operací sčítání je tedy

$$
(n-2)n\,. \tag{3.3}
$$

3.1.5 Počet operací pro iterační vztahy metody

Iterační vztahy pro soustavu o třech neznámých jsme uvedli v příkladu 3.1. Lze si všimnout, že potřebujeme 6 operací násobení/dělení a 6 operací sčítání/odečítání.

Iterační vztahy pro soustavu o čtyřech neznámých uvedeme na smyšleném příkladu níže.

$$\begin{aligned} x_1^{(r+1)} &= \frac{1}{10}(20 + 2x_2^{(r)} - 3x_3^{(r)} - x_4^{(r)}) \\ x_2^{(r+1)} &= -\frac{1}{8}(25 - 4x_1^{(r)} - x_3^{(r)} + 2x_4^{(r)}) \\ x_3^{(r+1)} &= \frac{1}{16}(30 + x_1^{(r)} - 4x_2^{(r)} - x_4^{(r)}) \\ x_4^{(r+1)} &= \frac{1}{5}(32 + 3x_1^{(r)} - 2x_2^{(r)} - 2x_4^{(r)}) \end{aligned}$$

Bylo potřeba 8 operací násobení/dělení a 12 operací sčítání/odečítání.

Pro výpočet jedné neznáme proměnné Jacobiho metodou, což odpovídá jedné buňce tabulky 3.1, jsou potřeba 2 operace násobení/dělení. Celkový počet neznámých je vždy n. Proto tedy celkový vztah pro počet operací násobení/dělení potřebných na jednu iteraci Jacobiho metody je $2n$.

Tabulka 3.2: Počty operací při ověřování, zda je matice soustavy ostře diagonálně dominantní

matice (řádky x sloupce)	porovnání	součty
2x2	2	0
3x3	3	3
4x4	4	8
5x5	5	15
6x6	6	24
7x7	7	35
8x8	8	48
9x9	9	63
10x10	10	80
...	...	...

Pro operace sčítání/odčítání postupujeme podobně. Pro výpočet jedné neznáme proměnné Jacobiho metodou, což odpovídá jedné buňce tabulky 3.1 je potřeba $(n-1)$ operací sčítání/odčítání. Celkový počet neznámých je vždy n. Proto tedy celkový vztah pro počet operací sčítání/odčítání potřebných na jednu iteraci Jacobiho metody je

$$(n-1)n\,. \tag{3.4}$$

Celkový počet operací pro výpočet jedné aproximace řešení (tedy jednoho řádku tabulky 3.1) znázorňuje vztah (3.5).

$$2n + (n-1)n \tag{3.5}$$

Tabulky 3.2 a 3.3 platí jak pro Jacobiho metodu, tak i pro Gauss-Seidelovu metodu. V tabulce 3.3 jsou uvedeny počty operací potřebné pro získání jedné aproximace řešení – tedy jednoho řádku tabulky. Počet operací závisí na přesnosti, které chceme dosáhnout. Pokud je stanoven přímo počet kroků metody, potom stačí hodnoty pro daný rozměr soustavy v tabulce 3.3 vynásobit počtem kroků.

Tabulka 3.3: Počty operací pro Jacobiho a Gauss-Seidelovu metodu

matice (řádky x sloupce)	součiny/podíly	součty/rozdíly
2x2	4	2
3x3	6	6
4x4	8	12
5x5	10	20
6x6	12	30
7x7	14	42
8x8	16	56
9x9	18	72
10x10	20	90
...	...	...

3.2 Gauss-Seidelova metoda

Gauss-Seidelova metoda je založena na stejném principu jako metoda Jacobiho. Odlišnost je v tom, že k výpočtu další aproximace řešení používá vždy nejnovější dostupné hodnoty $x_1, x_2, \ldots, x_n$, které jsou k dispozici. Člen $x_1^{(r+1)}$ vypočteme stejně jako u Jacobiho metody, při výpočtu $x_2^{(r+1)}$ ale ihned využíváme právě vypočteného $x_1^{(r+1)}$, při výpočtu $x_3^{(r+1)}$ využíváme nové hodnoty $x_1^{(r+1)}$ i $x_2^{(r+1)}$, atd.

3.2.1 Ověření konvergence

- Je-li matice soustavy ostře řádkově nebo sloupcově diagonálně dominantní, Gauss-Seidelova metoda konverguje.
- Je-li matice soustavy ve tvaru $Ax = b$ pozitivně definitní, Gauss-Seidelova metoda konverguje.

Pozitivní definitnost matice lze zjistit dvěma způsoby:

1. Sylvestrovo kritérium,
2. vlastní čísla.

Sylvestrovo kritérium

Je-li matice soustavy ve tvaru $Ax = b$ pozitivně definitní, Gauss-Seidelova metoda konverguje. Nechť

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

je symetrická matice. Pak **A** je pozitivně definitní právě tehdy, když

$$\left| \; a_{11} \; \right| > 0, \begin{vmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{vmatrix} > 0, \begin{vmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{vmatrix} > 0, \ldots, \begin{vmatrix} a_{11} & \ldots & a_{1n} \\ a_{21} & \ldots & a_{2n} \\ \vdots & \vdots & \ddots \\ a_{n1} & \ldots & a_{nn} \end{vmatrix} > 0 \,.$$

Tedy matice je pozitivně definitní, jsou-li všechny hlavní subdetetminanty kladné. Hlavními subdeterminanty rozumíme determinanty submatic 1×1, 2×2,..., $n\times n$ vyšetřované matice **A**.

Příklad 3.3. *Ověřte, zda je daná matice pozitivně definitní.*

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

Subdeterminant 1×1 je

$$\begin{vmatrix} 3 \end{vmatrix} = 3 > 0\,,$$

subdeterminant 2×2 je

$$\begin{vmatrix} 3 & 2 \\ 6 & 5 \end{vmatrix} = 3 > 0\,,$$

subdeterminant 3×3 je

$$\begin{vmatrix} 3 & 2 & 1 \\ 6 & 5 & 4 \\ 3 & 1 & 2 \end{vmatrix} = 9 > 0\,.$$

Matice je tedy pozitivně definitní. Je patrné, že musíme vždy vytvářet právě n determinantů a vyčíslit je. Vztahy pro určení počtu operací lze vyčíst z podkapitoly 2.1.

Vlastní čísla

Čtvercová matice **A** je pozitivně definitní právě tehdy, když všechna její vlastní čísla jsou kladná. Vlastní čísla získáme z výpočtu kořenů charakteristické rovnice. Tuto rovnici sestavíme tak, že determinant matice **A** vynásobíme s jednotkovou maticí E s konstantou λ.

$$|A - \lambda \cdot E| \tag{3.6}$$

Tento výraz položíme roven nule a dostáváme

$$|A - \lambda \cdot E| = 0\,. \tag{3.7}$$

Pokud pro kořeny charakteristické rovnice platí podmínka

$$Re(\lambda) > 0 \tag{3.8}$$

potom je matice **A** pozitivně definitní.

Příklad 3.4. *Najděte kořeny charakteristické rovnice. Je dána následjující matice.*

$$A = \begin{pmatrix} 1 & 2 \\ 0 & 4 \end{pmatrix}$$

Upravíme ji dle vztahu 3.7 a dostaneme

$$\begin{vmatrix} 1-\lambda & 2 \\ 0 & 4-\lambda \end{vmatrix} = 0\,.$$

Vyčíslíme determinant

$$(1-\lambda)\cdot(4-\lambda)-2\cdot 0\,.$$

Dostaneme charakteristickou rovnici

$$\lambda^2-5\lambda+4=0\,.$$

Pomocí vzorce pro výpočet kvadratické rovnice získáváme kořeny charakteristické rovnice.

$$\lambda_{1,2}=\frac{5\pm\sqrt{25-4\cdot 1\cdot 4}}{2}$$
$$\lambda_1=4,\quad \lambda_2=1$$

Oba kořeny charakteristické rovnice jsou kladné, a tedy daná matice je pozitivně definitní.

□

3.2.2 Iterační vztahy

Iterační vztahy mají stejný tvar jako u Jacobiho metody.

$$\begin{aligned}
x_1 &= \frac{1}{a_{11}}(b_1-a_{12}x_2-a_{13}x_3-\ldots-a_{1n}x_n)\\
x_2 &= \frac{1}{a_{22}}(b_2-a_{21}x_1-a_{23}x_3-\ldots-a_{2n}x_n)\\
&\vdots\\
x_n &= \frac{1}{a_{nn}}(b_n-a_{n1}x_1-a_{n2}x_2-\ldots-a_{nn-1}x_{n-1})
\end{aligned}$$

Počet operací pro iterační vztahy je totožný se vztahy pro Jacobiho metodu, jak bylo uvedeno v podkapitole 3.1.5.

3.2.3 Volba počáteční aproximace

Opět vycházíme ze vztahů v kapitole 3.1. Postup výpočtu je naprosto totožný s principem uvedeným u Jacobiho metody, rozdíl spočívá v rychlosti konvergence metod. Gauss-Seidelova metoda konverguje rychleji. Na závěr uvedeme příklad 3.5 na výpočet pomocí Gauss-Seidelovy metody.

Příklad 3.5. *Pomocí Gauss-Seidelovy metody vyřešte následující soustavu rovnic. Předpokládejte přesnost ε=0,01.*

$$\begin{aligned}
10x_1-2x_2+3x_3&=20\\
4x_1-8x_1+x_3&=25\\
x_1+4x_2+16x_3&=30
\end{aligned}$$

Zadání příkladu je stejné jako u příkladu 3.1 na Jacobiho metodu, aby bylo možné provést srovnání obou metod. Uvedeme tedy pouze tabulku 3.4.

Přesnost je stanovena na 0,01 a výpočet nyní můžeme po šesti iteracích ukončit, protože

$$\begin{aligned}
|x_1^6-x_1^5|&<0,01\,,\\
|x_2^6-x_2^5|&<0,01\,,\\
|x_3^6-x_3^5|&<0,01\,.
\end{aligned}$$

□

Tabulka 3.4: Aproximace řešení pomocí Gauss-Seidelovy metody

r	x_1^r	x_2^r	x_3^r
0	0	0	0
1	2	2,125	1,2188
2	2,0597	-1,9428	2,2320
3	0,9418	-2,3751	2,4099
4	0,8020	-2,4228	2,4306
5	0,7863	-2,4280	2,4329
6	0,7845	-2,4286	2,4331

Kapitola 4

Diferenciální rovnice

Deferenciální rovnice (dále DR) je rovnice, kde neznámou je funkce a rovnice obsahuje i derivaci této funkce. Na začátek uvádíme definici 4.1. Podrobnější informace jsou uvedeny v [6, 4, 14, 11].

Definice 4.1. *Rovnice ve tvaru* $F(y^n, y^{n-1}, \ldots, y', y, x) = 0$ *se nazývá* ***diferenciální rovnice n-tého řádu*** *pro funkci* $y = y(t)$. *Speciálně je*

$$F(y', y, t) = 0 \quad nebo \quad y' = f(t, y)$$

diferenciální rovnicí prvního řádu. Řád *diferenciální rovnice je řád nejvyšší proměnné hledané funkce* $y(x)$.

Rovnice, které jsme uvedli v definici 4.1, se nazývají *obyčejné diferenciální rovnice.* Diferenciální rovnice můžeme rozdělit podle derivací následovně:

- obyčejné DR – obsahují derivace hledané funkce podle jedné proměnné,
- parciální DR – obsahují derivace hledané funkce podle více proměnných.

Abychom získali jednoznačné řešení soustavy je nutné specifikovat pro každou rovnici počáteční podmínku.

Definice 4.2. *Nechť* $x_0, y_0 \in \mathbb{R}$. *Úloha najít řešení DR, které splňuje tzv.* ***počáteční podmínku***

$$y(x_0) = y_0,$$

se nazývá počáteční úloha. Jejím řešením je funkce, která splňuje počáteční podmínku a je na nějakém otevřeném intervalu, obsahujícím bod x_0, *řešením DR.*

4.1 Metody řešení diferenciálních rovnic

Diferenciální rovnice lze řešit:

- analyticky,
- numericky.

V případě analytického řešení provádíme integraci podle pravé strany rovnice. Analytické řešení je ovšem často příliš výpočetně náročné nebo není možné.

Pomocí numerického řešení získáváme přibližné řešení. Metod pro numerické řešení diferenciálních rovnic je několik.

Metody lze rozdělit na

- jednokrokové,
- vícekrokové.

Jednokrokové metody využívají pro výpočet informace pouze z jediného předchozího kroku. Nejjednoduší metodou tohoto typu je Eulerova metoda (jedná se o numerickou metodu prvního řádu). Další metoda je Runge-Kutta řádu 4. Její výpočet je sice náročný, ale je vyvážen vyšší přesností díky vyššímu řádu metody.

U vícekrokových metod využíváme pro výpočet informace z několika předchozích kroků. Mezi vícekrokové metody patří například Adams-Bashforthovy metody, Adams-Moultonovy metody a metoda prediktor-korektor.

