VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

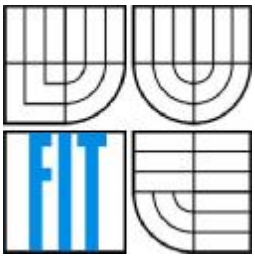
KONVERGENCE ŘEŠENÍ SOUSTAV ALGEBRAICKÝCH ROVNIC

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE PAVLA SEHNALOVÁ
AUTHOR

BRNO 2007

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

KONVERGENCE ŘEŠENÍ SOUSTAV ALGEBRAICKÝCH ROVNIC

ALGEBRAIC EQUATIONS SOLUTION CONVERGENCE

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE
AUTHOR

PAVLA SEHNALOVÁ

VEDOUCÍ PRÁCE
SUPERVISOR

DOC. ING. JIŘÍ KUNOVSKÝ, CSC.

BRNO 2007

Vysoké učení technické v Brně - Fakulta informačních technologií

Ústav inteligentních systémů Akademický rok 2006/2007

Zadání diplomové práce

Řešitel: **Sehnalová Pavla**
Obor: Výpočetní technika a informatika
Téma: **Konvergence řešení soustav algebraických rovnic**
Kategorie: Modelování a simulace

Pokyny:

1. Seznamte se s metodami řešení algebraických rovnic.
2. Zpracujte a analyzujte metodu převodu výpočtu algebraických rovnic na výpočet diferenciálních rovnic a porovnejte s analyzou doby výpočtu u přímých metod (podrobným rozborem matematických operací).
3. Navrhněte a implementujte uživatelské rozhraní pro výpočet a grafické rozhraní řešení soustavy algebraických rovnic.
4. Analyzujte vznik tuhých systémů při řešení algebraických rovnic převodem na diferenciální rovnice a zhodnoťte souvislost tuhých systémů se špatně podmíněnými soustavami algebraických rovnic.
5. Analyzujte iterační metody výpočtu a zhodnoťte rychlost konvergence řešení.

Literatura:

- Dle zadání vedoucího

Při obhajobě semestrální části diplomového projektu je požadováno:

- Bez požadavků.

Podrobné závazné pokyny pro vypracování diplomové práce naleznete na adrese http://www.fit.vutbr.cz/info/szz/

Technická zpráva diplomové práce musí obsahovat formulaci cíle, charakteristiku současného stavu, teoretická a odborná východiska řešených problémů a specifikaci etap, které byly vyřešeny v rámci ročníkového a semestrálního projektu (30 až 40% celkového rozsahu technické zprávy).

Student odevzdá v jednom výtisku technickou zprávu a v elektronické podobě zdrojový text technické zprávy, úplnou programovou dokumentaci a zdrojové texty programů. Informace v elektronické podobě budou uloženy na standardním paměťovém médiu (disketa, CD-ROM), které bude vloženo do písemné zprávy tak, aby nemohlo dojít k jeho ztrátě při běžné manipulaci.

Vedoucí: **Kunovský Jiří, doc. Ing., CSc.**, UITS FIT VUT
Datum zadání: 1. listopadu 2006
Datum odevzdání: 22. května 2007

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
Fakulta informačních technologií
Ústav inteligentních systémů
612 66 Brno, Božetěchova 2

L.S.

doc. Dr. Ing. Petr Hanáček
vedoucí ústavu

LICENČNÍ SMLOUVA
POSKYTOVANÁ K VÝKONU PRÁVA UŽÍT ŠKOLNÍ DÍLO

uzavřená mezi smluvními stranami

1. **Slečna**

Jméno a příjmení: **Pavla Selnalová**
Id studenta: 22855
Bytem: Gregorova 2432/6, 702 00 Ostrava
Narozena: 18. 02. 1982, Ostrava
(dále jen "autor")

a

2. **Vysoké učení technické v Brně**

Fakulta informačních technologií
se sídlem Božetěchova 2/1, 612 66 Brno, IČO 00216305
jejímž jménem jedná na základě písemného pověření děkanem fakulty:

.....
(dále jen "nabyvatel")

Článek 1
Specifikace školního díla

1. Předmětem této smlouvy je vysokoškolská kvalifikační práce (VŠKP):
diplomová práce

Název VŠKP: Konvergence řešení soustav algebraických rovnic
Vedoucí/školitel VŠKP: Kunovský Jiří, doc. Ing., CSc.
Ústav: Ústav inteligentních systémů
Datum obhajoby VŠKP: .....

VŠKP odevzdal autor nabyvateli v:

tištěné formě počet exemplářů: 1
elektronické formě počet exemplářů: 2 (1 ve skladu dokumentů, 1 na CD)

2. Autor prohlašuje, že vytvořil samostatnou vlastní tvůrčí činností dílo shora popsané a specifikované. Autor dále prohlašuje, že při zpracovávání díla se sám nedostal do rozporu s autorským zákonem a předpisy souvisejícími a že je dílo dílem původním.
3. Dílo je chráněno jako dílo dle autorského zákona v platném znění.
4. Autor potvrzuje, že listinná a elektronická verze díla je identická.

Článek 2
Udělení licenčního oprávnění

1. Autor touto smlouvou poskytuje nabyvateli oprávnění (licenci) k výkonu práva uvedené dílo nevýdělečně užít, archivovat a zpřístupnit ke studijním, výukovým a výzkumným účelům včetně pořizování výpisů, opisů a rozmnoženin.
2. Licence je poskytována celosvětově, pro celou dobu trvání autorských a majetkových práv k dílu.
3. Autor souhlasí se zveřejněním díla v databázi přístupné v mezinárodní síti:
   - ☐ ihned po uzavření této smlouvy
   - ☐ 1 rok po uzavření této smlouvy
   - ☐ 3 roky po uzavření této smlouvy
   - ☒ 5 let po uzavření této smlouvy
   - ☐ 10 let po uzavření této smlouvy

   (z důvodu utajení v něm obsažených informací)
4. Nevýdělečné zveřejňování díla nabyvatelem v souladu s ustanovením § 47b zákona č. 111/ 1998 Sb., v platném znění, nevyžaduje licenci a nabyvatel je k němu povinen a oprávněn ze zákona.

Článek 3
Závěrečná ustanovení

1. Smlouva je sepsána ve třech vyhotoveních s platností originálu, přičemž po jednom vyhotovení obdrží autor a nabyvatel, další vyhotovení je vloženo do VŠKP.
2. Vztahy mezi smluvními stranami vzniklé a neupravené touto smlouvou se řídí autorským zákonem, občanským zákoníkem, vysokoškolským zákonem, zákonem o archivnictví, v platném znění a popř. dalšími právními předpisy.
3. Licenční smlouva byla uzavřena na základě svobodné a pravé vůle smluvních stran, s plným porozuměním jejímu textu i důsledkům, nikoliv v tísni a za nápadně nevýhodných podmínek.
4. Licenční smlouva nabývá platnosti a účinnosti dnem jejího podpisu oběma smluvními stranami.

V Brně dne: .....

....

Nabyvatel Autor

Abstrakt

Práce podrobně popisuje metody řešení soustav lineárních algebraických a diferenciálních rovnic. Představuje metodu převodu ze soustav lineárních algebraických rovnic na soustavy rovnic diferenciálních. Vysvětluje metodu elementárního převodu, převod pomocí transformačního algoritmu a oba postupy demonstruje na jednoduchých příkladech s ukázkou jejich vlastností. Práce srovnává metody řešení soustav rovnic z hlediska přesnosti a rychlosti. Pro řešení příkladů a experimenty byly použity programy TKSL a TKSL/C. Program TKSL/C byl v rámci práce rozšířen o grafické uživatelské rozhraní určené k automatickému převodu soustav a jejich výpočtu.

Klíčová slova

Lineární algebraická rovnice, diferenciální rovnice, soustava rovnic, numerické metody, přímé metody, iterační metody, konvergence, tuhé systémy, TKSL, TKSL/C, TKSL/C - SLAR.

Abstract

The work describes techniques for solving systems of linear and differential equations. It explains the definition of conversion from system of linear to system of differential equations. The method of the elementary transmission and the transform algorithm are presented. Both of methods are demonstrated on simply examples and properties of conversion are shown. The work compares fast and accurate solutions of methods and algorithm. For computing examples and solving experiments following programs were used: TKSL and TKSL/C. The program TKSL/C was enriched with the graphic user interface which makes the conversion of systems and computing results easier.

Keywords

Linear algebraic equation, differential equation, system of equations, numeric method, direct method, iterative method, convergence, stiff system, TKSL, TKSL/C, TKSL/C - SLAR.

Citace

Pavla Sehnalová: Název práce v jazyce práce, diplomová práce, Brno, FIT VUT v Brně, 2007

Konvergence řešení soustav algebraických rovnic

Prohlášení

Prohlašuji, že jsem tuto diplomovou práci vypracovala samostatně pod vedením doc. Ing. Jiřím Kunovským, CSc.
Uvedla jsem všechny literární prameny a publikace, ze kterých jsem čerpala.

……
Pavla Sehnalová
15. 5. 2007

Poděkování

Velice děkuji panu doc. Ing. Jiřímu Kunovskému, CSc. za poskytnuté odborné rady a informace a pomoc při práci na práci.

Obsah

1 Úvod

1.1 Seznámení

Soustavy lineárních rovnic lze řešit několika způsoby. Buď známými numerickými metodami (přímými, iteračními nebo gradientními), nebo pomocí převodu do soustavy diferenciálních rovnic. Dostaneme-li k řešení soustavu, která obsahuje stovky až tisíce rovnic, výpočty numerických metod budou velice časově náročné.

Pro převod soustavy lineárních rovnic na soustavu diferenciálních rovnic a pro získání řešení soustavy používáme program TKSL. Rovnice je možné zadat v podobě, kterou dostaneme elementárním převodem z lineárních rovnic nebo v podobě obdržené stabilní transformací. Způsob řešení pomocí diferenciálních rovnic může být v mnoha případech rychlejší a lze jím vyřešit možné problémy, které se při řešení vyskytují.

1.2 Cíl práce

Cílem je vytvořit kompletní matematický i programátorský pohled na řešení soustav lineárních rovnic s očekáváním co nejrychlejších výsledků a odstranění problémů v kritických soustavách.

V každé z oblastí klasických metod řešení soustav lineárních rovnic jsou ještě nepopsané nebo nevyřešené problémy a otázky. Pokusíme se nalézt ukazatele či indikátory možných problémových rovnic a jejich řešení.

V následujících kapitolách se seznámíme s typy převodů soustav lineárních algebraických rovnic na soustavy rovnic diferenciálních. Na modelových příkladech si metody převodu předvedeme a ukážeme na důležité vlastnosti. Zajímat nás bude zejména stabilita řešení a způsoby, jak tuto stabilitu při převodu soustav zajistit. V každé z oblastí se pokusíme nalézt pravidla nebo vlastnosti převodu, které nám zaručí, že získané řešení zadané soustavy bude stabilní. Dalším důležitým faktorem bude rychlost konvergence jednotlivých metod.

Zpráva je rozdělena do sedmi kapitol. Na přiloženém CD-ROM jsou uvedeny zdrojové texty použitých příkladů pro program TKSL (verze 386), zdrojový text rozhraní TKSL/C-SLAR a uživatelská příručka pro tento program, který vznikl v rámci diplomové práce. Následuje stručný výčet obsahu jednotlivých kapitol.

Kapitola 2 vysvětluje teorii soustav lineárních algebraických rovnic a metody, které jsou nejběžněji používány k řešení. Na příkladech jsou použité matematické definice názorně předvedeny k pochopení principu metod.