4.2 Převod SLAR na SDR

Soustavu lineárních algebraických rovnic (SLAR) lze řešit pomocí převodu na soustavu diferenciálních rovnic (SDR). Podrobnější informace nalezneme v [12]. Důvod, proč je výhodné řešit SLAR pomoc SDR je ten, že nalezené řešení je potom vždy stabilní. Stabilitu můžeme zjistit pomocí vlastních čísel. Abychom získali vlastní čísla, je potřeba sestavit charakteristickou rovnici (4.1). Tu získáme tak, že determinant soustavy **A** vynásobíme s jednotkovou maticí **E** s konstantou λ, položíme roven nule a následně vyřešíme. Kořeny charakteristické rovnice nazýváme *vlastní čísla*.

$$|A - \lambda \cdot E| = 0 \tag{4.1}$$

Soustavu diferenciálních rovnic lze zapsat v exponenicálním tvaru $e^{\lambda t}$. Tento výraz lze rozepsat jako $e^{(Re(\lambda)+iIm(\lambda))t} = C + e^{Re(\lambda)t} \cdot e^{Im(\lambda)t}$. Absolutní hodnota členu $|e^{Im(\lambda)t}| = 1$, tento člen odpovídá popisu kružnice, nebude nás tedy dále zajímat. Zaměříme se na reálnou část výrazu, tedy na $e^{Re(\lambda)t}$. Pokud za proměnnou λ dosadíme číslo, které je větší než nula, potom dostaneme rostoucí exponenciální funkci. Na začátku jsme uvedli, že řešení soustavy diferenciálních rovnic lze vyjádřit právě pomocí této exponenciální funkce. V tomto případě ovšem bude řešení nestabilní, protože funkce je rostoucí.

Aby byla soustava diferenciálních rovnic stabilní, musí pro kořeny charakteristické rovnice platí podmínka (4.2), potom je zaručeno, že dostaneme řešení SLAR v ustáleném stavu.

$$Re(\lambda) < 0 \tag{4.2}$$

S přihlédnutím k podmínce (4.2) má potom soustava diferenciálních rovnic tvar $e^{-\lambda t} = \frac{1}{e^{\lambda t}}$. V tomto případě je exponenciální funkce klesající, tedy řešení soustavy diferenciálních rovnic je stabilní. Z výše uvedeného plyne, že je potřeba tedy zajistit, aby matice soustavy byla negativně definitní.

V příkladu 3.4 jsme mohli vidět situaci, kdy oba kořeny charakteristické rovnice byly kladné. Nyní si ukážeme příklad 4.1, kde tomu tak není.

Příklad 4.1.

$$A = \begin{pmatrix} 1 & 2 \\ 0 & -4 \end{pmatrix}$$

Matici upravíme podle výše zmíněného vztahu 4.1.

$$\begin{vmatrix} 1-\lambda & 2 \\ 0 & -4-\lambda \end{vmatrix} = 0$$

Vyčíslíme determinant.

$$(1-\lambda)\cdot(-4-\lambda) - 2\cdot 0$$

Dostaneme charakteristickou rovnici.

$$\lambda^2 + 3\lambda - 4 = 0$$

Pomocí vzorce pro výpočet kvadratické rovnice získáme kořeny charakteristické rovnice.

$$\lambda_{1,2} = \frac{-3 \pm \sqrt{9 - 4\cdot 1\cdot(-4)}}{2}$$

$$\lambda_1 = -4,\ \lambda_2 = 1$$

□

Kořen charakteristické rovnice λ_2 je kladný. Není tedy splněna podmínka (4.2) a daná soustava rovnic nebude stabilní.

Nyní si ukážeme, jak zaručit, aby byla vždy splněna podmínka (4.2), a tedy aby byla libovolná soustava vždy stabilní. Možné řešení nabízí využití transponované matice. Pro jistotu připomeneme definici transponované matice.

Definice 4.3. *Nechť $A = (a_{ij})$ je matice typu $m \times n$. Pak matice $A^T = (a_{ij}^T)$ typu $n \times m$, kde*

$$a_{ji}^T = a_{ij}$$

pro $i \in \{1, 2, \ldots, m\}$ a $i \in \{1, 2, \ldots, n\}$ se nazývá transponovaná matice *k matici A. Transponovanou matici získáme z původní matice záměnou řádků za sloupce.*

Algoritmus pro převod SLAR na SDR se nazývá *transformační algoritmus* a sestává z následujících kroků.

Algoritmus 4.1.

1. *Pro SLAR vytvoříme matici soustavy A a vektor pravých stran $\vec{b}$.*
2. *Matici soustavy A vynásobíme zleva zápornou transponovanou maticí A^T.*
3. *Podle vztahu 4.3 získáme SDR.*

$$-A^T \cdot A \cdot x + A^T \cdot \vec{b} = x' \tag{4.3}$$

Předpokládáme, že matice A je regulární. Matice $A \cdot A^T$ je za každých okolností pozitivně definitní. Také platí následující charakteristická rovnice (4.4).

$$|-A^T \cdot A - \lambda \cdot E| = 0 \tag{4.4}$$

Kořeny této charakteristické rovnice jsou vždy záporné a dostáváme tedy vždy stabilní řešení soustavy. Poznamenejme, že transformací (4.4) dochází ke zvýšení tuhosti systému. O tuhých systémech pojednává podkapitola 4.4. Nyní zkusíme pomocí algoritmu 4.1 upravit příklad 4.1 tak, abychom získali stabilní řešení.

Příklad 4.2. *Matice soustavy je*

$$A = \begin{pmatrix} 1 & 2 \\ 0 & -4 \end{pmatrix}.$$

Sestavíme charakteristickou rovnici k původní matici A.

$$A = \begin{vmatrix} 1-\lambda & 2 \\ 0 & -4-\lambda \end{vmatrix} = 0$$

Po vyčíslení determinantu $\det A$ *dostáváme následující kvadratickou rovnici.*

$$\lambda^2 + 3\lambda - 4 = 0$$

Kořeny charakteristické rovnice jsou

$$\lambda_1 = 1, \quad \lambda_2 = -0,25\,.$$

Sestavíme zápornou transponovanou matici k matici A.

$$A^T = \begin{pmatrix} -1 & 0 \\ -2 & 4 \end{pmatrix}$$

Součin matic $-A^T \cdot A$ *je uveden níže.*

$$\begin{pmatrix} 1 & 2 \\ 0 & -4 \end{pmatrix} \cdot \begin{pmatrix} -1 & 0 \\ -2 & 4 \end{pmatrix} = \begin{pmatrix} -5 & 8 \\ 8 & -16 \end{pmatrix}$$

Sestavíme charakteristickou rovnici.

$$\begin{vmatrix} -5-\lambda & 8 \\ 8 & -16-\lambda \end{vmatrix} = 0$$

Po vyčíslení determinantu $\det A$ *dostáváme následující kvadratickou rovnici.*

$$\lambda^2 + 21\lambda + 16 = 0$$

Kořeny charakteristické rovnice jsou

$$\lambda_1 = -20,2083, \;\; \lambda_2 = -0,7918\,.$$

Oba kořeny jsou záporné, je tedy zaručena stabilita řešení soustavy.

□

4.3 Zápis SLAR pomocí SDR v TKSL/C

Program TKSL/C vyžaduje pro svou funkčnost speciální zápis SDR. Nezapomeňme také, že pro každou DR je potřeba uvést počáteční podmínku. Pokud ji neuvedeme, program TKSL/C ji nastaví na nulu. Ukážeme si postupně dva možné postupy, kterými můžeme převést SLAR na SDR. Zároveň je uvedenými postupy zajištěna stabilita řešení. Více informací uvádí literatura [12, 9].

4.3.1 Transformační algoritmus

Tento postup použijeme v případě, že některý z kořenů charakteristické rovnice není záporný. Výpočet je založen na využití transponované matice, podobně jako jsme mohli vidět výše v algoritmu 4.1. Názvy proměnných si samozřejmě můžeme zvolit i jiné než ty, které používáme níže. Mějme SLAR v obecném tvaru, viz (4.5).

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n &= b_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n &= b_2 \\ &\vdots \\ a_{n1}x_1 + a_{n2}x_2 + \cdots + a_{nn}x_n &= b_n \end{aligned} \tag{4.5}$$

Nyní pravé strany všech rovnic převedeme na levou stranu. Na pravou stranu dosadíme parametry q_i, kde $i \in 1, \ldots, n$. Počet rovnic označuje písmeno n.

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n - b_1 &= q_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n - b_2 &= q_2 \\ &\vdots \\ a_{n1}x_1 + a_{n2}x_2 + \cdots + a_{nn}x_n - b_n &= q_n \end{aligned} \tag{4.6}$$

Zápisem (4.6) získáme první část zdrojového kódu. Druhou část získáme tak, že z matice soustavy vytvoříme transponovanou matici a nahradíme neznámé parametry následujícím způsobem.

$$x1 = q_1, x_2 = q_2, \ldots, x_n = q_n \tag{4.7}$$

Potom na pravé strany rovnic dosadíme záporné proměnné v diferenciálním tvaru a dostáváme již kompletní podobu zápisu druhé části zdrojového kódu (4.8).

$$\begin{aligned} a_{11}q_1 + a_{12}q_2 + \cdots + a_{1n}q_n &= -x_1' \\ a_{21}q_1 + a_{22}q_2 + \cdots + a_{2n}q_n &= -x_2' \\ &\vdots \\ a_{n1}q_1 + a_{n2}q_2 + \cdots + a_{nn}q_n &= -x_3' \end{aligned} \tag{4.8}$$

Konečný tvar je uveden ve vztahu (4.9). Rovnice jsme převedli z jedné strany na druhou a vynásobili číslem -1.

$$\begin{aligned} x_1' &= -(a_{11}q_1 + a_{12}q_2 + \cdots + a_{1n}q_n) \\ x_2' &= -(a_{21}q_1 + a_{22}q_2 + \cdots + a_{2n}q_n) \\ &\vdots \\ x_n' &= -(a_{n1}q_1 + a_{n2}q_2 + \cdots + a_{nn}q_n)) \end{aligned} \tag{4.9}$$

Pokud bychom si chtěli nastavit počáteční podmínky na jinou hodnotu, než na nulu, využijeme zápis (4.10), ve kterém se za znakem `&` nachází počáteční podmínky pp_n.

$$\begin{aligned} x_1' &= -(a_{11}q_1 + a_{12}q_2 + \cdots + a_{1n}q_n)\ \&pp_1 \\ x_2' &= -(a_{21}q_1 + a_{22}q_2 + \cdots + a_{2n}q_n)\ \&pp_2 \\ &\vdots \\ x_n' &= -(a_{n1}q_1 + a_{n2}q_2 + \cdots + a_{nn}q_n))\ \&pp_n \end{aligned} \tag{4.10}$$

Ukážeme si konkrétní příklad. Ve zdrojových kódech pro TKSL/C se jednotlivé rovnice oddělují středníkem.

Příklad 4.3. *Mějme SLAR v následujícím tvaru.*

$$\begin{aligned} 35y_1 + 17y_2 - 11y_3 &= 36 \\ 17y_1 + 11y_2 - 9y_3 &= 12 \\ -11y_1 - 9y_2 + 9y_3 &= -2 \end{aligned}$$

Vytvoříme první část zdrojového kódu dle vztahu (4.6).

$$\begin{aligned} 35y_1 + 17y_2 - 11y_3 - 36 &= q_1; \\ 17y_1 + 11y_2 - 9y_3 - 12 &= q_2; \\ -11y_1 - 9y_2 + 9y_3 + 2 &= q_3; \end{aligned}$$

Druhou část zdrojového kódu vytvoříme podle vztahu (4.8).

$$\begin{aligned} y1' &= -(35q_1 + 17q_2 - 11q_3); \\ y2' &= -(17q_1 + 11q_2 - 9q_3); \\ y3' &= -(-11q_1 - 9q_2 + 9q_3); \end{aligned}$$

□

4.3.2 Elementární převod

Druhou metodou převodu SLAR na SDR je elementární převod. Tento postup využijeme, pokud zjistíme, že kořeny charakteristické rovnice jsou záporné, a tedy není již potřeba provádět úpravy pomocí transponované matice, protože je již splněna podmínka stability řešení 4.2. Pomocí této metody můžeme tedy SLAR přepsat rovnou na SDR. U některých soustav může elementární převod také sloužit k tomu, aby bylo nalezeno stabilní řešení, podobně jako tomu bylo u transformačního algoritmu. Mějme SLAR v obecném tvaru.

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n &= b_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n &= b_2 \\ &\vdots \\ a_{n1}x_1 + a_{n2}x_2 + \cdots + a_{nn}x_n &= b_n \end{aligned} \tag{4.11}$$

Nyní pravé strany všech rovnic převedeme na levou stranu. Na pravé straně ponecháme nulu.

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n - b_1 &= 0 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n - b_2 &= 0 \\ &\vdots \\ a_{n1}x_1 + a_{n2}x_2 + \cdots + a_{nn}x_n - b_n &= 0 \end{aligned} \tag{4.12}$$

Na pravou stranu umístíme záporné proměnné v diferenciálním tvaru.

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n - b_1 &= -x_1' \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n - b_2 &= -x_2' \\ &\vdots \\ a_{n1}x_1 + a_{n2}x_2 + \cdots + a_{nn}x_n - b_n &= -x_3' \end{aligned} \tag{4.13}$$

Nakonec rovnice převedeme z jedné strany na druhou a vynásobíme číslem -1.

$$\begin{aligned} x_1' &= -(a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n) \\ x_2' &= -(a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n) \\ &\vdots \\ x_n' &= -(a_{n1}x_1 + a_{n2}x_2 + \cdots + a_{nn}x_n) \end{aligned} \tag{4.14}$$

Ukážeme si nyní, jak pomocí elementárního převodu získat z nestabilní SDR stabilní SDR. Danou SDR lze modifikovat pomocí úprav uvedených níže.

- Některou z rovnic nebo všechny rovnice násobíme číslem -1,
- zaměníme pořadí diferenciálních proměnných.

Pro každou upravenou soustavu rovnic potom musíme spočítat její determinant a zjistit, jestli je větší než nula. Pokud ano, pak jsme nalezli stabilní řešení. Pro lepší pochopení uvedeme příklady A.7 a A.8, které nalezneme v příloze A. Vidíme, že počet zápisů soustavy rovnic se s narůstajícím počtem rovnic značně zvyšuje. Proto nyní zkusíme odvodit obecný vztah, který bude určovat, kolika způsoby lze danou soustavu lineárních rovnic o n neznámých zapsat.

Rozdělíme si úkol na 2 části:

1. počet způsobů, jak můžeme měnit znaménka rovnic soustavy,
2. počet způsobů, jak můžeme zaměnit diferenciální proměnné.

Počet způsobů záměn znamének rovnic soustavy

U soustavy dvou lineárních rovnic jsme viděli, že znaménka lze zaměnit 4 způsoby, u soustavy tří lineárních rovnic lze totéž učinit 8 způsoby. Bude tedy zřejmě platit vztah 2^n, kde n je počet rovnic dané soustavy.

Počet způsobů záměn diferenciálních proměnných

Nyní se podíváme, kolika způsoby můžeme zaměnit pořadí diferenciálních proměnných na pravých stranách rovnice. Potřebujeme vybrat do skupiny právě n prvků a různě zaměnit jejich pořadí. Zvolíme proto pro výpočet permutaci.

Definice 4.4. *Permutace z n prvků je uspořádaná ntice z těchto prvků. Znamená to, že do skupiny vybíráme všech n prvků a jejich pořadí zaměňujeme. Počet všech permutací z n prvků značíme P(n). Počet permutací určíme jako P(n)=n!.*

Celkový vztah je uveden níže.

$$2^n.P(n) \tag{4.15}$$

Na závěr uvedeme tabulku 4.1, která udává, kolika způsoby lze SDR zapsat pomocí elementárního převodu.

Tabulka 4.1: Počty možných způsobů zápisů SDR pomocí elementárního převodu

matice (řádky x sloupce)	Dosazení do vzorce	Výsledek
2x2	$2^n.P(2)$	8
3x3	$3^n.P(3)$	162
4x4	$4^n.P(4)$	6144
5x5	$5^n.P(5)$	375000
6x6	$6^n.P(6)$	4150656720
...	...	...
10x10	$10^n.P(10)$	3715891200

4.4 Tuhé systémy

Tuhé systémy se také nazývají *soustavy se silným tlumením* nebo anglicky *stiff systems*. Pokud jsou vlastní čísla soustavy řádově rozdílná, potom dochází k situaci, kdy do celkového řešení na určitém intervalu některé z vlastních čísel přispívá svou složkou zcela minimálně. Potom je možné tuto složku zanedbat. Avšak toto vlastní číslo nutí k tomu, abychom zvolili integrační krok h malý. Tato volba zapříčiní, že výpočet je neefektivní a řešení konverguje velmi pomalu. Více je problematika rozvedena v [15, 1, 6].

K detekci tuhých systémů můžeme využít vlastí čísla soustavy. Pokud platí vztah

$$|Re\lambda_{max}| \gg |Re\lambda_{min}| \tag{4.16}$$

potom SLAR je tuhým systémem. Čím menší je $Re\lambda_{min}$, tím řešení konverguje pomaleji a tím delší časový interval k vyřešení soustavy potřebujeme. Uvedeme vzorec (4.17), pomocí něhož můžeme zjistit *koeficinent tuhosti* s. Někdy bývá také nazýván jako *stiff koeficient*.

$$s = \frac{|Re(\lambda_{max})|}{|Re(\lambda_{min})|} \tag{4.17}$$

Platí, že čím větší je tento koeficient, tím je systém více tuhý. Vrátíme se k příkladu 4.2 a vypočítáme koeficienty tuhosti.

Příklad 4.4. *Výpočtěte koeficienty tuhosti pro příklad 4.2. Pro výpočet použijeme vztah (4.17). Nejprve vypočítáme koeficient tuhosti pro původní matici soustavy A.*

$$s = \frac{1}{0,25} = 4$$

Nyní vypočítáme koeficient tuhosti pro transponovanou matici soustavy.

$$s = \frac{20,2083}{0,7918} = 25,5220$$

□

Koeficient tuhosti původní matice soustavy ($s = 4$) je menší než koeficient tuhosti transponované matice ($s = 25,5220$). Zde je tedy ukázáno, že násobením původní soustavy rovnic transponovanou maticí se zvyšuje tuhost systému.

Kapitola 5

Dosažené výsledky

5.1 Srovnání programů TKSL/C, Matlab a Maple

V této podkapitole se zaměříme, jak lze řešit SLAR pomocí programů TKSL/C, Matlab a Maple. Informace byly čerpány z [9, 8, 7].

5.1.1 Program TKSL/C

Program TKSL/C vznikl na Fakultě informačních technologií v Brně. Je napsán v jazyce C++ a k řešení využívá metodu Taylorova typu. Program TKSL/C řeší SLAR převodem na SDR. Program se ovládá z příkazové řádky. Chování programu lze ovlivnit řadou parametrů uvedených v tabulce 5.1.

Tabulka 5.1: Parametry programu TKSL/C

Parametr	Význam parametru
-h	vytiskne nápovědu
-V	verze programu
-s	velikost kroku (implicitně 0,1)
-t	horní mez výpočtu (implicitně 1)
-W	šířka čísel v bitech (implicitně 80)
-O	maximální řád (implicitně 50)
-A	přesnost (implicitně 1e-15)
-Z	počet nulových hodnot pro detekci konce kroku (implicitně 4)
-w	formát výstupu – šířka čísel (implicitně 16)
-p	formát výstupu – počet desetinných čísel (implicitně 10)
-f	formát výstupu – definice proměnných, pro které zobrazujeme výpočty
-x	formát výstupu – XML formát
-n	text pro nedefinovaou hodnotu (implicitně "n/a")
-c	kontroluje špatnou konvergenci (implicitně 0)
-i	zvýšení kroku o zadanou hodnotu
-I	horní hranice zvyšování kroku (implicitně 0,5)

Soustavu diferenciálních rovnice lze do programu TKSL/C zadat buď z textového souboru nebo manuálně přes příkazovou řádku.

Příklad 5.1. *Mějme soustavu lineárních algebraických rovnic.*

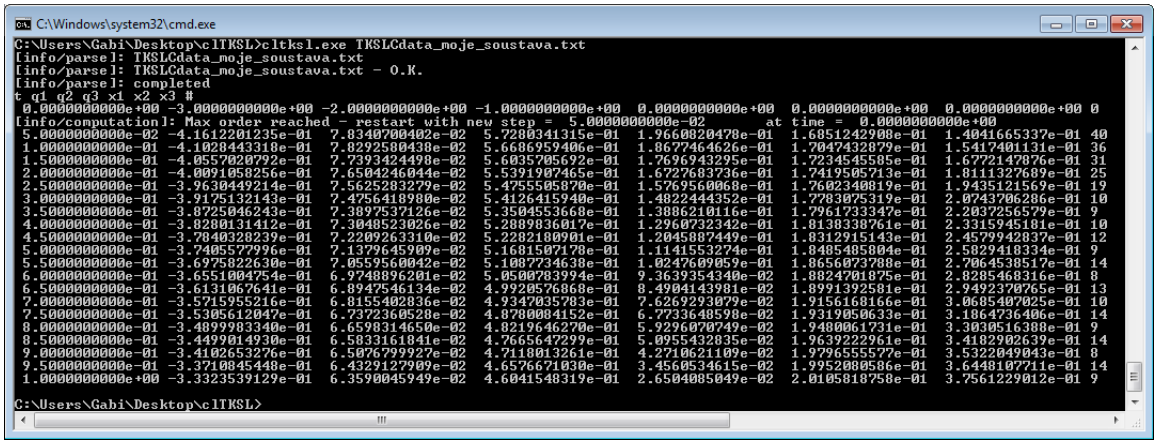
$$\begin{aligned} 6x_1 + 5x_2 + 4x_3 &= 3 \\ 5x_1 + 4x_2 + 3x_3 &= 2 \\ 4x_1 + 3x_2 + 2x_3 &= 1 \end{aligned}$$

Pomocí vztahů pro transformační algoritmus (4.6) a (4.8) ji převedeme na soustavu diferenciálních rovnic. Dostáváme zápis, který je již připraven pro program TKSL/C. Počáteční podmínky jsou nulové, proto by zde nemusely být uvedeny. Pro přehlednost je ale raději uvedeme.

$$\begin{aligned} q1 &= +6 * x1 + 5 * x2 + 4 * x3 - 3; \\ q2 &= +5 * x1 + 4 * x2 + 3 * x3 - 2; \\ q3 &= +4 * x1 + 3 * x2 + 2 * x3 - 1; \\ x1' &= -(+6 * q1 + 5 * q2 + 4 * q3)\ \&0; \\ x2' &= -(+5 * q1 + 4 * q2 + 3 * q3)\ \&0; \\ x3' &= -(+4 * q1 + 3 * q2 + 2 * q3)\ \&0; \end{aligned}$$

□

Výstup programu TKSL/C je znázorněn na obrázku 5.1. Jednotlivé sloupce odpovídají po řadě proměnným: `t,` $q_1, \ldots, q_n$`,` $x1, \ldots, x_n$`,` `#`.



Obrázek 5.1: Program TKSL/C – ukázka výstupu programu

5.1.2 Program Matlab

Program Matlab je integrované prostředí pro vědeckotechnické výpočty, modelování, návrhy algoritmů, simulace, atp. V programu Matlab lze k řešení soustav lineárních algebraických rovnic využít funkce `linsolve` a `mldivide`. Postupně si vysvětlíme, jak se obě tyto funkce používají.

Funkce `linsolve`

Funkce `linsolve` má dva povinné parametry a jeden nepovinný. Mezi povinné parametry patří matice soustavy a vektor pravých stran. Třetím a nepovinným parametrem je potom

název metody, pomocí které si přejeme soustavu řešit. Můžeme vybítaz z metod, které jsou uvedeny v tabulce 5.2.

Tabulka 5.2: Metody řešení SLAR pro funkci `linsolve`

Název parametru	Význam parametru
LT	Dolní trojúhelníková matice
UT	Horní trojúhelníková matice
UHESS	Horní Hessenbergova matice
SYM	Reálná symetrická nebo komplexní Hermitovská matice
POSDEFF	Pozitivně definitní matice
RECT	Obecná čtvercová matice
TRANSA	Transponovaná matice

Předpokládejme, že v proměnné A máme matici soustavy a v proměnné x máme vektor pravých stran. Pokud chceme například nastavit, aby se soustava rovnic řešila pomocí horní trojúhelníkové matice, musíme ve struktuře `opts` nastavit položku `LT` na hodnotu true. Poznamenejme, že do veškerých položek struktury `opts` lze ukládat pouze logické hodnoty, tedy true nebo false. Defaultně jsou všechny položky struktury nastaveny na hodnotu false. Implicitní metoda řešení soustavy rovnic je LU rozklad.

Následující tabulka ukazujem jak lze kombinovat jednotlivé metody mezi sebou. Hodnota true je označena jako T, hodnota false jako F.

Tabulka 5.3: Možné kombinace parametrů udávající typ metody pro funkci `linsolve`

LT	UT	UHESS	SYM	POSTDEFF	RECT	TRANSA
T	F	F	F	F	T/F	T/F
F	T	F	F	F	T/F	T/F
F	F	T	F	F	F	T/F
F	F	F	T	T/F	F	T/F
F	F	F	F	F	T/F	T/F

Nyní uvedeme příklad 5.2, ze kterého budeme i nadále vycházet.