Kapitola 3 představuje pohled do teorie soustav diferenciálních rovnic. Pomocí řešených příkladů si opět předvedeme metody řešení soustav diferenciálních rovnic a v závěru kapitoly jsou uvedeny problémy, které se při řešení pomocí klasických metod mohou vyskytovat.

Obě kapitoly s teoretickým seznámením byly vytvořeny v návaznosti na ročníkovou a semestrální práci.

Kapitola 4 přináší nový pohled na řešený problém a vysvětluje souvislost soustav lineárních a diferenciálních rovnic. Seznamuje s metodami převodu mezi jednotlivými soustavami a na konkrétních příkladech si vysvětlíme chování soustav a principy řešení. Některé vlastnosti elementárního převodu uvedené v této kapitole byly experimentálně zjišťovány již v semestrální práci.

V kapitole 5 je uvedeno srovnání klasických metod řešení s metodou převodu vzhledem k různým aspektům výpočtu.

Kapitola 6 popisuje program TKSL/C, který vznikl na Fakultě informačních technologií VUT v Brně. Detailně seznamuje s rozhraním TKSL/C-SLAR, které bylo vytvořeno výhradně se zaměřením na řešení soustav lineárních algebraických rovnic. Rozhraní bylo napsáno pro program TKSL/C a umožňuje výpočty bez nutnosti vytváření dalších zdrojových textů.

Dosažené výsledky a zhodnocení jsou uvedeny v kapitole 7 spolu s naznačením směru dalšího vývoje a možnosti zkoumání.

2 Soustava lineárních algebraických rovnic

Soustava lineárních algebraických rovnic (dále jen SLAR) v obecném tvaru je zadána:

$$\sum_{i=1}^{j} a_{ij} \cdot x_j = b_j$$

každá taková soustava je řešitelná, má-li aspoň jedno řešení. Říkáme, že soustava je jednoznačně řešitelná, má-li právě jedno řešení. Soustava je víceznačně řešitelná, má-li více než jedno řešení.

Soustava rovnic se dá vhodně zapsat v maticovém tvaru:

$$A \cdot x = B \tag{1}$$

kde A nazýváme maticí soustavy ve tvaru:

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

a B je matice pravých stran:

$$B = \begin{pmatrix} b_1 \\ b_2 \\ \mathbf{M} \\ b_n \end{pmatrix}$$

Je-li matice soustavy A regulární, soustava má právě jedno řešení, které můžeme určit pomocí Cramerova pravidla[1] (Cramerových vzorců).

2.1 Metody řešení

Numerické metody pro řešení SLAR se dělí na dvě třídy: přímé metody a iterační metody. Použitím přímých metod dostaneme přesné řešení problému po konečném počtu elementárních

[1] **Gabriel Cramer** (31. 7. 1704 – 4. 1. 1752) švýcarský matematik, přírodovědec a technik. V matematice se věnoval hlavně geometrii a teorii pravděpodobnosti. V roce 1750 vydal knihu o algebraických křivkách, kde je v dodatku uveden způsob vyloučení *n-1* neznámých ze soustavy *n* rovnic o *n* neznámých. Přes nevhodnou symboliku tím položil základy teorii determinantů.

operací, kdyby se ve výpočtech nevyskytovaly zaokrouhlovací chyby. Iterační metody hledají řešení soustavy jako limitu posloupnosti přibližných řešení.

2.1.1 Přímé metody

2.1.1.1 Cramerovo pravidlo

Při způsobu řešení soustavy Cramerovým pravidlem dostaneme přesné řešení, ale prakticky se metoda nedá dobře využít, protože je výpočetně velmi náročná díky vyčíslování determinantů. Metoda předpokládá regulárnost matice koeficientů, proto se nedá použít pro libovolnou soustavu rovnic. Princip metody je následující:

Určíme determinant *D* matice koeficientů $|A|$. Je-li tento determinant nenulový, soustava je jednoznačně řešitelná. Spočítáme determinanty matic D_i, kde matice D_i vznikne z matice *A* záměnou i-tého sloupce za sloupec pravých stran. Pro i-tou složku řešení x_i platí:

$$x_i = \frac{D_i}{D}$$

Příklad: Řešte Cramerovým pravidlem soustavu rovnic.

$$\begin{aligned} 2\cdot x + 3\cdot y - \quad z &= 5 \\ 3\cdot x - 2\cdot y + 2\cdot z &= 5 \\ 4\cdot x - \quad y + 3\cdot z &= 11 \end{aligned} \tag{2}$$

Pomocné výpočty:

$$D = \begin{vmatrix} 2 & 3 & -1 \\ 3 & -2 & 2 \\ 4 & -1 & 3 \end{vmatrix} \Rightarrow D = 2\cdot(-2)\cdot 3 + 3\cdot 2\cdot 4 + (-1)\cdot 3\cdot(-1) - (-1)\cdot(-2)\cdot 4 - 2\cdot(-1)\cdot 2 - 3\cdot 3\cdot 3 =$$

$$= -12 + 24 + 3 - 8 + 4 - 27 = -16$$

$$D_1 = \begin{vmatrix} 5 & 3 & -1 \\ 5 & -2 & 2 \\ 11 & -1 & 3 \end{vmatrix} \Rightarrow D_1 = -30 + 5 + 66 - 22 - 45 + 10 = -16$$

$$D_2 = \begin{vmatrix} 2 & 5 & -1 \\ 3 & 5 & 2 \\ 4 & 11 & 3 \end{vmatrix} \Rightarrow D_2 = 30 - 33 + 40 + 20 - 45 - 44 = -32$$

$$D_3 = \begin{vmatrix} 2 & 3 & 5 \\ 3 & -2 & 5 \\ 4 & -1 & 11 \end{vmatrix} \Rightarrow D_3 = -44 - 15 + 60 + 40 - 99 + 10 = -48$$

Výsledné řešení soustavy je:

$$\begin{aligned} x &= \frac{D_1}{D} = \frac{-16}{-16} = 1 \\ y &= \frac{D_2}{D} = \frac{-32}{-16} = 2 \\ z &= \frac{D_3}{D} = \frac{-48}{-16} = 3 \end{aligned} \tag{3}$$

2.1.1.2 Řešení s trojúhelníkovou maticí

Vhodnější metodou je upravování jednotlivých členů matice podle daného vzorce tzv. trojúhelníkovou maticí podle předpisu:

$$x_k = \frac{1}{u_{kk}} \left(b_k - \sum_{j=k+1}^{n} u_{kj} \cdot x_j \right), \tag{4}$$

kde u_{kk} jsou členy trojúhelníkové matice U a k, j jsou indexy členů.

K výpočtu libovolného x_k je potřeba nejvýše *n* vnitřních cyklů (1 sčítání + 1 násobení).

Příklad: Použitím metody řešení trojúhelníkovou maticí vypočítejte soustavu.

$$\begin{aligned} 3 \cdot x + 2 \cdot y + \quad z &= 10 \\ 2 \cdot x + 4 \cdot y + 5 \cdot z &= 25 \\ 3 \cdot x + 4 \cdot y + 8 \cdot z &= 35 \end{aligned} \tag{5}$$

Postup řešení:

1. první rovnici vynásobíme koeficientem $-\frac{2}{3}$ a přičteme k druhé rovnici, docílíme eliminace proměnné *x* v druhé rovnici, pak první rovnici vynásobíme koeficientem -1 a sečteme s třetí rovnicí, dostaneme soustavu:

$$\begin{aligned} 3 \cdot x \quad + 2 \cdot y \quad + \quad z &= 10 \\ \frac{8}{3} \cdot y \quad + \frac{13}{3} \cdot z &= \frac{55}{3} \\ 2 \cdot y \quad + 7 \cdot z &= 25 \end{aligned} \tag{6}$$

2. první dvě rovnice necháme nezměněny. Druhou rovnici násobíme koeficientem $-2\cdot\frac{3}{8}$ a sečteme se třetí rovnicí, kde docílíme eliminace proměnné *y*,

$$\begin{aligned} 3\cdot x &+2\cdot y &+ z &= 10 \\ &\frac{8}{3}\cdot y &+\frac{13}{3}\cdot z &= \frac{55}{3} \\ & & \frac{15}{4}\cdot z &= \frac{45}{4} \end{aligned} \tag{7}$$

3. z tohoto „trojúhelníkového tvaru" vypočteme *z*, následně *y* a nakonec *x*.

Řešení:

$$\begin{aligned} &z = 3 \\ &\frac{8}{3}\cdot y + \frac{13}{3}\cdot 3 = \frac{55}{3} \Rightarrow y = 2 \\ &3\cdot x + 2\cdot 2 + 3 = 10 \Rightarrow x = 1 \end{aligned} \tag{8}$$

2.1.1.3 Gaussova eliminace

Při použití této metody se upravují jednotlivé členy matice násobením multiplikátorem tak, aby výsledná matice měla všechny členy kromě hlavní diagonály nulové. Postupujeme následovně:

1. upravíme matici tak, aby se prvek $a_{11} \neq 0$, např. vhodným přehozením rovnic,
2. odečteme vhodný násobek prvního řádku od ostatních řádků tak, abychom dostali novou „ekvivalentní" matici s nulovými členy pod a_{11},
3. analogicky postupujeme dále, nulujeme sloupce pod prvky na hlavní diagonále, abychom dostali konečný tvar:

$$\left(\begin{array}{cccc|c} a_{11} & a_{12} & \mathbf{K} & a_{1n} & b_1 \\ 0 & a_{21} & \ldots & a_{2n} & b_2 \\ 0 & 0 & \ldots & \ldots & \ldots \\ 0 & 0 & 0 & a_{mn} & b_m \end{array}\right)$$

Hlavním nenulovým prvkům v diagonále se také říká pivoty.

Příklad: Pomocí metody Gaussovy eliminace vypočtěte soustavu:

$$\begin{aligned} x &+2\cdot y &+3\cdot z &+ w &= 2 \\ & y &+2\cdot z &+ w &= 1 \\ 2\cdot x &+\cdot y &+ z & &= 0 \\ 3\cdot x &+2\cdot y &+4\cdot z &+3\cdot w &= 1 \end{aligned} \tag{9}$$

Postup řešení:

$$\left|\begin{matrix}1&2&3&1&2\\0&1&2&1&1\\2&1&1&0&0\\3&2&4&3&1\end{matrix}\right| \left|\cdot(-2);\ \cdot(-3)\right| \left|\begin{matrix}1&2&3&1&2\\0&1&2&1&1\\0&-3&-5&-2&-4\\0&-4&-5&0&-5\end{matrix}\right| \left|\cdot 3;\ \cdot 4\right| \left|\begin{matrix}1&2&3&1&2\\0&1&2&1&1\\0&0&1&1&-1\\0&0&3&4&-1\end{matrix}\right| \left|\cdot(-3)\right| \approx$$

$$\approx \left|\begin{matrix}1&2&3&1&2\\0&1&2&1&1\\0&0&1&1&-1\\0&0&0&1&2\end{matrix}\right| \Rightarrow \begin{matrix}(4.\ krok) & x_1+2\cdot 5+3\cdot(-3)+1\cdot 2=2 \Rightarrow x_1=-1\\ (3.\ krok) & x_2+2\cdot(-3)+1\cdot 2=1 \Rightarrow x_2=5\\ (2.\ krok) & x_3+2\cdot 1=-1 \Rightarrow x_3=-3\\ (1.\ krok) & x_4=2\end{matrix}$$

Metoda Gaussovy eliminace je přímou metodou, ale není vždy použitelná pro obecné matice. Můžeme totiž dostat soustavu rovnic, která není řešitelná. To znamená, že může nastat případ, kdy upravováním řádků dostaneme rovnici odpovídající *i*-tému řádku soustavy ve tvaru:

$$0x_1+0x_2+\ldots+0x_n=c, \text{ kde } c\neq 0$$

a výsledná rovnice ani soustava nemá řešení.