Příklad 5.2. *Příklad použití funkce* `linsolve`*. Definujeme matici A, vektor pravých stran b a zvolíme metodu řešení například pomocí horní trojúhelníkové matice.*

```
A = [1 2 3; 2 3 5; 5 6 2];
b = [4; 7; 1];
opts.UT=true;
linsolve(A,b,opts)
```

□

Funkce `linsolve` má nevýhodu v tom, že nás nijak neupozorní, pokud zvolíme metodu nevhodnou k řešení dané soustavy rovnic. Funkce vrátí výsledky, které ovšem nemusí být správné.

Funkce `mldivide`

Funkce `mldivide` má dva povinné parametry – matici soustavy a vektor pravých stran. Na rozdíl od funkce `linsolve` si nemůžeme zvolit metodu, kterou se bude soustava rovnic řešit. Funkce `mldivide` totiž provede sadu různých testů a na jejich základě vyhodnotí, která metoda je pro danou soustavu rovnic nejlepší. Výhodou je to, že nemůžeme zvolit metodu, která by nebyla pro řešení soustavy rovnic vyhovující. Zápis `mldivide (A,b)` představuje ekvivalent k `A\b` (levé maticové dělení). Zápis `mrdivide (b,A)` představuje ekvivalent k `b/A` (pravé maticové dělení).

Příklad 5.3. *Příklad použití funkce* `mldivide`*. Definujeme matici A, vektor pravých stran b. Uvažujme matici soustavy a vektor pravých stran jako v příkladu 5.2.*

```
A = [1 2 3; 2 3 5; 5 6 2];
b = [4; 7; 1];
mldivide(A,b)
```

□

Funkce pro řešení diferenciálních rovnic

Nyní se podíváme, jak lze v systému Matlab řešit soustavy diferenciálních rovnic. K tomuto účelu je k dispozici řada funkcí, jak můžeme vidět níže:

- `ode23` – pro non-stiff rovnice (Runge-Kuttova metoda, řád 2–3),
- `ode45` – pro non-stiff rovnice (Runge-Kuttova metoda, řád 4–5),
- `ode113` – pro non-stiff rovnice (Runge-Kuttova metoda, řád 2–3),
- `ode113` – pro non-stiff rovnice (Adams-Bashforth-Moultonova metoda),
- `ode23t` – pro průměrně stiff rovnice (lichoběžníkové pravidlo),
- `ode15s` – pro stiff rovnice (Gearova metoda),
- `ode23s` – pro stiff rovnice (modifikovaná Rosenbrockova metoda),
- `ode23b` – pro stiff rovnice (implicitní Runge-Kuttova metoda kombinovaná se zpětnou diferenční formulí druhého řádu).

Syntaxi všech výše zmíněných funkcí lze zapsat ve tvaru `[T,Y] = ode_funkce(funkce, interval_integrace, vektor_PP)`. Proměnná `T` označuje sloupcový vektor časových údajů, proměnná `Y` obsahuje pole hodnot řešení. Každý řádek koresponduje s časovým údajem uloženým v proměnné `T`. Nyní se dostáváme k parametrům funkce. Prvním parametrem je ukazatel na funkci, která vypočítá pravou stranu diferenciální rovnice. Druhý parametr určuje interval integrace. Třetím parametrem je vektor počátečních podmínek.

Příklad 5.4. *Příklad použití funkce* `ode45` *a také definice funkce* `myfun`*. Opět uvažujeme matici soustavy a vektor pravých stran jako v příkladech 5.2.*

□

```
global At bt;
A = [1 2 3; 2 3 5; 5 6 2];
b = [4; 7; 1];
At = -A' * A;
bt = -A' * b;
[t y] = ode45(@myfun,[0 100],zeros(50, 1));
```

5.1.3 Program Maple

Program Maple je systém počítačové algebry pro výuku a využití matematiky v přírodovědných, technických a ekonomických oborech. Byl vyvinut firmou Waterloo Maple Software.

Funkce pro řešení lineárních algebraických rovnic

Nejdříve si ukážeme, jak se programu Matlab řeší soustavy lineárních algebraických rovnic. K tomuto účelu můžeme využít funkci `solve`.

Obecný zápis funkce má tvar `solve(soustava_rovnic, promenne)`. Funkce má 2 parametry, prvním je soustava rovnic, kterou si přejeme vyřešit, druhým parametrem je proměnná (nebo proměnné), které chceme z rovnice vypočítat. Pokud nezadáme druhý parametr, potom se rovnice vyřeší pro všechny proměnné, které obsahuje. Funkce `solve` má různé varianty (`msolve, dsolve, isolve, fsolve`).

Příklad 5.5. *Příklad použití funkce* `solve`*. Zadání opět vychází z příkladu 5.2.*

```
soustava:={x+2*y+3*z=4, 2*x+3*y+5*z=7, 5*x+6*y+2*z=1};
solve(soustava);
```

□

Funkce pro řešení diferenciálních rovnic

Pro řešení soustav diferenciálních rovnic je v programu Maple k dispozici funkce `dsolve`. Povinným parametrem je diferenciální rovnice. Pro vyjádření derivace se používá příkaz `diff`. Počáteční podmínky jsou uvedeny ve složených množinových závorkách spolu s rovnicí. Pokud si přejeme řešit diferenciální rovnice numericky, použijeme volbu `type=numeric`. Pomocí volby `method` si lze zvolit metodu, pomocí které se bude výpočet provádět. Metody můžeme volit následující:

- rkf45 – metoda Runge-Kuttova typu (Fehlberg) 4–5 řádu,
- dverk78 – metoda Runge-Kuttova typu 7–8 řádu,
- classical – Eulerova metoda,
- gear – Gearova jednokroková extrapolační metoda,
- lsode – metody pro stiff rovnice, Adamsova metoda, zpětná diferenční formule,

- taylorseries – metody Taylorových rozvojů.

Pokud neuvedeme volbu `method`, potom se implicitně použije metoda `rkf45`. Poznamenejme ještě, jaké parametry má příkaz `diff`. Prvním parametrem je derivovaná funkce a druhým parametrem je proměnná podle které budeme derivovat. V případě derivací funkce jedné proměnné to znamená, že kolikrát danou proměnnou uvedeme, tolikátou derivaci bude systém Maple počítat. Pokud chceme vypočítat například čtvrtou derivaci funkce f, bude mít příkaz `diff` tvar `diff(f,x,x,x,x)`. Můžeme také použít zkrácenou formu zápisu `diff(f,x$4)`.

Existuje také podobný příkaz `Diff`. Ten se liší od příkazu `diff` tím, že neprovede přímo symbolickou derivaci, ale pouze ho rozepíše do řádné matematické notace.

Příklad 5.6. *Příklad použití funkce* `dsolve`. *Řešte rovnici* $y' = y + y^2$, *počáteční podmínka je* $y(0) = 1$. *K řešení využijeme metodu Runge-Kutta 4–5 řádu, tato metoda je implicitní, proto ji nemusíme uvádět.*

`diff_rovnice` := ({`diff(y(x),x) = y(x) + y(x) * y(x), y(0) = 1`}, `y(x), type = numeric`);
Program vrátí výsledek ve tvaru uvedeném níže.
`diff_rovnice` := `proc(rkf45_x) ... endproc`
Pro získání hodnoty řešení například v bodě $0,5$, musíme napsat příkaz `diff_rovnice(0.5)`;.

□

5.2 Srovnání časové náročnosti řešení SLAR a SDR

Pro srovnání časové náročnosti řešení SLAR a SDR byly provedeny následující experimenty:

- výpočet SLAR programem Matlab,
- výpočet SDR programem Matlab,
- výpočet SDR programem TKSL/C.

Nyní přiblížíme postup jednotlivých experimentů a vyhodnotíme získané výsledky. V rámci všech experimentů byl výpočet s daným řádem matice soustavy proveden vždy dvacetkrát. Z těchto hodnot byl poté vypočítán aritmetický průměr. Experimenty byly prováděny pod systémem Linux, na školním serveru `merlin.fit.vutbr.cz`.

5.2.1 Program pro generování koeficientů soustavy rovnic

Pro snadnější průběh tohoto experimentu byl napsán pomocný program v systému Matlab. Program `gensystem.m` sloužil k vygenerování matice soustavy o daném rozměru. Koeficienty matice soustavy byly generovány náhodně, v rozsahu $1 - 10$. Program také uložil vygenerovanou matici soustavy do datového souboru Matlabu.

Dále byla vytvořena funkce `saveeq.m`, která převedla vygenerované koeficienty matice soustavy na textový soubor se soustavou rovnic podle obecného tvaru, viz vztah (2.1), soubor byl poté uložen. Tato funkce byla volána z programu `gensystem.m`.

5.2.2 Výpočet SLAR programem Matlab

Program pro výpočet SLAR byla použita funkce `mldivide` (viz podkapitola 5.1.2). Výsledky testů jsou uvedeny v tabulce 5.4.

Tabulka 5.4: Průměrný čas řešení SLAR funkcí `mldivide` v programu Matlab

rozměr matice $n \times n$	Průměrný čas řešení
50×50	$4,068500 \cdot 10^{-4}$
100×100	$7,294000 \cdot 10^{-4}$
150×150	$1,239050 \cdot 10^{-3}$
200×200	$3,257750 \cdot 10^{-3}$
250×250	$3,564200 \cdot 10^{-3}$
300×300	$5,771050 \cdot 10^{-3}$
350×350	$1,111190 \cdot 10^{-2}$
400×400	$1,642725 \cdot 10^{-2}$
450×450	$2,265670 \cdot 10^{-2}$

5.2.3 Výpočet SDR programem TKSL/C

K tomuto účelu byl vytvořen skript v jazyce Python, který volal program TKSL/C. Parametry TKSL/C byly nastaveny takto: `-t 1.0 -A 1e-15`. Význam parametrů byl již uveden v tabulce 5.1. Výsledky experimentů ukazuje tabulka 5.5.

Tabulka 5.5: Průměrný čas řešení SDR v programu TKSL/C

rozměr matice $n \times n$	Průměrný čas řešení
50×50	$0,0940515518188$
100×100	$0,356560730934$
150×150	$0,795030450821$
200×200	$1,58277004957$
250×250	$2,3172401049$
300×300	$3,19485945702$
350×350	$4,31315324306$
400×400	$5,38396792412$
450×450	$6,83516882658$

5.2.4 Výpočet SDR programem Matlab

Cílem experimentu bylo vyzkoušet, jak rychle dokáže program Matlab vyřešit SLAR převodem na SDR. Opět byl vytvořen speciální program, který celý experiment automatizoval. Pro řešení SDR byla použita nejprve funkce `ode45`, viz sekce 5.1.2. Výsledky testů znázorňuje tabulka 5.6.

Z tabulky 5.6 si můžeme všimnout, že výpočet je pomalý. To může být zapříčiněno tím, že mohla být vygenerována soustava, která je tuhá. Funkce `ode45` (viz podkapitola 5.1.2 je určena k řešení non-stiff rovnic, tedy rovnic, které nejsou tuhými systémy.

Tabulka 5.6: Průměrný čas řešení SDR funkcí `ode45` v programu Matlab

rozměr matice $n \times n$	Průměrný čas řešení
2×2	$1,808855$
4×4	$2,642267$
6×6	$9,162519$
8×8	$16,84653$
10×10	$41,37285$

Proto byla provedena další sada experimentů, tentokrát s využitím funkce `ode15s`, která se s tuhými systémy umí vypořádat. Funkce `ode15s` patří narozdíl od funkce `ode45` mezi vícekrokové metody a je velmi efektivní. Výsledky můžeme vidět níže, v tabulce 5.7.

Tabulka 5.7: Průměrný čas řešení SDR funkcí `ode15s` v programu Matlab

rozměr matice $n \times n$	Průměrný čas řešení
2×2	$1,714856 \cdot 10^{-2}$
4×4	$2,518945 \cdot 10^{-2}$
6×6	$3,695310 \cdot 10^{-2}$
8×8	$3,992515 \cdot 10^{-2}$
10×10	$4,047385 \cdot 10^{-2}$
...	...
50×50	$6,025805 \cdot 10^{-2}$
100×100	$9,699655 \cdot 10^{-2}$
150×150	$2,257759 \cdot 10^{-1}$
200×200	$2,616328 \cdot 10^{-1}$
250×250	$4,716819 \cdot 10^{-1}$
300×300	$6,140634 \cdot 10^{-1}$
350×350	$8,533643 \cdot 10^{-1}$
400×400	$1,196343$
450×450	$1,621755$

5.3 Shrnutí získaných výsledků

V podkapitole 5.2 jsme mohli vidět srovnání programů TKSL/C a Matlab. Z výše uvedených experimentů vyplývá, že SLAR byly nejrychleji vyřešeny v programu Matlab při použití metod pro výpočet SLAR.

Pokud srovnáme řešení SDR pomocí programu Matlab a TKSL/C, můžeme konstatovat, že v tomto ohledu je systém Matlab rychlejší než program TKSL/C, ovšem za předpokladu, že k řešení SDR použijeme metody programu Matlab takové, které jsou vhodné pro počítání i s tuhými systémy.