2.1.1.4 Výběr hlavního prvku (pivoting)

Metoda je podobná Gaussově eliminaci, volíme úpravy matice vedoucí:

1. k úplnému výběru hlavního prvku – postup je pomalý a zdlouhavý,
2. k částečnému výběru hlavního prvku – upravování v daném sloupci nebo řádku pomocí porovnávání absolutních hodnot v původní matici.

Prezentovaná metoda je vhodná a použitelná pro většinu obecných matic. Stejně jako u Gaussovy eliminace může nastat nepřesnost v počítání v důsledku upravování hlavních prvků. Pro obecné matice s počtem prvků větší než padesát je nutná dvojitá přesnost.

2.1.1.5 Inverzní matice

Řešení pomocí inverzní matice lze použít pouze pro soustavy, jejichž matice je regulární. K matici A existuje inverzní matice A^{-1} taková, že platí: $A\cdot A^{-1}=E$, kde E je jednotková matice. Použití metody vyžaduje postup ve dvou krocích:

1. K matici soustavy vypočítáme inverzní matici. To můžeme udělat dvojím způsobem. Buď použijeme rozšířenou metodu Gaussovy eliminace, kdy za matici soustavy připojíme jednotkovou matici a upravujeme tak dlouho, dokud v levé části nedostaneme

jednotkovou matici. Pravá část je pak hledanou inverzní maticí. Nebo k výpočtu inverzní matice použijeme pravidlo (10) s výpočtem pomocí determinantů.

$$A^{-1}{}_{ij} = (-1)^{i+j} \cdot \frac{|A_{ji}|}{|A|} \tag{10}$$

kde $|A_{ji}|$ je determinant vzniklý z matice A vypuštěním i-tého řádku a j-tého sloupce, tzv. minoru.

2. Zleva násobíme soustavu inverzní maticí podle předpisu:

$$A \cdot x = B \qquad | \cdot A^{-1}$$

$$A^{-1} \cdot A \cdot x = A^{-1} \cdot B$$

$$x = A^{-1} \cdot B$$

Matice, ke které existuje matice inverzní, se nazývá regulární. V opačném případě mluvíme o matici singulární. Matice soustavy je regulární právě tehdy, když je čtvercová a její determinant je nenulový. Naopak u singulární matice platí, že má nulový determinant, tzn. že minimálně dva řádky jsou lineárně závislé. Tyto vlastnosti matic jsou pro nás při počítání soustav velice důležité, připomeneme si je později.

2.1.2 Iterační metody

Iterační metody jsou vhodné pro řešení řídkých matic, které vznikají například při řešení parciálních diferenciálních rovnic metodou sítí, protože jsou méně náročné na paměť počítače. Řídká soustava se vyznačuje velkým počtem nulových prvků. Ve srovnání s přímými metodami mohou iterační metody dopadnout příznivěji, pokud jde o celkový počet výpočtů.

Nevýhodou je, že i když vyjdeme z přesných hodnot a nedojde k zaokrouhlovací chybě, nemusíme dojít k výsledku. Příčinou je problém konvergence jednotlivých metod. Výpočty bývají ukončeny, jakmile je dosaženo řešení s předem danou požadovanou přesností.

2.1.2.1 Prostá iterace

Metoda prosté iterace se pro soustavy lineárních rovnic prakticky nepoužívá, protože je neefektivní. Ale na postupu si ukážeme, jak iterační metody přibližně pracují.

Danou lineární rovnici převedeme na tvar $y = (E - A) \cdot x + B$, kde E je jednotková matice. Prostá iterace je tedy dána vzorcem (11).

$$x^{(k+1)} = (E - A) \cdot x^k + B \tag{11}$$

Příklad: Pomocí metody prosté iterace vypočítejte řešení soustavy rovnic (12) ve čtvrtém kroku a srovnejte s řešením vypočteným klasickou metodou ($x_1 = 1,\ x_2 = 2,\ x_3 = -1$).

$$\begin{aligned} 10 \cdot x_1 + x_2 + x_3 &= 11 \\ x_1 + 10 \cdot x_2 + x_3 &= 20 \\ x_1 + x_2 + 10 \cdot x_3 &= -7 \end{aligned} \tag{12}$$

Postup řešení:

1. soustavu převedeme podle vztahu pro prostou iteraci (11) na tvar:

$$\begin{aligned} x_1 &= 1{,}1 - 0{,}1 \cdot x_2 - 0{,}1 \cdot x_3 \\ x_2 &= 2{,}0 - 0{,}1 \cdot x_1 - 0{,}1 \cdot x_3 \\ x_3 &= -0{,}7 - 0{,}1 \cdot x_1 - 0{,}1 \cdot x_2 \end{aligned}$$

2. do Tabulky 1 zapíšeme první zvolené aproximace x_i, obvykle volíme $x_i = 0$,
3. zvolené aproximace dosadíme do rovnic, výsledky zapíšeme do Tabulky 1 a považujeme jako vstup pro další iteraci,
4. blíží-li se výsledky k řešení, říkáme, že metoda konverguje; k ukončení výpočtu je potřeba zadat počet kroků iterací nebo požadovanou přesnost.

Výsledky vypočítané prostou iterací ve 4. kroku se ke známému řešení velmi blíží. Je ale nutné další iterování, abychom docílili přesného řešení.

počet kroků	x_1	x_2	x_3
1	0	0	0
2	1,1	2,0	-0,7
3	0,97	1,96	-1,01
4	1,005	2,004	-0,993
…	…	…	…

Tabulka 1: Kroky výpočtu prostou iterací

2.1.2.2 Jacobiho metoda

Předpokladem pro řešení soustavy $A \cdot x = B$ pomocí Jacobiho metody je matice *A*, jejíž diagonální prvky jsou nenulové, $a_{ii} \neq 0$. Pokud matice danou podmínku nesplňuje, musíme ji upravit do předepsaného tvaru pomocí úprav. Metoda používá pro výpočet hodnot x_i v *(k+1)*iteraci hodnot x_i, vypočítaných na předcházející iteraci, tj. nejdříve se celý vektor přibližného řešení změní a potom se použije v další iteraci. Jacobiho iterace pro výpočet řešení má tvar (13). Jacobiho metoda se také nazývá metodou současné opravy.

$$\mathbf{r}_{x^{(k+1)}} = \sum_{j=1}^{n} c_{ij} \cdot x_j^{(k)} + d_i \tag{13}$$

pro $i=1, 2, \ldots n$ a $c_{ij} = -\frac{a_{ij}}{a_{ii}}, d_{ij} = \frac{b_i}{a_{ii}}$.

Pro konvergenci metody Jacobiho iterace platí, že matice *C* musí být ostře diagonálně dominantní. Tzn. platí-li podmínka (14), pak Jacobiho metoda konverguje bez ohledu na volbu počáteční aproximace.

$$\sum_{j=1, j\neq i}^{n} \frac{|a_{ij}|}{|a_{ii}|} \tag{14}$$

tj. prvky na diagonále musí mít větší hodnotu než prvky mimo diagonálu.

2.1.2.3 Gauss-Seidlova metoda

Je podobná Jacobiho metodě, liší se jen tím, že při postupném výpočtu hodnot vektoru $x^{(k+1)}$ se tyto hodnoty používají hned při výpočtu již vypočítané složky vektoru. Při výpočtu nemusíme uchovávat v paměti všechny hodnoty vektoru. Iterační tvar u Gauss-Seidlovy metody má tvar:

$$\mathbf{r}_{x^{(k+1)}} = \sum_{j=1}^{i-1} c_{ij} \cdot x_j^{(k)} + \sum_{j=i+1}^{n} c_{ij} \cdot x_j^{(k)} + d_i . \tag{15}$$

Aby metoda konvergovala, musí platit, že matice popisující soustavu je diagonálně dominantní. Tedy platí podmínka (14) nebo je matice pozitivně definitní, tj. je symetrickou čtvercovou maticí, jejíž vlastní čísla jsou větší než nula. U této metody platí, že konverguje, i když při řešení Jacobiho metodou k výsledku nedojdeme.

2.1.2.4 Superrelaxační metoda

Superrelaxační metoda urychluje konvergenci Gauss-Seidlovy metody. Výpočet lze provést s ohledem na Gauss-Seidlovu metodu následující iterací:

$$x_i^{(k+1)} = x_i^{(k)} + w \cdot \Delta x_i^{(k)}, \tag{16}$$

kde $\Delta x_i^{(k)} = x_i^{(k+1)} - x_i^{(k)}$ je rozdíl iterací jednotlivých metod. Hlavní je relaxační faktor ω, který je z intervalu (0,2), nejčastěji <1,2> a slouží k urychlení konvergence metody.

Gauss-Seidlova metoda je speciálním případem superrelaxační metody, kdy relaxační faktor je roven číslu 1.

3 Soustava diferenciálních rovnic

Diferenciální rovnice (dále jen DR) je matematická rovnice, ve které jsou proměnnými derivace funkcí. Základními typy rovnic jsou:

1. obecná DR, která obsahuje derivace funkce podle jedné proměnné,
2. parciální DR, jejíž proměnnými jsou funkce derivované podle více proměnných.

Tyto rovnice můžeme dále dělit podle řádu derivace na DR prvního řádu a DR vyšších řádů, přičemž za řád rovnice považujeme řád nejvyšší derivace, která je v rovnici obsažena. Máme-li *m* DR s *n* neznámými, pracujeme se soustavou diferenciálních rovnic (dále jen SDR).

Nejvýhodnější a jediný způsob, jak pracovat se SDR v souvislosti se SLAR, je používat SDR 1. řádu. Obecnou DR 1. řádu lze obecně zapsat takto:

$$y' + a(x) \cdot y = b(x) \tag{17}$$

Setkáme-li se s obyčejnou diferenciální rovnicí *n*-tého řádu ve tvaru:

$$f\left(x, y, y', y'', y''', ..., y^{(n)}\right) = g(x)$$

lze ji převést na SDR 1. řádu s použitím substitucí $y' = z_1$, $y'' = z_2$, …, $y^{(n-1)} = z_{n-1}$ nebo tzv. metody snižování řádu derivace.

Výsledná soustava bude ve tvaru:

$$\begin{aligned} y' &= z_1 \\ z_1' &= z_2 \\ z_2' &= z_3 \\ &... \\ z_{n=1}' &= \tilde{g}\left(x, y, z_1, z_2, ..., z_{n=1}\right) \end{aligned} \tag{18}$$

Abychom dostali jednoznačné řešení soustavy, musíme na počátku úlohy zadat každé rovnici podmínku. Podle zadání podmínek rozlišujeme dva typy úloh:

1. počáteční problém – všechny podmínky rovnic jsou zadány v jednom bodě, tzn. řešení lze sledovat z jednoho vycházejícího bodu,
2. okrajový problém – podmínky nejsou zadány z jednoho bodu, bývají určeny ve dvou krajních bodech (např. intervalu řešení), ale mohou být zadány i jinak.

Principem řešení DR je diskretizace proměnných. Znamená to, že přibližné řešení nekonstruujeme jako spojitou funkci, ale vygenerujeme body x_0, x_1, x_2,... a určujeme čísla y_0, y_1, y_2,..., která aproximují $y(x_0)$, $y(x_1)$, $y(x_2)$, ... Z toho vyplývá, že analytické

řešení SDR je ve velkém množství případů příliš obtížné na výpočet, než abychom dostali přesné řešení. K řešení používáme raději numerické řešení, kterým získáme přibližné výsledky.

3.1 Metody numerického řešení

3.1.1 Eulerova metoda

Přibližnou hodnotu řešení $y(x_{i+1})$ DR nalezneme pomocí dvou členů Taylorova rozvoje 1. stupně. Výpočet je dán rekurentním vztahem (19) Eulerovy metody a její lokální chybou.

$$y(x_{i+1}) = y(x_i) + h_i \cdot y'(x_i) + \frac{1}{2} h_i^2 \cdot y''(c) \tag{19}$$

kde $c = (x_i, x_{i+1})$.

Odtud vidíme, že je-li $y'(x_i) = f(x_i, y(x_i))$, dostáváme rekurentní vztah:

$$y_{i+1} = y_i + h_i \cdot f(x_i, y_i). \tag{20}$$

a lokální chyba je dána

$$d_i = \frac{1}{2} h_i^2 (c). \tag{21}$$

Příklad: Eulerovou metodou s konstantními kroky $h_1 = 0{,}1$ a $h_2 = 0{,}2$ řešte rovnici.

$$y' = x - y, \quad y(0) = 1, \quad x \in \langle 0;\ 0, \rangle \tag{22}$$

Řešení: Použijeme rekurentního vzorce (20) a postupnými výpočty dostaneme řešení v Tabulce 2, kde první řádek je dán počátečními podmínkami. Počáteční podmínka y_0 je zadána, x_0 vhodně zvolíme a dále počítáme dle zadaného kroku. Chybu známého řešení oproti řešení počítané touto metodou vyjádříme: $e_n = y_{n\_známé} - y_{n\_vypoč}$.

Pomocné výpočty pomocí rekurentního vzorce vypadají následovně:

$y_1 = y_0 + h_1 \cdot (x_0 - y_0) = 1 + 0{,}1 \cdot (0 - 1) = 1 - 0{,}1 = 0{,}9$

$y_2 = y_1 + h_1 \cdot (x_1 - y_1) = 0{,}9 + 0{,}1 \cdot (0{,}1 - 0{,}9) = 0{,}9 - 0{,}08 = 0{,}82$

atd.

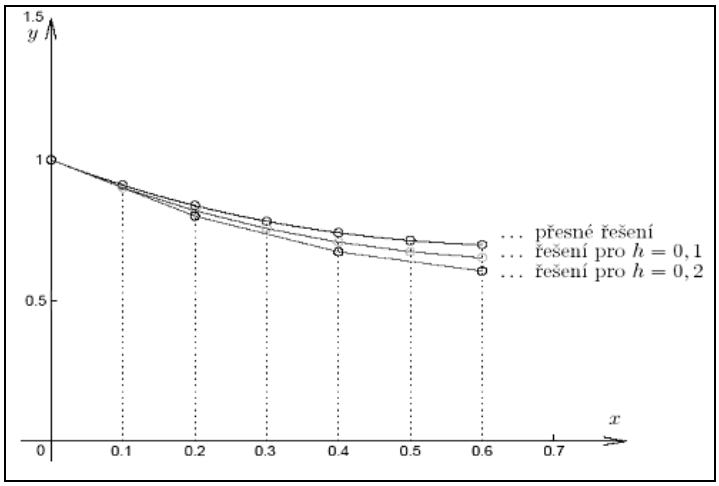
$y_1 = y_0 + h_2 \cdot (x_0 - y_0) = 1 + 0{,}2 \cdot (0 - 1) = 1 - 0{,}2 = 0{,}8$

$y_2 = y_1 + h_1 \cdot (x_1 - y_1) = 0{,}8 + 0{,}2 \cdot (0{,}2 - 0{,}8) = 0{,}8 - 0{,}12 = 0{,}68$

atd.

x_n	$y(x_n)$ známé řešení pro porovnání	$h_1 = 0{,}1$		$h_2 = 0{,}2$	
		y_n	e_n	y_n	e_n
0	1,000	1,000	0,000	1,000	0,000
0,1	0,910	0,900	0,010		
0,2	0,837	0,820	0,017	0,800	0,037
0,3	0,782	0,758	0,024		
0,4	0,741	0,712	0,029	0,680	0,061
0,5	0,713	0,681	0,032		
0,6	0,698	0,663	0,035	0,624	0,074

Tabulka 2: Výsledky příkladu řešeného Eulerovou metodou



Obrázek 1: Grafické znázornění výsledků z Tabulky 2

Eulerovu metodu řadíme do metod 1. řádu. Je málo přesná kvůli chybě, která v jednom kroku úměrně stoupá s druhou mocninou. Postup řešení vyžaduje malý krok, protože při velkém kroku rychle stoupá chyba metody. Eulerova metoda se v praxi velice málo využívá.

3.1.2 Metoda Taylorova typu

Metoda je založena na aproximování $y(x_{n+1})$ pomocí Taylorova rozvoje vyššího řádu. Předchozí prezentovaná Eulerova metoda je vlastně metodou Taylorova typu 1. řádu. Vztah pro vyšší řády vypadá:

$$y(x_{n+1}) = y(x_n + h_n) = y(x_n) + h_n \cdot y'(x_n) + \frac{1}{2} h_n^2 \cdot y''(x_n) + \ldots + \frac{1}{p!} h_n^p \cdot y^{(p)}(x_n). \qquad (23)$$

Obvykle je h_n konstantní, takže píšeme jen *h*. Obecně lze určit rekurentní vztah pro $n \geq 1$:

$$y^{(n+1)} = f^n(x, y(x)) = f_x^{(n-1)}(x, y(x)) + f_y^{(n-1)}(x, y(x)) \cdot f(x, y(x)). \qquad (24)$$

A jednotlivé derivace y lze určit postupným derivování funkce *f* podle: $f'(x, y) = f_x' + f_y' \cdot f$.

Použití metody pro výpočet si ukážeme na příkladě. Odvoďte metodu Taylorova typu 2. řádu pro řešení úlohy: $y' = x - y$, $y(0) = 1$, $x \in \langle 0;0,6 \rangle$ s konstantními krokem $h = 0,2$.

Řešení: přípravné výpočty

$$f(x, y) = x - y$$
$$f'(x, y) = f_x' + f_y' \cdot f(x, y) = 1 + (-1) \cdot (x - y) = 1 - x + y$$

Pomocí upraveného rekurentního vztahu (23) pro první derivaci počítáme

$$y_1 = y_0 + h \cdot (x_0 - y_0) + \frac{1}{2} h^2 \cdot (1 - x_0 + y_0) = 1 - 0,2 + \frac{1}{2} \cdot 0,2^2 \cdot 2 = 1 - 0,2 + 0,04 = 0,84$$

atd.

x_n	y_n	e_n	$y(x_n)$ **123** *známé řešení pro porovnání*
0	1,000	0,000	1,000
0,2	**0,840**	-0,003	0,837
0,4	0,745	-0,004	0,741
0,6	0,703	-0,005	0,0698

Tabulka 3: Výsledky řešení metodou Taylorova typu

Vidíme, že metoda je mnohem přesnější než řešení Eulerovou metodou s menším krokem *h=0,1*. Ale ani metody založené na Taylorově rozvoji se v praxi nepoužívají, protože u složitějších rovnic je potřeba počítat s vyššími derivacemi a to je časově náročné.

3.1.3 Metody Runge-Kuttova typu

Uvedením metod založených na postupném zpřesňování hodnot x_n a x_{n+1} se seznámíme s principem, který je pro počítání diferenciálních rovnic v praxi běžně používán. Vychází také z Taylorova polynomu, ale nepoužívá jej přímo, aby nebylo nutné počítat derivace a jejich hodnoty. Hledané řešení je pak kombinací několika hodnot funkce *f* vypočítaných ve strategicky zvolených bodech v daném intervalu. Metod založených na tomto principu je velké množství.

Výpočetní vzorec je:

$$y_{n+1} = y_n + h \cdot q_{RK}(x_n, y_n, h) \tag{25}$$

kde platí

$$q_{RK}(x_n, y_n, h) = a_{r1} \cdot k_1(h) + a_{r2} \cdot k_2(h) + \ldots + a_{rr} \cdot k_r(h) \tag{26}$$

zároveň a_{rr} jsou konstanty a pro konstanty $k_1,\ k_2, \mathbf{K},\ k_r$ platí:

$$\begin{aligned} k_1(h) &= f(x_n, y_n) \\ k_2(h) &= f(x_n + a_2 \cdot h \cdot y_n + h \cdot b_{21} k_1) \\ &\ldots \\ k_r(h) &= f\left[x_n + a_{2r} \cdot h \cdot y_n + h \cdot \left(b_{r1} k_1 + b_{r2} k_2 + \ldots + b_{r,r-1} k_{r-1}\right)\right] \end{aligned} \tag{27}$$

Pokud zvolíme např. *r=1* dostaneme vzorec pro Eulerovu metodu. Pro rovnice vyšších řádů např. pro metodu řádu 5 volíme *r=6* apod.

Nejpoužívanějším typem je metoda Runge-Kutta čtvrtého řádu. K výpočtu jednoho kroku se používají vzorce:

$$\begin{aligned} k_1(h) &= f(x_1, y_1) \\ k_2(h) &= f\left(x_2 + \frac{h}{2}, y_2 + \frac{h}{2} \cdot k_1\right) \\ k_3(h) &= f\left(x_3 + \frac{h}{2}, y_3 + \frac{h}{2} \cdot k_2\right) \\ k_4(h) &= f\left(x_4 + \frac{h}{2}, y_4 + h \cdot k_3\right) \\ y_{n+1} &= y_n + \frac{h}{6}(k_1 + 2 \cdot k_2 + 2 \cdot k_3 + k_4) \end{aligned} \tag{28}$$

Chyba jednoho kroku je $e \approx h^5$, chyba metody se v zadaném intervalu zvětšuje lineárně.

3.1.4 Vícekrokové metody

Všechny předchozí metody byly jednokrokové, neboť k výpočtu y_{i+1} bylo třeba znát jen řešení y_i z předchozího kroku výpočtu. Pro vícekrokové metody platí, že si musíme uchovávat historii kroků a pro výpočet y_{i+1} používat více předchozích hodnot. Čím větší je počet předchozích používaných kroků, tím větší je přesnost aktuálního řešení. Vícekrokových metod je nepřeberné množství, nejčastěji uváděné jsou:

1. metoda numerického derivování,
2. metoda numerické integrace,

3. Adamsovy-Bashforthovy metody,
4. Adamsovy-Moultonovy metody aj.

Obecná lineární *k*-kroková metoda požaduje pro vyřešení zadaného problému tzv. startovací úsek. Tj. máme-li funkci

$$y_{i+1} = \sum_{j=1}^{k} a_j \cdot y_{i+1-j} + h \cdot \sum_{j=0}^{k} b_j \cdot f_{i+1-j} \tag{29}$$

musíme znát hodnoty $y_0, y_1, ..., y_{k-1}$, resp. $f_0, f_1, ..., f_{k-1}$. Tyto hodnoty startovacího úseku můžeme získat vhodnou jednokrokovou metodou. Také musí platit, že k je přirozené číslo a alespoň jedna z konstant a_k, b_k je různá od nuly. Je-li například $b_0 = 0$ říkáme, že metoda je explicitní a řešení počítáme přímo dle vzorce (29). Je-li $b_0 \neq 0$, pak je metoda implicitní a v každém kroku musíme počítat rovnici

$$y_{i+1} = h \cdot b_0 \cdot f(x_{i+1}, y_{i+1}) + g\,. \tag{30}$$

3.2 Problémy při řešení

3.2.1 Lokální a globální chyba

Při numerickém řešení soustavy ovlivňují výsledek kromě zaokrouhlovacích chyb také chyby způsobené diskretizací úlohy. Chybu, které se dopouštíme diskretizací úlohy v jednom kroku, nazýváme lokální diskretizační chyba. Je-li *z(x)* řešení úlohy $y' = f(x, y(x))$ s počáteční podmínkou $y(x_{i=1}) = y_{i=1}$, pak pro lokální diskretizační chybu platí vztah: $d_i = z(x_i) - y_i$. Globální (akumulovaná) diskretizační chyba je výslednicí diskretizačních chyb ve všech předchozích krocích. Platí-li, že x_i je řešení úlohy $y(x_i)$, pak platí: $e_i = y(x_i) - y_i$. Obě uvedené chyby jsou chybami jednokrokových metod, nezahrnují chyby při zaokrouhlování.