Kapitola 6

Uživatelské rozhraní pro TKSL/C

Program TKSL/C lze ovládat pouze přes příkazovou řádku, proto bylo v rámci bakalářské práce navrženo grafické uživatelské rozhraní, které by práci s programem TKSL/C usnadnilo. Aplikace byla vytvořena v jazyce C# ve vývojovém prostředí Microsoft Visual Studio 10.0. Pro správnou funkčnost aplikace je potřeba mít nainstalovaný .NET framework redistributable[1]. Uživatel zadá do programu vstupní data (buď manuálně anebo ze souboru), program data transformuje na tvar, který přijímá TKSL/C, viz vztahy (4.6) a (4.8). Obrázek B.1 ukazuje, jak uživatelské rozhraní vypadá.

6.1 Způsob zadávání vstupních dat

Program umožňuje zadat vstupní data dvěma způsoby:

- manuálně,
- ze souboru.

V případě manuálního zadávání koeficientů rovnic je nutné zadat řád soustavy rovnic. Poznamenejme, že manuální zadávání je možné, pokud je řád soustavy menší nebo roven deseti. Pro rozsáhlejší soustavy by bylo manuální zadávání nepohodlné, proto rozsáhlejší rovnice je možné zadávat pouze ze souboru.

Pokud jsme zadali číslo menší nebo rovno deseti, po stisknutí tlačítka *Zadat* se zobrazí formulář B.2. sloupce označené $x_1, \ldots, x_n$ slouží pro zadání jednotlivých koeficientu soustavy, sloupec označený jako *P.Str.* slouží pro zadání pravých stran soustavy.

Nyní tedy můžeme zadat jednotlivé koeficienty rovnic, nevyplněné koeficienty jsou automaticky nastaveny na hodnotu 0. Po stisknutí tlačítka *OK* se zobrazí hlavní okno aplikace, viz obrázek B.1.

Pokud se rozhodneme zadat data ze souboru, označíme přepínač *Ze souboru* v horní části obrazovky. Po stisknutí tlačítka *Procházet* se zobrazí dialog, pomocí kterého můžeme zvolit příslušný textový soubor, ve kterém máme nachystanou soustavu lineárních algebraických rovnic v obecném tvaru (4.5). Na každém řádku vstupního souboru se nachází jedna rovnice. Popsanou situaci lze vidět na obrázku B.3. Příklad vstupního textového souboru je na obrázku B.4. Vstupní soubor může obsahovat i blokové komentáře. Text komentáře je uzavřen mezi dvojicí znaků { a }. Názvy proměnných mohou být různé, jen nesmí začínat prefixem `_tkslc`, protože takto jsou označovány interní proměnné v programu TKSL/C UI.

[1] Dostupný zdarma na `http://www.microsoft.com/en-us/download/details.aspx?id=17718`.

6.2 Nastavení parametrů

Pro nastavení parametrů pro program TKSL/C můžeme využít pravou spodní část okna aplikace. Pro zadání parametru nejprve zatrhneme políčko a poté do textového pole vepíšeme požadovanou hodnotu daného parametru. Význam jednotlivých parametrů byl popsán v tabulce 5.1.

Na obrázku B.5 můžeme vidět situaci po vybrání textového souboru. Obsah souboru se zobrazuje v pravé horní části obrazovky.

6.2.1 Parametr -A a parametr Uživatelská přesnost

Vysvětlíme si nyní rozdíl mezi parametrem -A (Accuracy) a parametrem Uživatelská přesnost. Parametr -A je parametr programu TKSL/C, který zajišťuje nastavení přesnosti tak, jak je zvykem u numerických metod. Tedy kontroluje, zda rozdíl hodnot v poslední aproximaci řešení a předposlední aproximaci řešení je menší než stanovená přesnost daná parametrem -A. Implicitní hodnota tohoto parametru je $1 \cdot 10^{-50}$.

Uživatelská přesnost funguje na podobném principu jako přesnost daná parametrem -A, ovšem bere v úvahu rozdíl hodnot neznámých $x_1, \ldots, x_n$. Máme například soustavu dvou lineárních algebraických rovnic o dvou neznámých, jejichž jména jsou x_1 a x_2. Potom z výsledků, které získáme od programu TKSL/C kontrolujeme, zda rozdíl poslední hodnoty x_1 a předposlední hodnoty x_1 nebo rozdíl poslední hodnoty x_2 a předposlední hodnoty x_1 není menší než uživatelská přesnost. Stačí aby tato podmínka byla splněna u jedné neznámé. Znamená to, že jsme nalezli ustálené řešení soustavy. Po dosažení uživatelské přesnosti výpočet ukončen. Implicitní hodnota tohoto parametru je $1 \cdot 10^{-10}$.

6.2.2 Parametr -t a postup výpočtu

Vysvětlíme nyní, jak do výpočtu zasahuje tento parametr. Parametr -t určuje horní mez výpočtu. Implicitní hodnota je 1. Uživatel může tuto hodnotu libovolně změnit. Řekněme, že parametr -t je nastaven na implicitní hodnotu a uživatel zadal parametr Uživatelská přesnost $1 \cdot 10^{-5}$. Spustíme výpočet s těmito parametry. Jakmile se vrátí výsledky od programu TKSL/C, zkontrolujeme, zda byla hodnota parametru Uživatelská přesnost u některé z neznámých proměnných dosažena. Pokud ne, potom zdvojnásobíme hodnotu parametru -t a spustíme výpočet znovu.

Bylo ovšem nutné ošetřit případ, kdy koeficienty soustavy rovnic jsou zadány tak, že soustava diferenciálních rovnic konverguje velmi pomalu nebo je tuhým systémem. Dalším případem, kdy tato situace může nastat je, pokud uživatel nastaví velkou hodnotu parametru Uživatelská přesnost (například $1 \cdot 10^{-20}$) a malou přesnost parametru -A (například $1 \cdot 10^{-2}$). Proto bylo zvoleno řešení pomocí časovače, který je spuštěn po dobu 30 sekund od spuštění výpočtu. Pokud ani po této době není hodnota parametru Uživatelská přesnost dosažena, potom je výpočet ukončen a zobrazí se varovné hlášení, jak můžeme vidět na obrázku B.6. Zde je tato situace vyvolána velkým rozdílem hodnot mezi parametry -t a Uživatelská přesnost.

6.3 Spuštění výpočtu

Před samotným spuštěním výpočtu si ještě můžeme vybrat, zda budeme chtít výsledky zobrazovat ve vědeckém formátu nebo zda budeme chtít zobrazit postup výpočtu. Pokud jsme zadali vstupní data a případně parametry, lze spustit výpočet pomocí tlačítka *Start*. Zobrazené výsledky i s postupem výpočtu můžeme vidět na obrázku B.7. V levé spodní části okna je uvedena informace, v jaké hodnotě parametru -t bylo dosaženo uživatelské přesnosti.

Aplikaci lze ukončit tlačítkem *Konec*. Pro zobrazení nápovědy k programu TKSL/C UI, k programu TKSL/C či informací o verzích programů slouží menu v levé horní části obrazovky.

Kapitola 7

Závěr

Cílem práce bylo zpracování přehledu metod řešení algebraických rovnic, uvést principy výpočtu algebraických rovnic pomocí rovnic diferenciálních, a také detekovat vznik tuhých systémů. Bylo vytvořeno uživatelské rozhraní TKSL/C UI, které umožňuje převod SLAR na SDR a zajišťuje pohodlnou komunikaci s programem TKSL/C.

Byl proveden rozbor matematických operací přímých i iteračních metod. Výsledkem rozboru bylo nalezení obecných vztahů vyjadřujících počty operací násobení/dělení a sčítání/odčítání. Tyto vztahy je možno uplatnit i při zkoumání výpočetní náročnosti pro libovolný rozměr matic soustavy. Dále byly provedeny experimenty srovnávající výpočetní náročnost v programech TKSL/C a Matlab. Na základě výsledků těchto experimentů lze říci, že program Matlab je rychlejší pro řešení SDR, ovšem jen tehdy, pokud zvolíme správnou metodu. Oproti tomu program TKSL/C nenabízí možnost zvolit si metodu řešení SDR. To ovšem může být výhodné, protože uživatel tedy nemůže zvolit metodu, která by byla pro řešení SDR zcela nevyhovující.

Možným zdokonalením programu TKSLC/UI by mohla být detekce, zda je matice singulární nebo regulární, což by usnadnilo detekci tuhých soustav. Dalším rozšířením by mohla být možnost grafického výstupu aplikace ve formě grafů pro vizualizaci řešení soustav.

Literatura

[1] *Stiff systems.* [cit. 20-4-2012], URL: http://www.biomatematica.it/urbino2002/programmi/Italy1.pdf [online].

[2] *Gaussova eliminace.* [cit. 21-4-2012], URL: http://www.karlin.mff.cuni.cz/~tuma/2002/NLinalg2.pdf [online].

[3] Bartsch, H.-J.: *Matematické vzorce.* 4. vydání, Academia, 2006, ISBN 80-200-1448-9.

[4] Fajmon, B.; Růžičková, I.: *Matematika 3.* Brno: Vysoké učení technické v Brně – Fakulta elektrotechniky a komunikačních technologií.

[5] Kovár, M.: *Maticový a tenzorový počet.* Brno: Vysoké učení technické v Brně – Fakulta elektrotechniky a komunikačních technologií.

[6] Kubíček, M.; Dubcová, M.; Janovská, D.: *Numerické metody a algoritmy.* VŠCHT Praha, 2005, ISBN 80-7080-558-7.

[7] Maplesoft: *Math Software for Engineers, Educators and Students.* [cit. 6-5-2012], URL: http://www.maplesoft.com [online].

[8] The MathWorks: *MATLAB and Simulink for Technical Computing.* [cit. 5-5-2012], URL: http://www.mathworks.com [online].

[9] TKSL software: *High Performance Computing.* [cit. 28-4-2012], URL: http://www.fit.vutbr.cz/~kunovsky/TKSL/index.html.en [online].

[10] Ralston, A.: *Základy numerické matematiky.* Praha: Academia, 1973.

[11] Rektorys, K.: *Přehled užité matematiky.* Praha: Prometheus, 2009, ISBN 978-80-7196-180-2.

[12] Sehnalová, P.: *Konvergence soustav algebraických rovnic.* Diplomová práce, FIT VUT v Brně, 2007.

[13] Vicher, M.: *Numerická matematika.* Univerzita JE Purkyně, 2003.

[14] Vitásek, E.: *Základy teorie numerických metod pro řešení diferenciálních rovnic.* Academia, Praha, 1994, ISBN 80-200-0281-2.

[15] Šátek, V.: *Analýza stiff soustav diferenciálních rovnic.* Disertační práce, FIT VUT v Brně, 2011.

Příloha A

Příklady

Příklad A.1. *Soustavu rovnic řešte pomocí Gaussovy eliminační metody.*

$$A = \left(\begin{array}{ccc|c} 1 & 2 & 3 & 4 \\ 2 & 3 & 5 & 7 \\ 5 & 6 & 2 & 1 \end{array}\right)$$

Řešení pomocí Gaussovy eliminační metody vypadá následovně.

$$\left(\begin{array}{ccc|c} 1 & 2 & 3 & 4 \\ 2 & 3 & 5 & 7 \\ 5 & 6 & 2 & 1 \end{array}\right) \mid \cdot(-2) \approx \left(\begin{array}{ccc|c} 1 & 2 & 3 & 4 \\ 0 & -1 & -1 & -1 \\ 5 & 6 & 2 & 1 \end{array}\right) \mid \cdot(-5) \approx$$

$$\approx \left(\begin{array}{ccc|c} 1 & 2 & 3 & 4 \\ 0 & -1 & -1 & -1 \\ 0 & -4 & -13 & -19 \end{array}\right) \mid \cdot(-4) \approx \left(\begin{array}{ccc|c} 1 & 2 & 3 & 4 \\ 0 & -1 & -1 & -1 \\ 0 & 0 & -9 & -15 \end{array}\right)$$

□

Příklad A.2. *Následující soustavu rovnic, zapsanou v maticové podobě, vyřešte pomocí Gaussovy eliminační metody s částečným výběrem hlavního prvku.*

$$A = \left(\begin{array}{ccc|c} 1 & 2 & 3 & 4 \\ 2 & 3 & 4 & 7 \\ 5 & 6 & 2 & 1 \end{array}\right)$$

Postup výpočtu je uveden níže.