Globální chybu k-krokových metod ovlivňují faktory, které se u jednokrokových metod neprojevují. Jsou to chyby stanovení počátečních hodnot $y_0, y_1, ..., y_{k-1}$. U implicitních metod se může projevit chyba řešení rovnice, tj. řešíme-li diferenciální rovnici náhradou diferenční rovnicí, pak má tato rovnice kromě aproximujícího teoretického řešení rovnice parazitní složky, které způsobují poruchové členy v numerickém řešení. Pokud je vliv parazitních složek malý a v limitě pro krok blížící se k nule vymizí, pak říkáme, že *k*-kroková metoda je stabilní při $h \to 0$. Tato stabilita určuje stabilní šíření lokální chyby při $h \to 0$.

4 Převod SLAR na SDR

Jedním z důvodů, proč jsou SDR používány pro řešení SLAR, je zaručení stability. V následující kapitole si ukážeme možnosti, jak SLAR řešit, abychom získali stabilní řešení. S využitím programu TKSL představíme dvě metody. První postup řešení využívá transformačního algoritmu a pro SLAR je vždy stabilní. Druhou z metod nazveme tzv. elementárním převodem, protože je založena na elementárních úpravách soustavy.

Pro ustálený stav platí, že dvě po sobě jdoucí hodnoty numerického řešení jsou shodná. Nastavíme-li $x' = 0$ a $y' = 0$, hodnoty *y* a *x* se nastaví na hledané řešení v ustáleném stavu. Jak uvidíme dále, stabilní stav přináší mnoho problémů. Některé soustavy stabilní být nemusí nebo se do ustáleného stavu dostanou až po dlouhém časovém intervalu a další.

Zda-li je soustava stabilní, lze rychle určit podle hodnot vlastních čísel. Vlastní čísla získáme vypočtením kořenů charakteristické rovnice. Charakteristickou rovnici (31) sestavíme, když determinant vzniklý vynásobením matice soustavy *A* s jednotkovou maticí *E* s konstantou λ, položíme roven nule a vyřešíme.

$$\left|A - \lambda \cdot E\right| = 0 \tag{31}$$

Platí-li pro kořeny této charakteristické rovnice (tj. pro vlastní čísla) podmínka (32)

$$\mathrm{Re}\left|\lambda\right| > 0 \tag{32}$$

pak dostaneme stabilní řešení SLAR v ustáleném stavu.

Obecně platí, že vytvoříme-li z levých stran soustavy symetrickou čtvercovou matici a jsou-li vlastní čísla této matice kladná, pak říkáme, že matice je pozitivně definitní a soustava je stabilní. Pak také platí, že nezáleží na metodě, kterou použijeme pro výpočet stabilního řešení. Postup výpočtu hodnot vlastních čísel si ukážeme na příkladě v následující podkapitole.

Výše popsaný lze použít pro symetrické matice typu $n \times n$, kde $n \leq 5$, protože výpočty jsou jednoduché. Pro větší SLAR nebo soustavy s nesymetrickými maticemi se používají složitější numerické metody pro výpočet vlastních čísel, jako např. Jacobiho metoda nebo metoda QR.

Vlastní čísla souvisí s analytickým výpočtem. Máme-li počítat s lineárními DR s konstantními koeficienty pomocí charakteristické rovnice a jejích vlastních čísel, vypočítáme analytické řešení. V obecném má řešení tvar:

$$y = c_1 \cdot e^{\lambda_1 \cdot x} + c_2 \cdot e^{\lambda_2 \cdot x} + \mathbf{K} + c_n \cdot e^{\lambda_n \cdot x} \tag{33}$$

kde $c_1, c_2, \mathbf{K}, c_n$ jsou reálné koeficienty závislé na počátečních podmínkách. Analytické řešení můžeme zkoumat z hlediska zjištění stability. Jednotlivé složky řešení jsou tvořeny přirozenými exponenciálními funkcemi (základem je Ludolfovo číslo), které lze v grafu zobrazit pomocí exponenciálních křivek. Se znalostí souvislosti exponenciální a logaritmické funkce výsledek zobrazíme tak, abychom jednoduše určili stabilitu soustavy. Definičním oborem exponenciální funkce jsou reálná čísla a obor hodnot je definován jako interval $\langle 0, \infty \rangle$. Logaritmická funkce je k exponenciální inverzní funkcí, takže definiční obor exponenciální funkce je oborem hodnot funkce logaritmické a naopak. V důsledku této vlastnosti lze zobrazit i záporné číslo na ose *y* a vykreslením grafu logaritmické funkce zjistíme, zda je řešení ustálené. Kreslení výsledků používá také program TKSL. Zobrazení výsledku je důležité i pro sledování konvergence a její rychlosti. Dané vlastnosti si v praktickém použití připomeneme později.

4.1 Metoda převodu transformací

Řešte rovnici:

$$\begin{aligned} x + 2 \cdot y &= -1 \\ x + 3 \cdot y &= 4 \end{aligned} \tag{34}$$

Vypočteme matici soustavy:

$$A = \begin{pmatrix} 1 & 2 \\ 1 & 3 \end{pmatrix} \tag{35}$$

Matici upravíme podle vzorce (31):

$$\begin{vmatrix} 1-\lambda & 2 \\ 1 & 3-\lambda \end{vmatrix} = 0 \tag{36}$$

a vyčíslením determinantu $(1-\lambda)\cdot(3-\lambda) - 2 \cdot 1 = 0$ dostaneme charakteristickou rovnici:

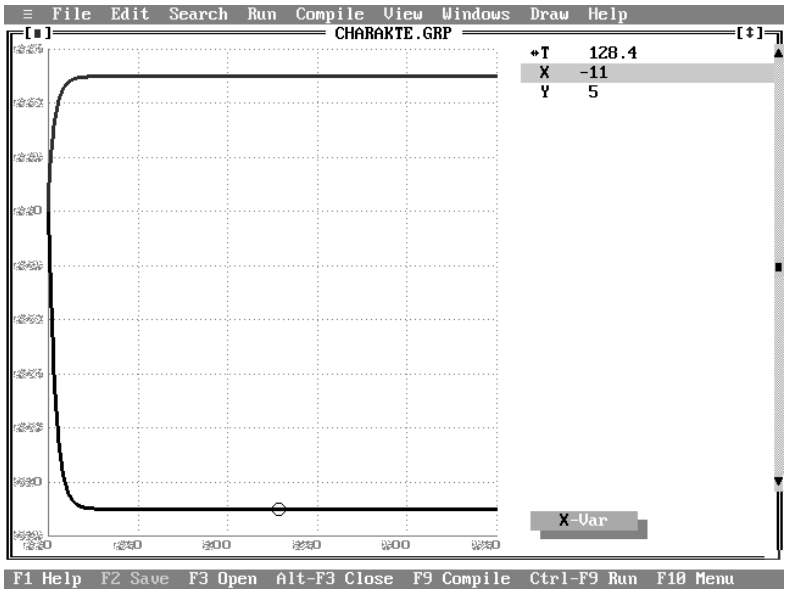
$$\lambda^2 - 4 \cdot \lambda + 1 = 0 \tag{37}$$

ze které vypočítáme kořeny $\lambda_{1,2} = \begin{pmatrix} 0{,}26795 \\ 3{,}73205 \end{pmatrix}$.

Vidíme, že podmínka (32) je splněna, protože oba kořeny charakteristické rovnice jsou kladné, matice je pozitivně definitní a řešení počítané soustavy (34) je v ustáleném stavu. Řešení této soustavy můžeme vypočítat jak elementárními úpravami, tak transformačním převodem.

Pro potvrzení je v Obrázku 2 zobrazen stabilní průběh počítané soustavy pomocí elementárního převodu, který si ukážeme později. Logaritmické křivky zobrazují výsledky řešení

a hodnoty odpovídají výsledkům řešené soustavy (34). Stabilitu lze z grafu vyčíst tak, že křivka se v určitém časovém okamžiku nemění, ustálí se na výsledné hodnotě. Důležité je uvědomit si, že výsledky *x*, *y* soustavy rovnic se neshodují s hodnotami kořenů λ_1, λ_2 charakteristické rovnice. Hodnoty kořenů používáme jen pro detekci stabilního stavu a případně pro výpočet analytického řešení.



Obrázek 2: Stabilní řešení soustavy (34)

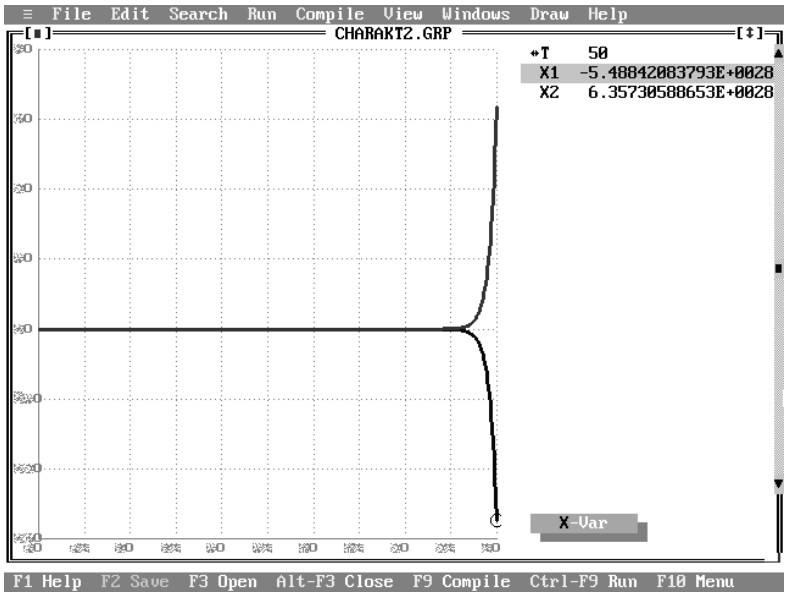
Pokud se setkáme s příkladem, jehož řešení nebude stabilní, budeme muset postupovat jinak. Ukážeme si soustavu, která podmínku (32) nesplňuje:

$$\begin{aligned} x_1 - 2 \cdot x_2 &= -1 \\ -5 \cdot x_1 + 3 \cdot x_2 &= 4 \end{aligned} \tag{38}$$

Soustavu (38) upravíme podobným postupem jako v předchozím příkladě, získáme charakteristickou rovnici:

$$\begin{vmatrix} 1-\lambda & -2 \\ -5 & 3-\lambda \end{vmatrix} = (1-\lambda)\cdot(3-\lambda) - (-2)\cdot(-5) = 0 \text{, tj. } \lambda^2 - 4\cdot\lambda - 7 = 0 ,$$

ze které po vypočítání dostaneme výsledky $\lambda_{1,2} = \begin{pmatrix} -1{,}3166 \\ 5{,}3166 \end{pmatrix}$. Jedno z vlastních čísel je záporné, matice není pozitivně definitní. Pro ověření zobrazíme v Obrázku 3 výsledky, které dostaneme řešením v programu TKSL, a vidíme, že jsou nestabilní. Zvyšováním intervalu výpočtu bychom dostávali narůstající hodnoty.



Obrázek 3: Nestabilní řešení soustavy (38)

Chceme-li zaručit stabilitu řešení jakékoli soustavy, musíme zaručit splnění podmínky (32). Jedním ze způsobů, jak soustavu s nestabilním průběhem upravit na soustavu se stabilním průběhem, je použití transponované matice a již avizované metody transformačního algoritmu.

Algoritmus pro převod SLAR na SDR obsahuje tyto kroky:

1. pro SLAR vytvoříme matici soustavy A a matici pravých stran *B*,
2. vypočteme transponovanou matici k matici soustavy A^T,
3. podle vzorce (39) získáme SDR.

$$A^T \cdot A \cdot x - A^T \cdot B = -x' \tag{39}$$

Je-li matice A regulární, což je obecná podmínka řešitelnosti jakékoli SLAR, výsledná matice $A^T \cdot A$ je vždy pozitivně definitní. Pak také platí, že charakteristická rovnice je dána

$$\left| A^T \cdot A - \lambda \cdot E \right| = 0 \tag{40}$$

a její vlastní čísla jsou kladná. Díky vlastnosti pozitivně definitní matice dostáváme vždy stabilní řešení.

Vrátíme se k příkladu dané soustavou (38), která není v elementárním tvaru stabilní, a ukážeme postup transformace s ověřením podmínky pozitivně definitní matice. Pro matici soustavy určíme matici transponovanou:

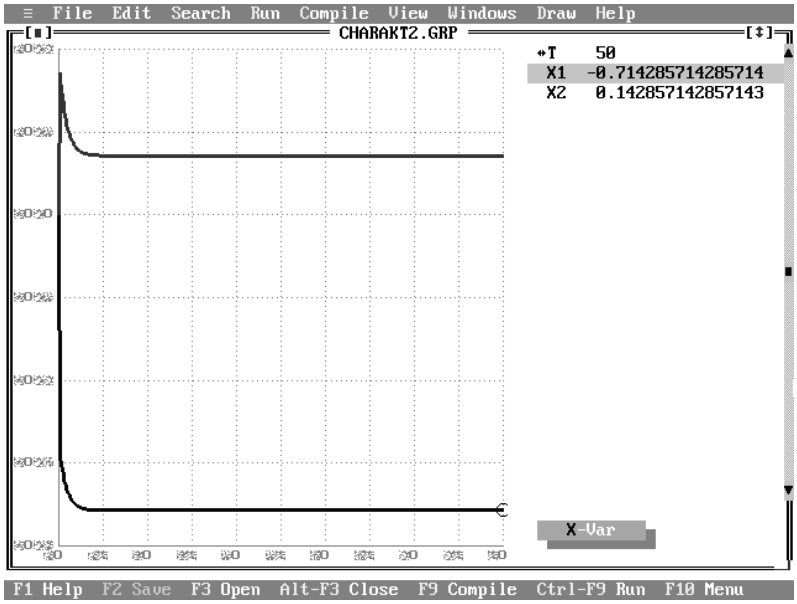
$$A = \begin{vmatrix} 1 & -2 \\ -5 & 3 \end{vmatrix},\ A^T = \begin{vmatrix} 1 & -5 \\ -2 & 3 \end{vmatrix} \tag{41}$$

Součin matic je roven: $A^T \cdot A = \begin{vmatrix} 1 & -5 \\ -2 & 3 \end{vmatrix} \cdot \begin{vmatrix} 1 & -2 \\ -5 & 3 \end{vmatrix} = \begin{vmatrix} 26 & -17 \\ -17 & 13 \end{vmatrix}$

a vypočteme charakteristickou rovnici:

$$\begin{vmatrix} 26-\lambda & -17 \\ -17 & 13-\lambda \end{vmatrix} = 0 \text{, tj. } \lambda^2 - 39 \cdot \lambda + 49 = 0 \tag{42}$$

Kořeny charakteristické rovnice (41) jsou $\lambda_{1,2} = \begin{pmatrix} 1{,}25 \\ 37{,}75 \end{pmatrix}$ kladné a výsledný průběh řešení je stabilní (Obrázek 4).



Obrázek 4: Stabilní řešení soustavy (38) použitím transformace

Zvláštností je, že některá literatura uvádí k teorii vlastních čísel a charakteristické rovnici poněkud jiné zpracování a jiný způsob řešení, který je ale ve výsledku shodný s postupem uvedeným výše. Determinant lze totiž napsat ve tvaru:

$$\left| A + \lambda \cdot E \right| = 0 \tag{43}$$

a pro stabilní soustavu vyjdou vlastní čísla záporná. V uvedených postupech je tato jediná „znaménková“ odchylka a záleží na počtáři, kterou z metod pro zjišťování stability zvolí. První z postupů koresponduje s transformačním algoritmem (shoda znamének) a vlastnost o pozitivně definitní matici.

Následující kapitola popíše postup, jak využít program pro zápis soustav do TKSL. Hlavní výhoda programu spočívá v rychlém zapsání soustav do kódu. U počítání pomocí transformačního algoritmu je velkou úsporou času to, že program si transformaci, resp. součin matice soustavy a transponované matice provádí sám, nemusíme nic počítat, také odpadá zjišťování vlastních čísel, protože transformační algoritmus vždy zajistí stabilní výsledek.

4.2 Zdrojové texty programu TKSL

Pro zapsání SDR pomocí transformačního převodu do zdrojového kódu pro program TKSL postupujeme jednodušeji než jak je popsána teorie k transformačnímu algoritmu v kapitole 4.1. Uvedli jsme, že program TKSL si součin matic počítá sám, takže mu zadáme jen soustavu SLAR, kterou chceme vypočítat, a předpis pro transformovanou matici v DR. Opět nesmíme zapomenout na hodnoty počáteční podmínek pro každou DR.

Pro obecnou soustavu je postup pro zápis rovnic do kódu následující. Zadaná SLAR, kterou chceme vypočítat, má tvar:

$$\begin{aligned} a_{11} \cdot x + a_{12} \cdot y + a_{13} \cdot z &= b_1 \\ a_{21} \cdot x + a_{22} \cdot y + a_{23} \cdot z &= b_2 \\ a_{31} \cdot x + a_{32} \cdot y + a_{33} \cdot z &= b_3 \end{aligned} \tag{44}$$

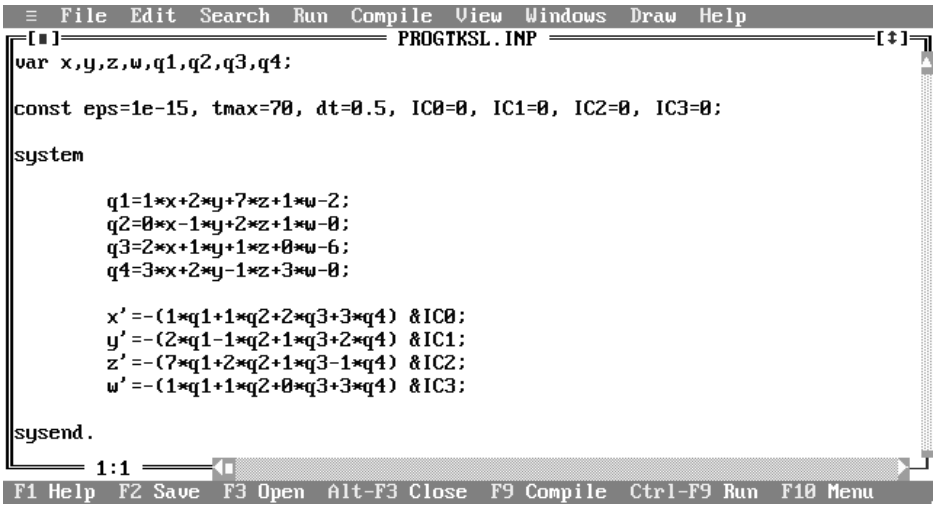
Pravé strany rovnic převedeme na levé a na pravou stranu dosadíme parametry q_i takto:

$$\begin{aligned} a_{11} \cdot x + a_{12} \cdot y + a_{13} \cdot z - b_1 &= q_1 \\ a_{21} \cdot x + a_{22} \cdot y + a_{23} \cdot z - b_2 &= q_2 \\ a_{31} \cdot x + a_{32} \cdot y + a_{33} \cdot z - b_3 &= q_3 \end{aligned} \tag{45}$$

Dostaneme první část zápisu kódu. Druhou část vytvoříme tak, že z matice soustavy vytvoříme transponovanou matici (jednoduchou záměnou $a_{ij} = a^T{}_{ji}$) a nahradíme neznámé parametry: $x = q_1$, $y = q_2$, $z = q_3$,… Na pravé strany dosadíme záporné proměnné v diferenciálním tvaru:

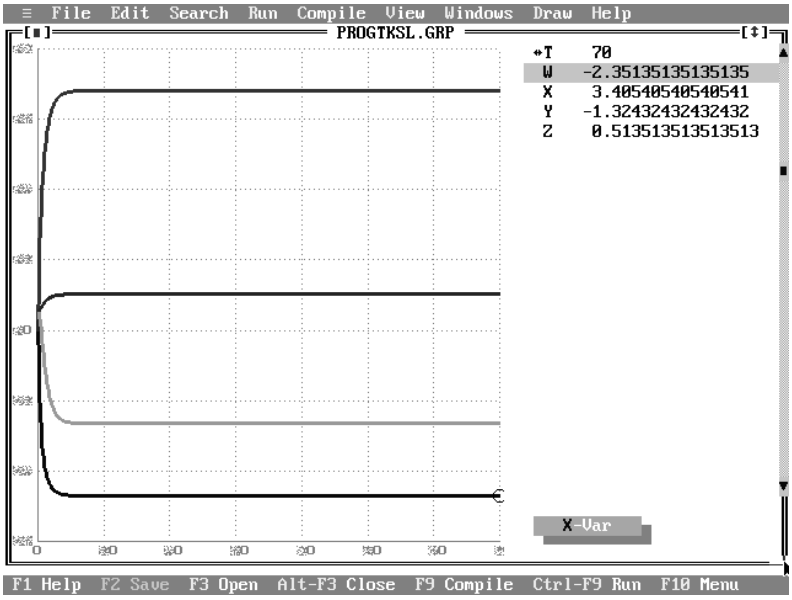
$$\begin{aligned} a_{11} \cdot q_1 + a_{21} \cdot q_2 + a_{31} \cdot q_3 &= -x' \\ a_{12} \cdot q_1 + a_{22} \cdot q_2 + a_{32} \cdot q_3 &= -y' \\ a_{13} \cdot q_1 + a_{23} \cdot q_2 + a_{33} \cdot q_3 &= -z' \end{aligned} \tag{46}$$

Elegantní řešení s parametry na levých stranách docílíme základními úpravami rovnic (převod z jedné strany na druhou, násobení číslem -1). Získanou soustavu rovnic zapíšeme programu, jehož kód bude ve tvaru znázorněném na Obrázku 5.