$$\left(\begin{array}{ccc|c} 1 & 2 & 3 & 4 \\ 2 & 3 & 4 & 7 \\ 5 & 6 & 2 & 1 \end{array}\right) \mid \cdot \text{výměna pivota} \approx \left(\begin{array}{ccc|c} 5 & 6 & 2 & 1 \\ 2 & 3 & 4 & 7 \\ 1 & 2 & 3 & 4 \end{array}\right) \mid \cdot -\tfrac{2}{5} \approx$$

$$\approx \left(\begin{array}{ccc|c} 5 & 6 & 2 & 1 \\ 0 & 0,6 & 4,2 & 6,6 \\ 1 & 2 & 3 & 4 \end{array}\right) \mid \cdot -\tfrac{1}{5} \approx \left(\begin{array}{ccc|c} 5 & 6 & 2 & 1 \\ 0 & 0,6 & 4,2 & 6,6 \\ 0 & 0,8 & 2,6 & 3,8 \end{array}\right) \mid \cdot \text{výměna pivota} \approx$$

$$\approx \left(\begin{array}{ccc|c} 5 & 6 & 2 & 1 \\ 0 & 0,8 & 2,6 & 3,8 \\ 0 & 0,6 & 4,2 & 6,6 \end{array}\right) \mid \cdot -\tfrac{3}{4} \approx \left(\begin{array}{ccc|c} 5 & 6 & 2 & 1 \\ 0 & 0,6 & 4,2 & 6,6 \\ 0 & 0 & 2,25 & 3,75 \end{array}\right)$$

Řešení soustavy rovnic je

$$z = \frac{5}{3},\ y = -\frac{2}{3},\ x = \frac{1}{3}.$$

□

Příklad A.3. *Vypočtěte následující soustavu lineárních rovnic pomocí Gaussovy eliminace s úplným výběrem hlavního prvku.*

$$\left(\begin{array}{ccc|c} 1 & 2 & 3 & 14 \\ 4 & 5 & 6 & 32 \\ 9 & 6 & 5 & 36 \end{array}\right) \approx \left(\begin{array}{ccc|c} 9 & 6 & 5 & 36 \\ 4 & 5 & 6 & 32 \\ 1 & 2 & 3 & 14 \end{array}\right) \ |\cdot\tfrac{4}{9} \approx \left(\begin{array}{ccc|c} 9 & 6 & 5 & 36 \\ 0 & -\frac{7}{3} & -\frac{34}{9} & -16 \\ 1 & 2 & 3 & 14 \end{array}\right) \ |\cdot\tfrac{1}{9} \approx$$

$$\approx \left(\begin{array}{ccc|c} 9 & 6 & 5 & 36 \\ 0 & -\frac{7}{3} & -\frac{34}{9} & -16 \\ 0 & -\frac{12}{9} & -\frac{22}{9} & -10 \end{array}\right) \approx \left(\begin{array}{ccc|c} 9 & 6 & 5 & 36 \\ 0 & -\frac{21}{9} & -\frac{34}{9} & -16 \\ 0 & -\frac{12}{9} & -\frac{22}{9} & -10 \end{array}\right) \approx$$

$$\approx \left(\begin{array}{ccc|c} 9 & 6 & 5 & 36 \\ 0 & -\frac{34}{9} & -\frac{21}{9} & -16 \\ 0 & -\frac{22}{9} & -\frac{12}{9} & -10 \end{array}\right) \ |\cdot\tfrac{11}{17} \approx \left(\begin{array}{ccc|c} 9 & 6 & 5 & 36 \\ 0 & -\frac{34}{9} & -\frac{21}{9} & -16 \\ 0 & 0 & -\frac{145}{51} & -\frac{346}{17} \end{array}\right)$$

□

Příklad A.4. *Vyřešte soustavu rovnic pomocí Gauss-Jordanovy metody.*

$$\left(\begin{array}{ccc|c} 1 & 2 & 3 & 4 \\ 2 & 3 & 5 & 7 \\ 5 & 6 & 2 & 1 \end{array}\right) \ |\cdot(-2) \approx \left(\begin{array}{ccc|c} 1 & 2 & 3 & 4 \\ 0 & -1 & -1 & -1 \\ 5 & 6 & 2 & 1 \end{array}\right) \ |\cdot(-5) \approx$$

$$\approx \left(\begin{array}{ccc|c} 1 & 2 & 3 & 4 \\ 0 & -1 & -1 & -1 \\ 0 & -4 & -13 & -19 \end{array}\right) \ |\cdot(-4) \approx \left(\begin{array}{ccc|c} 1 & 2 & 3 & 4 \\ 0 & -1 & -1 & -1 \\ 0 & 0 & -9 & -15 \end{array}\right) \ |\cdot(-1) \approx$$

$$\approx \left(\begin{array}{ccc|c} 1 & 2 & 3 & 4 \\ 0 & 1 & 1 & 1 \\ 0 & 0 & -9 & -15 \end{array}\right) \ |\cdot -\tfrac{1}{9} \approx \left(\begin{array}{ccc|c} 1 & 2 & 3 & 4 \\ 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & \frac{5}{3} \end{array}\right) \ |\cdot(-1) \approx$$

$$\approx \left(\begin{array}{ccc|c} 1 & 2 & 3 & 4 \\ 0 & 1 & 0 & -\frac{2}{3} \\ 0 & 0 & 1 & \frac{5}{3} \end{array}\right) \ |\cdot(-3) \approx \left(\begin{array}{ccc|c} 1 & 2 & 0 & -1 \\ 0 & 1 & 0 & -\frac{2}{3} \\ 0 & 0 & 1 & \frac{5}{3} \end{array}\right) \ |\cdot(-2) \approx$$

$$\approx \left(\begin{array}{ccc|c} 1 & 0 & 0 & \frac{1}{3} \\ 0 & 1 & 0 & -\frac{2}{3} \\ 0 & 0 & 1 & \frac{5}{3} \end{array}\right)$$

Ihned lze vyčíst řešení soustavy

$$x_1 = \frac{1}{3},\ x_2 = -\frac{2}{3},\ x_3 = \frac{5}{3}.$$

□

Příklad A.5. *Je dána soustava rovnic o 3 neznámých. Řešte soustavu pomocí Gauss-Jordanovy eliminační metody.*

$$\begin{aligned} 3x - 2y + 2z &= 2 \\ 2x + 3z &= 3 \\ 6x - 3y + 2z &= 1 \end{aligned}$$

$$\left(\begin{array}{ccc|ccc} 3 & -2 & 2 & 1 & 0 & 0 \\ 2 & 0 & 3 & 0 & 1 & 0 \\ 6 & -3 & 2 & 0 & 0 & 1 \end{array}\right) \begin{array}{l} \\ \\ \\ \end{array} \begin{array}{l} \\ \\ \cdot \frac{2}{3} \end{array} \approx \left(\begin{array}{ccc|ccc} 3 & -2 & 2 & 1 & 0 & 0 \\ 0 & \frac{4}{3} & \frac{5}{3} & -\frac{2}{3} & 1 & 0 \\ 6 & -3 & 2 & 0 & 0 & 1 \end{array}\right) \begin{array}{l} \\ \\ \cdot (-2) \end{array} \approx$$

$$\approx \left(\begin{array}{ccc|ccc} 3 & -2 & 2 & 1 & 0 & 0 \\ 0 & \frac{4}{3} & \frac{5}{3} & -\frac{2}{3} & 1 & 0 \\ 0 & 1 & -2 & -2 & 0 & 1 \end{array}\right) \begin{array}{l} \\ \\ \cdot -\frac{4}{3} \end{array} \approx \left(\begin{array}{ccc|ccc} 3 & -2 & 2 & 1 & 0 & 0 \\ 0 & \frac{4}{3} & \frac{5}{3} & -\frac{2}{3} & 1 & 0 \\ 0 & 0 & \frac{13}{3} & 2 & 1 & -\frac{4}{3} \end{array}\right) \begin{array}{l} \\ \\ \cdot \frac{1}{3} \end{array} \approx$$

$$\approx \left(\begin{array}{ccc|ccc} 1 & -\frac{2}{3} & \frac{2}{3} & \frac{1}{3} & 0 & 0 \\ 0 & \frac{4}{3} & \frac{5}{3} & -\frac{2}{3} & 1 & 0 \\ 0 & 0 & \frac{13}{3} & 2 & 1 & -\frac{4}{3} \end{array}\right) \begin{array}{l} \\ \\ \cdot \frac{4}{3} \end{array} \approx \left(\begin{array}{ccc|ccc} 1 & -\frac{2}{3} & \frac{2}{3} & \frac{1}{3} & 0 & 0 \\ 0 & 1 & \frac{5}{4} & -\frac{1}{2} & \frac{3}{4} & 0 \\ 0 & 0 & \frac{13}{3} & 2 & 1 & -\frac{4}{3} \end{array}\right) \begin{array}{l} \\ \\ \cdot \frac{1}{13} \end{array} \approx$$

$$\approx \left(\begin{array}{ccc|ccc} 1 & -\frac{2}{3} & \frac{2}{3} & \frac{1}{3} & 0 & 0 \\ 0 & 1 & \frac{5}{4} & -\frac{1}{2} & \frac{3}{4} & 0 \\ 0 & 0 & 1 & \frac{6}{13} & \frac{3}{13} & \frac{4}{13} \end{array}\right) \begin{array}{l} \\ \\ \cdot -\frac{5}{4} \end{array} \approx \left(\begin{array}{ccc|ccc} 1 & -\frac{2}{3} & \frac{2}{3} & \frac{1}{3} & 0 & 0 \\ 0 & 1 & 0 & -\frac{14}{13} & \frac{6}{13} & -\frac{5}{13} \\ 0 & 0 & 1 & \frac{6}{13} & \frac{3}{13} & \frac{4}{13} \end{array}\right) \begin{array}{l} \\ \\ \cdot -\frac{2}{3} \end{array} \approx$$

$$\approx \left(\begin{array}{ccc|ccc} 1 & -\frac{2}{3} & 0 & \frac{1}{39} & -\frac{2}{13} & -\frac{8}{39} \\ 0 & 1 & 0 & -\frac{14}{13} & \frac{6}{13} & -\frac{5}{13} \\ 0 & 0 & 1 & \frac{6}{13} & \frac{3}{13} & \frac{4}{13} \end{array}\right) \begin{array}{l} \\ \\ \cdot \frac{2}{3} \end{array} \approx \left(\begin{array}{ccc|ccc} 1 & 0 & 0 & -\frac{9}{13} & \frac{2}{13} & -\frac{6}{13} \\ 0 & 1 & 0 & -\frac{14}{13} & \frac{6}{13} & -\frac{5}{13} \\ 0 & 0 & 1 & \frac{6}{13} & \frac{3}{13} & \frac{4}{13} \end{array}\right)$$

Pro nalezení vektoru neznámých proměnných musíme vypočítat $x = A^{-1} \cdot b$.

$$\begin{pmatrix} -\frac{9}{13} & \frac{2}{13} & -\frac{6}{13} \\ -\frac{14}{13} & \frac{6}{13} & -\frac{5}{13} \\ \frac{6}{13} & \frac{3}{13} & \frac{4}{13} \end{pmatrix} \begin{pmatrix} 2 \\ 1 \\ 3 \end{pmatrix} = \begin{pmatrix} -\frac{18}{13} \\ -\frac{15}{13} \\ \frac{25}{13} \end{pmatrix}$$

□

Příklad A.6. *Najděte inverzní matici pomocí determinantů.*

$$A = \begin{pmatrix} 3 & -4 & 5 \\ 2 & -3 & 1 \\ 3 & -5 & -1 \end{pmatrix}$$

Nejprve vypočítáme determinant matice $\mathbf{A}$.

$$A = \begin{vmatrix} 3 & -4 & 5 \\ 2 & -3 & 1 \\ 3 & -5 & -1 \end{vmatrix} = 9 - 12 - 50 - 8 + 15 + 45 = -1$$

Nyní postupně vytvoříme n^2 *determinantů řádu 2 a postupně je vyčíslíme.*

$$a_{11}^{-1} = \begin{vmatrix} -3 & 1 \\ -5 & -1 \end{vmatrix} = 8$$

$$a_{12}^{-1} = \begin{vmatrix} -4 & 5 \\ -5 & -1 \end{vmatrix} = 29$$

$$a_{13}^{-1} = \begin{vmatrix} -4 & 5 \\ -3 & 1 \end{vmatrix} = 11$$

$$a_{21}^{-1} = \begin{vmatrix} 2 & 1 \\ 3 & -1 \end{vmatrix} = -5$$

$$a_{22}^{-1} = \begin{vmatrix} 3 & 5 \\ 3 & -1 \end{vmatrix} = -18$$

$$a_{23}^{-1} = \begin{vmatrix} 3 & 5 \\ 2 & 1 \end{vmatrix} = -7$$

$$a_{31}^{-1} = \begin{vmatrix} 2 & -3 \\ 3 & -5 \end{vmatrix} = -1$$

$$a_{32}^{-1} = \begin{vmatrix} 3 & -4 \\ 3 & -5 \end{vmatrix} = -3$$

$$a_{33}^{-1} = \begin{vmatrix} 3 & -4 \\ 2 & -3 \end{vmatrix} = -1$$

Nyní postupujeme podle vzorce, který jsme uvedli na začátku podkapitoly 2.27.