Obrázek 5: Zdrojový kód pro SLAR (44) pomocí transformačního algoritmu

Spuštěním výpočtu dostaneme stabilní řešení na Obrázku 6.

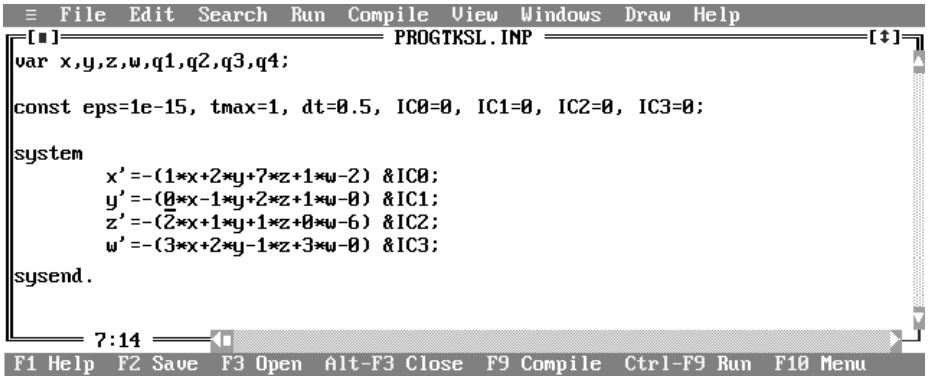


Obrázek 6: Stabilní řešení soustavy (44) dle Obrázku 5

Pro přepis elementárním převodem bude mít kód pro program podobu dle Obrázku 7. Postup převodu pomocí elementárních úprav si ukážeme v kapitole 4.3. Do programu zapíšeme tytéž rovnice, které převodem získáme, a do programu musíme zadat počáteční podmínku pro každou DR.

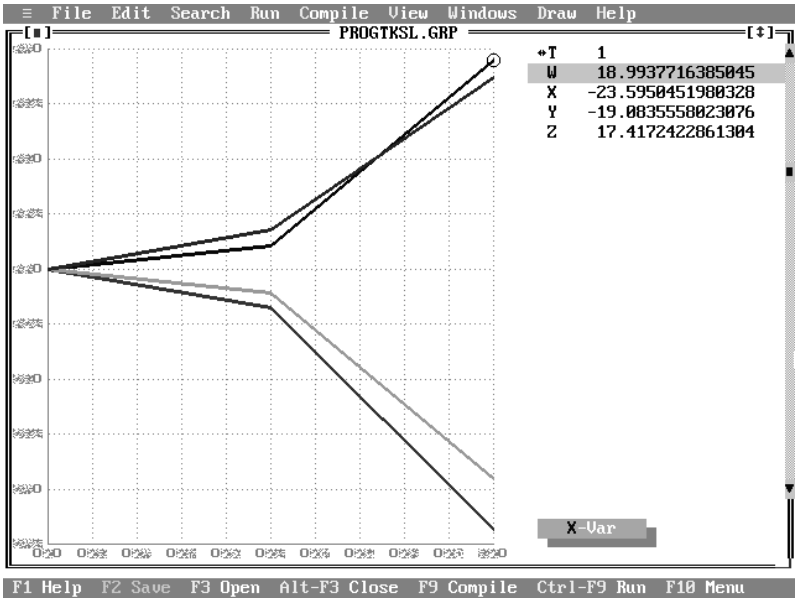
Máme řešit SLAR ve tvaru:

$$\begin{aligned} x &+ 2\cdot y &+ 7\cdot z &+ w &= 2 \\ &- y &+ 2\cdot z &+ w &= 0 \\ 2\cdot x &+ y &+ z & &= 6 \\ 3\cdot x &+ 2\cdot y & - z &+ 3\cdot w &= 0 \end{aligned} \tag{47}$$



Obrázek 7: Zdrojový kód SLAR (47) s elementárním převodem na SDR

Spuštěním výpočtu dostaneme nestabilním řešení, které je zobrazeno na Obrázku 8.



Obrázek 8: Nestabilní řešení soustavy (47) dle Obrázku 7

4.3 Elementární převod

Pokud řešení není stabilní, musíme použít transformační algoritmus. Zjistíme-li však ověřením hodnot vlastních čísel soustavy, že řešení je stabilní, použijeme elementární převod bez rozpaků. Existuje způsob, jak u některých soustav rovnic dosáhnout aplikováním metody elementárního převodu k získání stabilního řešení podobně jako při použití transformačního algoritmu. Je nutno dodržovat následující pravidla.

Způsob upravení SLAR si ukážeme na příkladě. Řešte soustavu:

$$\begin{aligned} x - y &= 3 \\ x + y &= 1 \end{aligned} \tag{48}$$

Libovolnou klasickou metodou lze určit řešení: $x = 2,\ y = -1.$

Pro elementární převod upravíme soustavu (45) následovně. V prvním kroku anulujeme:

$$\begin{aligned} 0 &= -(x - y - 3) \\ 0 &= -(x + y - 1) \end{aligned} \tag{49}$$

a přepíšeme do tvaru SDR:

$$\begin{aligned} x' &= -(x - y - 3) \\ y' &= -(x + y - 1) \end{aligned} \tag{50}$$

Pro počáteční podmínky např. $x(0) = 0$ a $y(0) = 0$ je řešení SDR (50) pomocí programu TKSL na Obrázku 9. Důležité je, že řešení soustavy (48) má stabilní průběh.

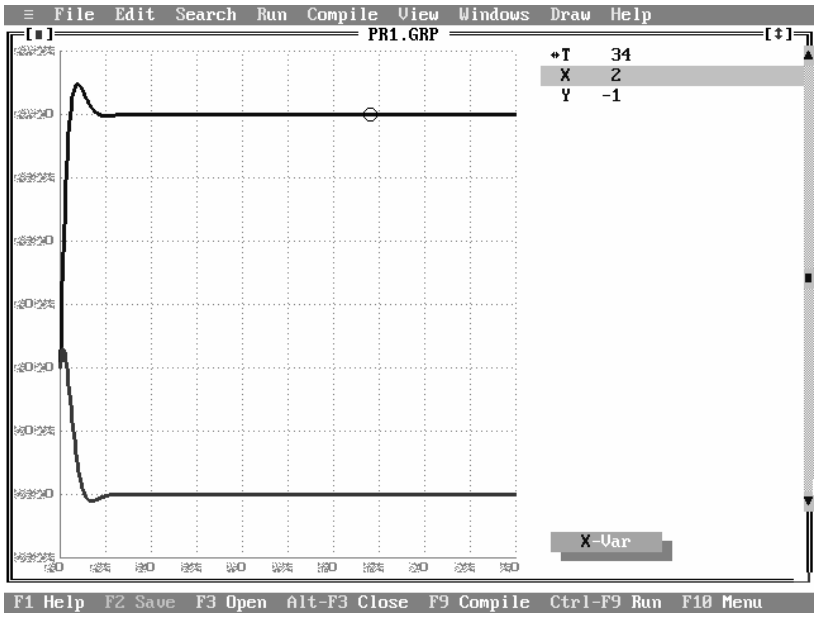
Na začátku kapitoly je zmíněno, že pro většinu soustav je metoda elementární transformace nestabilní. Soustava (48) je jednou z výjimek a byla zvolena záměrně. Upravováním této soustavy si ukážeme, že jakoukoli jinou úpravou dostaneme nestabilní řešení.

Násobíme-li první rovnici soustavy (48) číslem -1, abychom dostali elegantnější rovnici, výsledkem bude nestabilní soustava a nedosáhneme požadovaného řešení. Získáme nestabilní řešení, které je znázorněno na Obrázku 10.

$$\begin{aligned} x' &= \ \ x - y - 3 \\ y' &= -x - y + 1 \end{aligned} \tag{51}$$

Podobně úpravami druhé rovnice nebo obou rovnic dostaneme nestabilní řešení.

Vyskytují se případy SLAR, které nemají žádnou variantu stabilní soustavy získanou elementárním převodem, existují SLAR, u kterých je jedna stabilní varianta soustavy, a některé SLAR, které mají dokonce dvě varianty stabilního řešení.



Obrázek 9: Řešení soustavy (50) a (52)

Dále lze říci, že soustava (48) je jedním z těch příkladů, které mají dvě varianty soustavy stabilního řešení. Následuje postup, který je pracovně nazván „rošáda“. Ukazuje, jak lze rovnice v soustavě (50) zaměnit tak, aby řešení soustavy bylo opět stabilní:

1. na levých stranách ponecháme derivované proměnné,
2. pravou stranu první rovnice násobíme číslem -1,
3. pravé strany rovnic vzájemně vyměníme.

Získáme soustavu:

$$\begin{aligned} x' &= -x - y + 1 \\ y' &= -x + y + 3 \end{aligned} \tag{52}$$

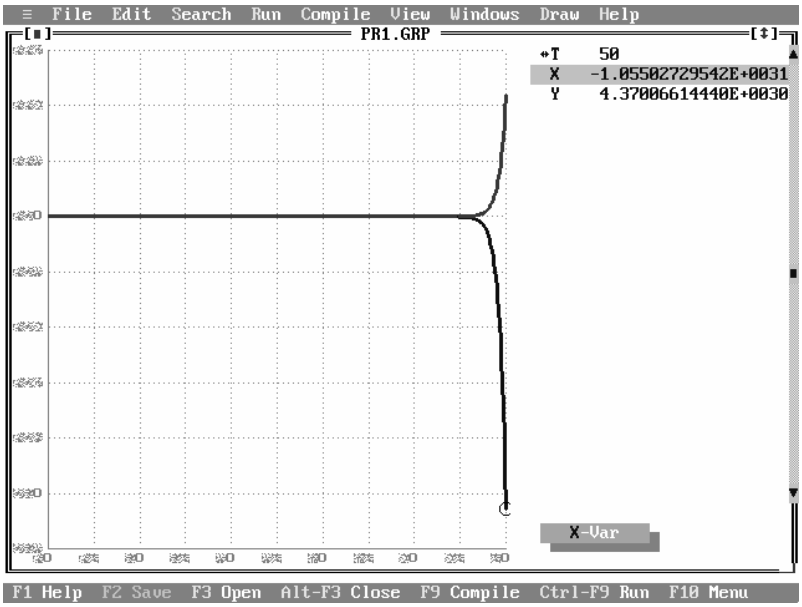
Výsledkem bude stabilní řešení jako v Obrázku 9.