$$a_{11}^{-1} = \frac{(-1)^{1+1} \cdot 8}{-1} = -8$$

$$a_{12}^{-1} = \frac{(-1)^{1+2} \cdot 29}{-1} = 29$$

$$a_{13}^{-1} = \frac{(-1)^{1+3} \cdot 11}{-1} = -11$$

$$a_{21}^{-1} = \frac{(-1)^{2+1} \cdot (-5)}{-1} = -5$$

$$a_{22}^{-1} = \frac{(-1)^{2+2} \cdot (-18)}{-1} = 18$$

$$a_{23}^{-1} = \frac{(-1)^{2+3} \cdot 8}{(-7)} = -7$$

$$a_{31}^{-1} = \frac{(-1)^{3+1} \cdot (-1)}{-1} = 1$$

$$a_{32}^{-1} = \frac{(-1)^{3+2} \cdot 8}{(-3)} = -3$$

$$a_{33}^{-1} = \frac{(-1)^{3+3} \cdot (-1)}{-1} = 1$$

Výsledná inverzní matice je uvedena níže.

$$A^{-1} = \begin{pmatrix} -8 & 29 & -11 \\ -5 & 18 & -7 \\ 1 & -3 & 1 \end{pmatrix}$$

Pro získání výsledku musíme vypočítat vektor neznámých dle vztahu $x = A^{-1} \cdot b$.

$$x = \begin{pmatrix} -8 & 29 & -11 \\ -5 & 18 & -7 \\ 1 & -3 & 1 \end{pmatrix} \cdot \begin{pmatrix} 5 \\ 4 \\ 2 \end{pmatrix} = \begin{pmatrix} 54 \\ 33 \\ -5 \end{pmatrix}$$

□

Příklad A.7. *Je dána soustava dvou lineárních rovnic.*

$$x + y = 1$$
$$x - y = 3$$

Upravíme ji na následující tvar.

$$x + y - 1 = 0$$
$$x - y - 3 = 0$$

1. rovnice – základní tvar.

$$x + y - 1 = x'$$
$$x - y - 3 = y'$$

2. rovnice – změna znaménka u 1.rovnice.

$$-x - y + 1 = x'$$
$$x - y - 3 = y'$$

3. rovnice – změna znaménka u 2.rovnice.

$$x + y - 1 = x'$$
$$-x + y + 3 = y'$$

4. rovnice – změna znaménka u obou rovnic.

$$-x - y + 1 = x'$$
$$-x + y + 3 = y'$$

Nyní provedeme daší druh úpravy, zaměníme pořadí diferenciálních proměnných.
5. rovnice - záměna proměnných – vycházíme z 1. rovnice.

$$x + y - 1 = y'$$
$$x - y - 3 = x'$$

6. rovnice - záměna proměnných – vycházíme z 2. rovnice.

$$-x - y + 1 = y'$$
$$x - y - 3 = x'$$

7. rovnice - záměna proměnných – vycházíme z 3. rovnice.

$$x + y - 1 = y'$$
$$-x + y + 3 = x'$$

8. rovnice - záměna proměnných – vycházíme z 4. rovnice.

$$-x - y + 1 = y'$$
$$-x + y + 3 = x'$$

Celkem vychází tedy 8 možností zápisu této soustavy lineárních rovnic.

□

Příklad A.8. *Je dána soustava tří lineárních rovnic.*

$$x + y + z = 1$$
$$x - y - z = 3$$
$$x + y - z = 5$$

Upravíme na tvar uvedený níže.

$$x + y + z - 1 = 0$$
$$x - y - z - 3 = 0$$
$$x + y - z - 5 = 0$$

1. rovnice – základní tvar.

$$x + y + z - 1 = x'$$
$$x - y - z - 3 = y'$$
$$x + y - z - 5 = z'$$

2. rovnice – změna znaménka u 1. rovnice.

$$-x - y - z + 1 = x'$$
$$x - y - z - 3 = y'$$
$$x + y - z - 5 = z'$$

3. rovnice – změna znaménka u 2. rovnice.

$$\begin{aligned} x+y+z-1 &= x' \\ -x+y+z+3 &= y' \\ x+y-z-5 &= z' \end{aligned}$$

4. rovnice – změna znaménka u 3. rovnice.

$$\begin{aligned} x+y+z-1 &= x' \\ x-y-z-3 &= y' \\ -x-y+z+5 &= z' \end{aligned}$$

5. rovnice – změna znaménka u 1. a 2. rovnice.

$$\begin{aligned} -x-y-z+1 &= x' \\ -x+y+z+3 &= y' \\ x+y-z-5 &= z' \end{aligned}$$

6. rovnice – změna znaménka u 2. a 3. rovnice.

$$\begin{aligned} x+y+z-1 &= x' \\ -x+y+z+3 &= y' \\ -x-y+z+5 &= z' \end{aligned}$$

7. rovnice – změna znaménka u 1. a 3. rovnice.

$$\begin{aligned} -x-y-z+1 &= x' \\ x-y-z-3 &= y' \\ -x-y+z+5 &= z' \end{aligned}$$

8. rovnice – změna znaménka u všech 3 rovnic.

$$\begin{aligned} -x-y-z+1 &= x' \\ -x+y+z+3 &= y' \\ -x-y+z+5 &= z' \end{aligned}$$

Nyní provedeme další druh úpravy, zaměníme pořadí derivovaných proměnných.
9. rovnice - záměna proměnných.

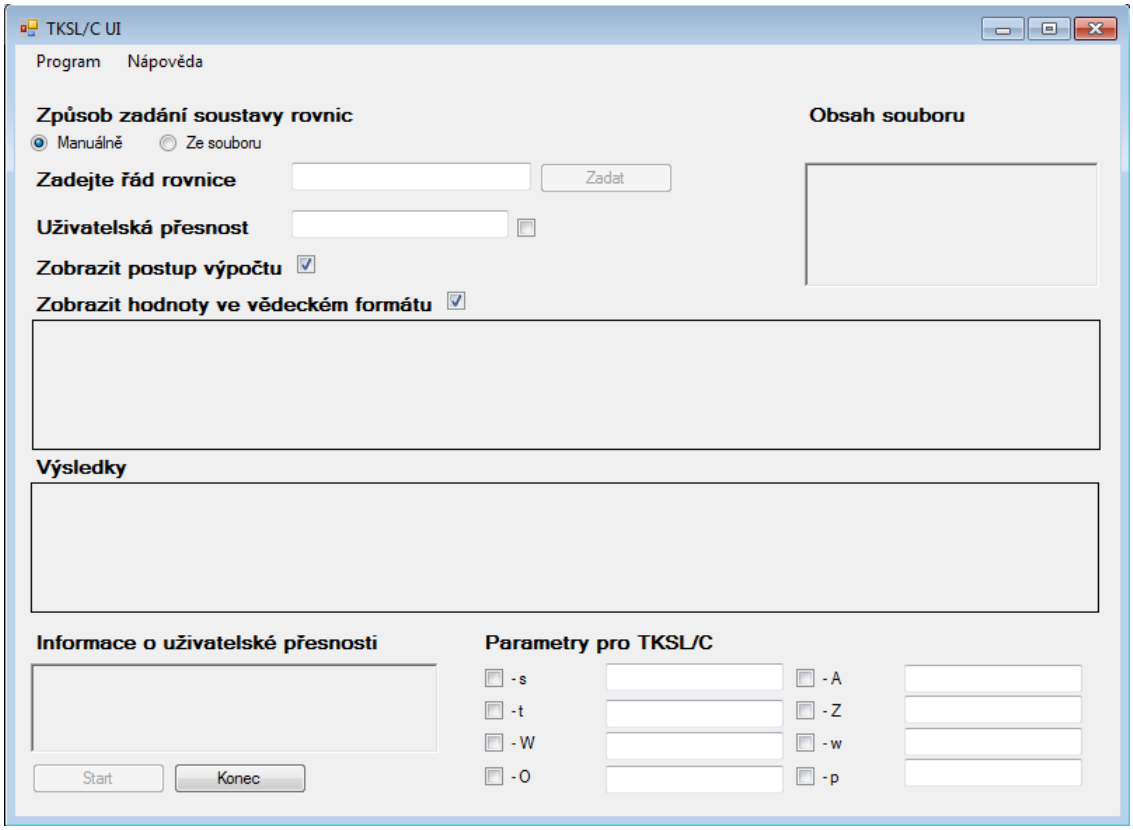
$$\begin{aligned} x+y+z-1 &= x' \\ x-y-z-3 &= z' \\ x+y-z-5 &= y' \end{aligned}$$

$$\dots$$

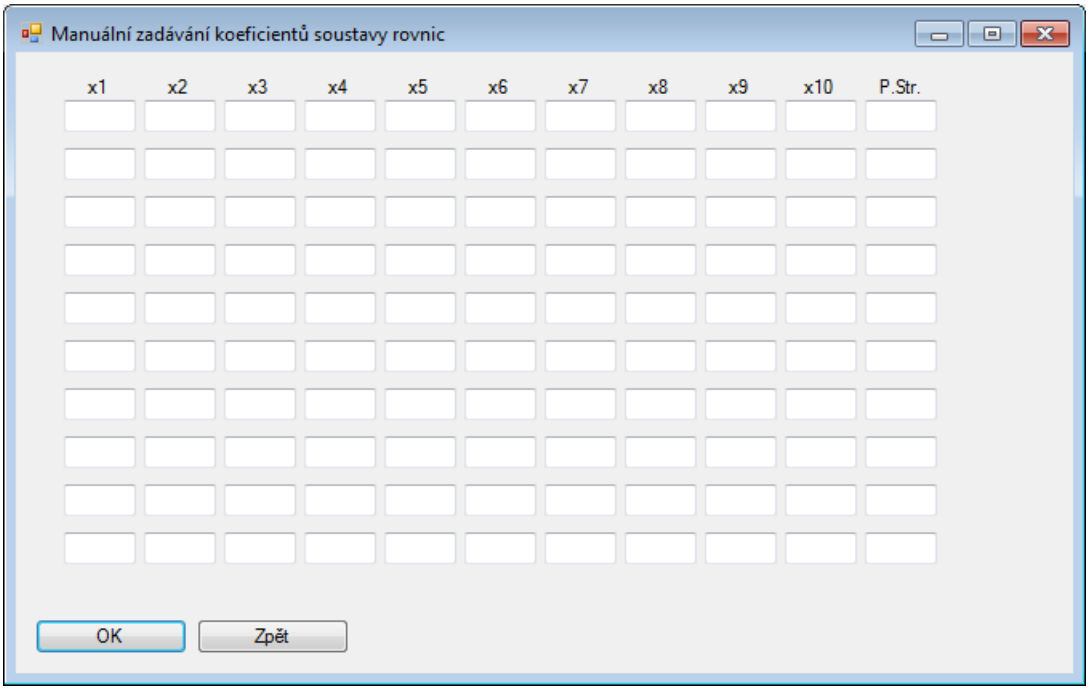
□

Příloha B

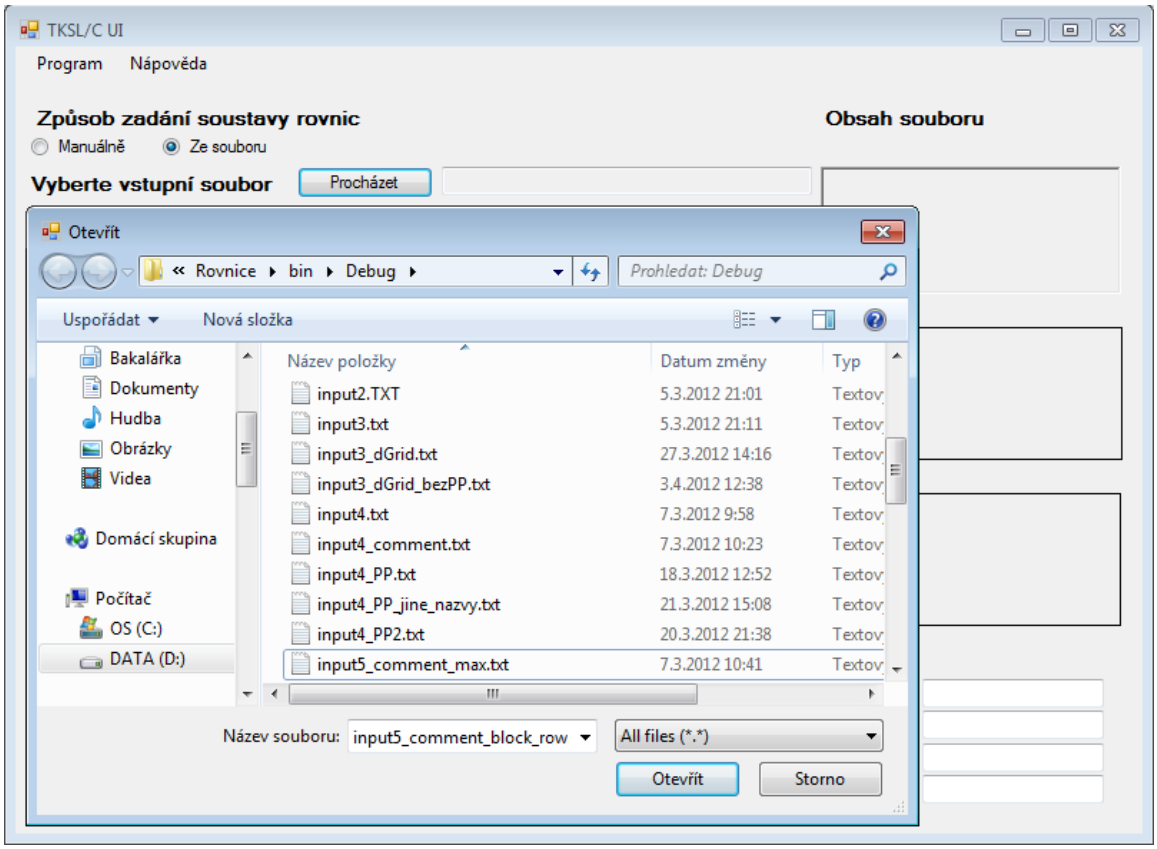
Obrázky



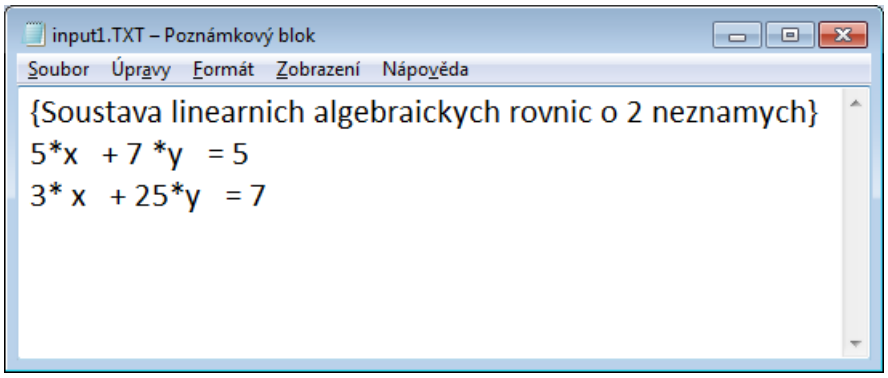
Obrázek B.1: Uživatelské rozhraní pro TKSL/C



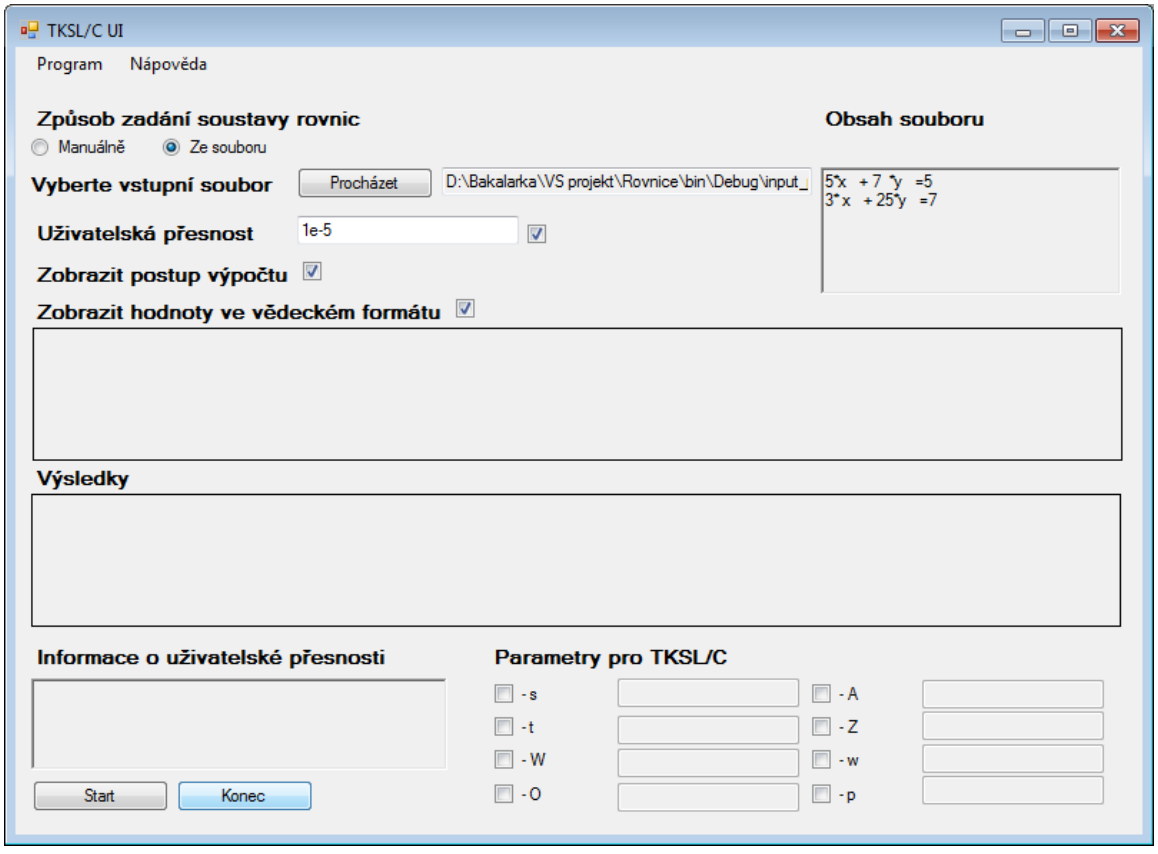
Obrázek B.2: Uživatelské rozhraní pro TKSL/C – manuální zadávání koeficientů rovnic



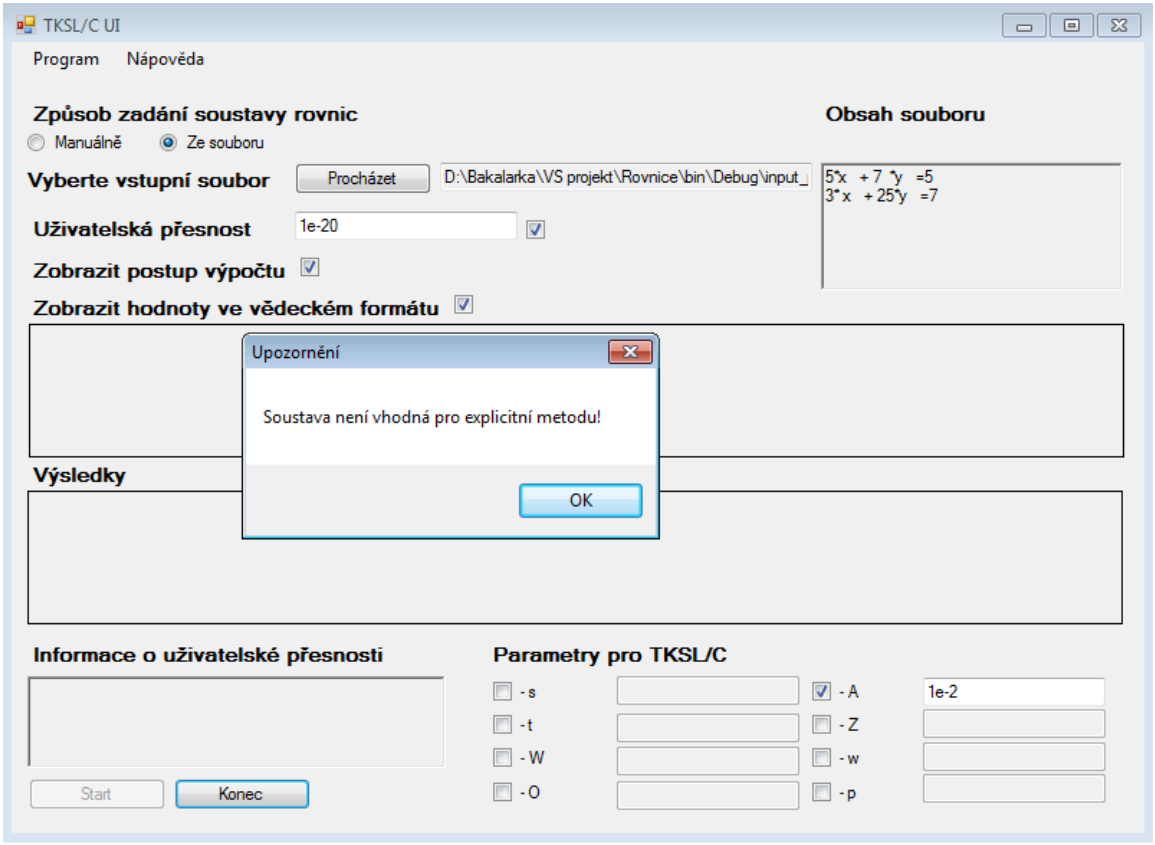
Obrázek B.3: Uživatelské rozhraní pro TKSL/C – zadávání dat z textového souboru



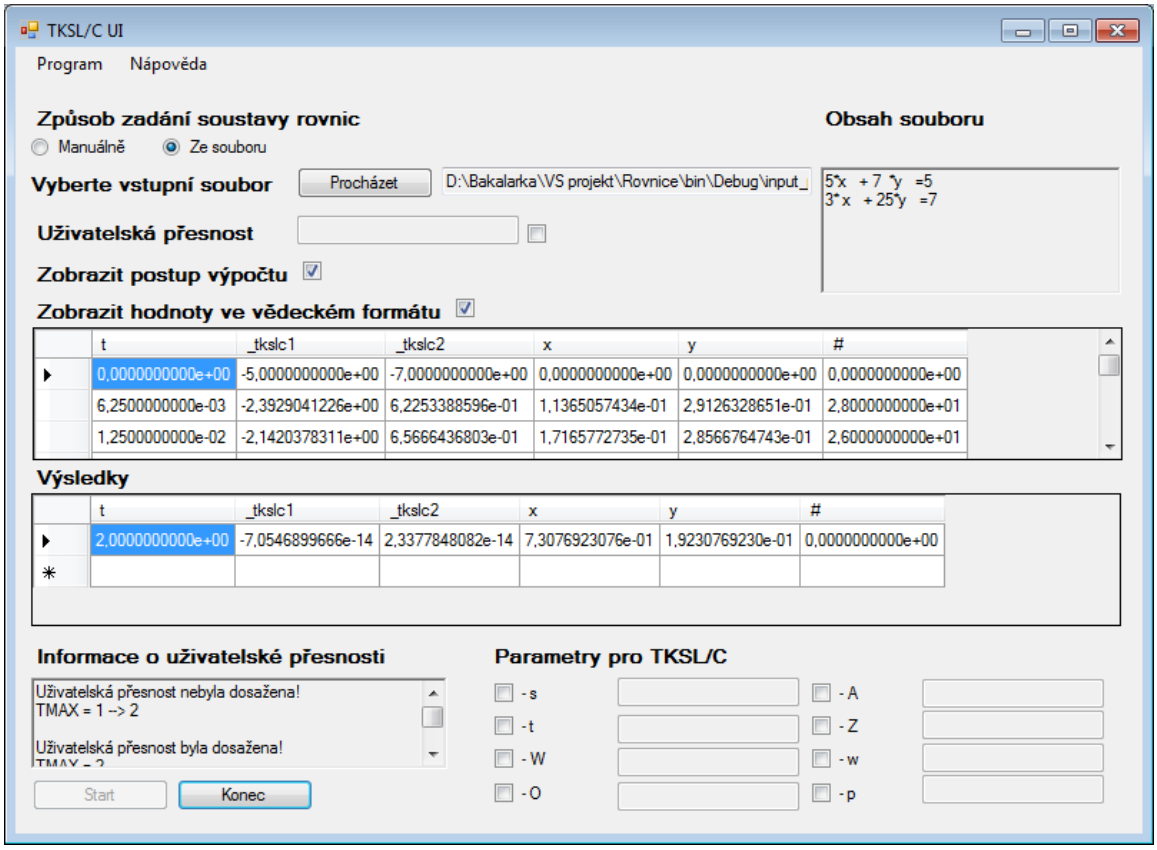
Obrázek B.4: Uživatelské rozhraní pro TKSL/C – formát textového souboru



Obrázek B.5: Uživatelské rozhraní pro TKSL/C – po vybrání dat z textového souboru



Obrázek B.6: Uživatelské rozhraní pro TKSL/C – varovné hlášení při nedosažení hodnoty parametru Uživatelská přesnost



Obrázek B.7: Uživatelské rozhraní pro TKSL/C – zobrazení výsledků výpočtu

Příloha C

Obsah přiloženého CD

Přiložené CD obsahuje následující položky:

- soubor `BP_Necasova.pdf` – technická zpráva bakalářské práce,
- složka `BP_Text` – zdrojové kódy a další soubory bakalářské práce,
- složka `BP_UI` – zdrojové kódy uživatelského rozhraní TKSL/C UI,
- složka `BP_Matlab` – zdrojové kódy, které byly použity pro provádění experimentů v kapitole 5.