Následné úpravy jednotlivých rovnic násobením číslem -1 způsobí opět nestabilní řešení. Namátkou např. úpravou:

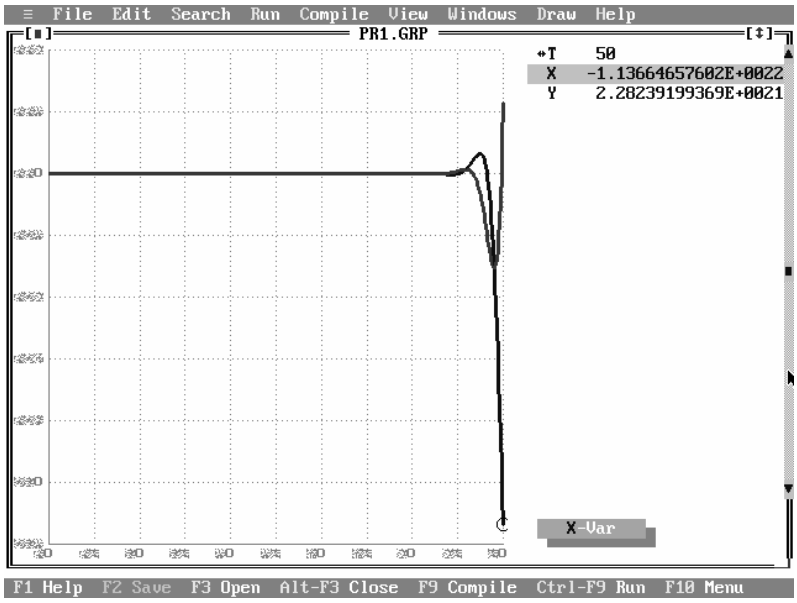
$$\begin{aligned} x' &= -x - y + 1 \quad \left| \cdot (-1) \right. \\ y' &= x - y - 3 \quad \left| \cdot (-1) \right. \end{aligned} \tag{53}$$

dostaneme soustavu rovnic (54) a nestabilní řešení znázorněné na Obrázku 11:

$$\begin{aligned} x' &= x + y - 1 \\ y' &= -x + y + 3 \end{aligned} \tag{54}$$



Obrázek 10: Řešení soustavy (51)



Obrázek 11: Řešení soustavy (54)

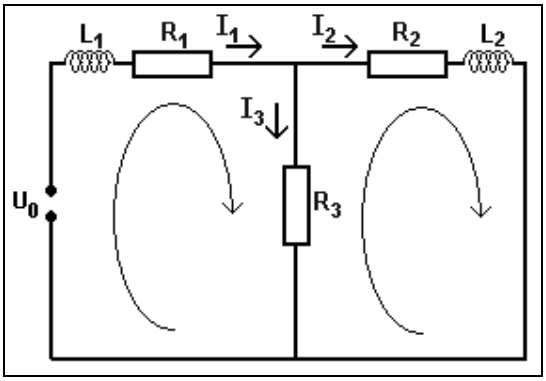
4.3.1 Elementární převod v praktických příkladech

Se SLAR se setkáváme také v jiných oborech než je matematika a lze je pomocí elementárního převodu v programu TKSL řešit. Musíme dbát na správný postup korektního zpracování příkladu,

aby byly rovnice získány v souvislosti s fyzikálními pravidly a aby úpravy, které s rovnicemi provádíme, neměnily charakter soustavy. Při dodržení správného řešení vyjde vždy stabilní řešení. Předveďme si postup na jednoduchém příkladě řešení elektrického obvodu.

4.3.1.1 Příklad – Elektrický obvod 1

Řešte elektrický obvod zadaný Obrázkem 12 a hodnotami: U_0=10V, $R_1 = R_2 = R_3 = 100\Omega$, $L_1 = L_2 = 1\text{C}$.



Obrázek 12: Zadání příkladu 1

Pro upřesnění podmínek a vnesení ustáleného stavu do soustavy zakomponujeme do obvodu parazitní složku realizovanou cívkami před rezistorem R_1 a za R_2. Tyto cívky zaručí, že v okamžiku, kdy zdroj mění polaritu napětí, v cívkách se indukuje proud opačného směru a obvod bude v ustáleném stavu. Při tvoření rovnic pomocí metody uzlových napětí neupravujeme rovnice násobením číslem -1 ani jinými nedovolenými úpravami.

Vyjdeme za základní věty o smyčkových napětí:

$$\begin{aligned} U_{L1} + U_1 + U_3 &= U_0 \\ U_3 + U_2 + U_{L2} &= 0 \end{aligned} \tag{55}$$

Upravíme:

$$\begin{aligned} U_{L1} + R_1 \cdot I_1 + R_3 \cdot (I_1 - I_2) &= U_0 \\ R_3 \cdot (I_2 - I_1) + R_2 \cdot I_2 + U_{L2} &= 0 \end{aligned} \tag{56}$$

Dosazením vzorce pro indukci $U_L = L \cdot I_L'$:

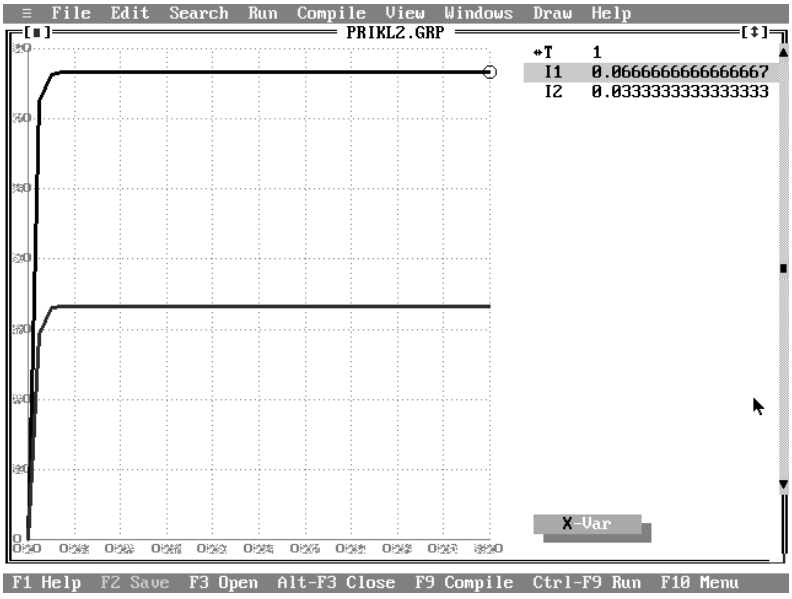
$$\begin{aligned} L_1 \cdot I_{L1}' + R_1 \cdot I_1 + R_3 \cdot (I_1 - I_2) &= U_0 \\ R_3 \cdot (I_2 - I_1) + R_2 \cdot I_2 + L_2 \cdot I_{L2}' &= 0 \end{aligned} \tag{57}$$

Roznásobením a anulováním dostaneme:

$$I'_{L1} = \frac{1}{L_1} \cdot \left[U_0 - I_1 \cdot \left(R_1 + R_3\right) + I_2 \cdot R_3\right]$$
$$I'_{L2} = \frac{1}{L_2} \cdot \left[I_1 \cdot R_3 - I_2 \cdot \left(R_2 + R_3\right)\right] \tag{58}$$

Dosazením hodnot získáme soustavu (56), pro kterou po spuštění výpočtu s nulovými počátečními podmínkami dostaneme řešení na Obrázku 13:

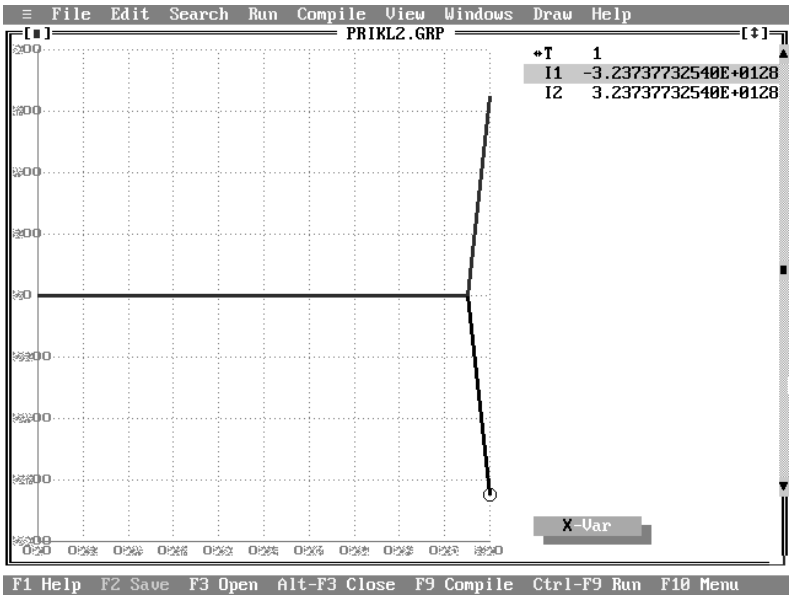
$$I'_{L1} = -200 \cdot I_1 + 100 \cdot I_2 + 10$$
$$I'_{L2} = 100 \cdot I_1 - 200 \cdot I_2 \tag{59}$$



Obrázek 13: Stabilní řešení soustavy (59)

Stabilní řešení dané soustavy odpovídá výsledkům vyřešeným některou z klasických metod, tedy proudy v jednotlivých částech elektrického obvodu jsou: $I'_{L1} = I_1 = 0{,}0\overline{6}A$, $I'_{L2} = I_2 = 0{,}0\overline{3}A$. Proud na rezistoru R_3 lze dopočítat Kirchhofovým zákonem pro smyčkové proudy: $I_3 = I_1 - I_2 = 0{,}0\overline{6} - 0{,}0\overline{3} = 0{,}0\overline{3}A$.

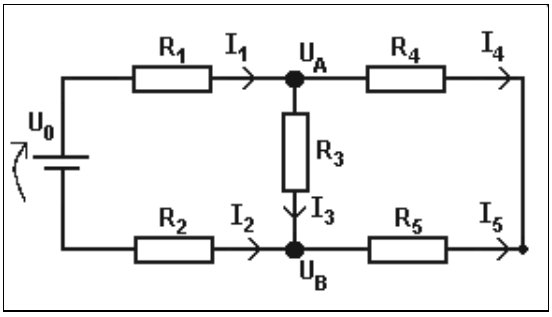
Při upravování rovnic si musíme dát pozor, abychom neudělali úpravu, která může ovlivnit stabilitu řešení. Pokud bychom např. násobili první rovnici soustavy (59) číslem -1, abychom dostali „elegantnější“ rovnici, nebo bychom parazitní proudy na jednotlivých cívkách nesprávným postupem zaměnili, dostali bychom nestabilní soustavu. Řešení by vypadalo podle Obrázku 14.



Obrázek 14: Nestabilní řešení SDR (59) po nedovolené úpravě

4.3.1.2 Příklad – Elektrický obvod 2

Na obdobném příkladu z oboru elektrotechniky si ukážeme další možnost, jak obvod vyřešit jednoduše elementárním převodem. Zadání příkladu elektrického obvodu v Obrázku 15 má parametry: $U_0 = 10\text{V},\ R_1 = R_2 = R_3 = R_4 = R_5 = 100\Omega$. Úkolem je vypočítat napětí $U_A,\ U_B$.



Obrázek 15: Zadání příkladu 2

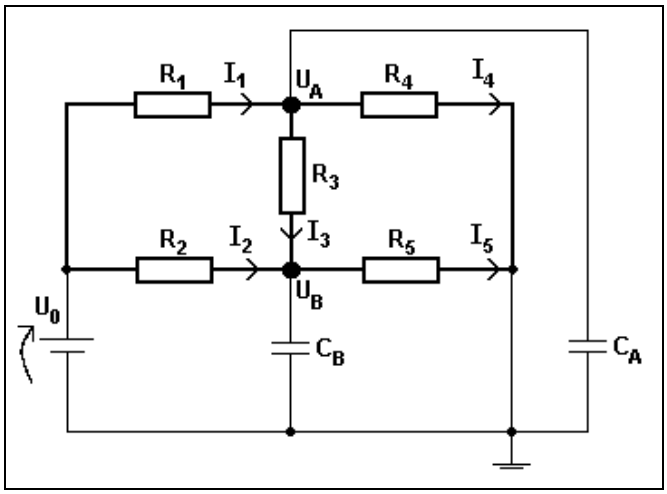
Běžným postupem pro vypočet hledaných uzlových napětí vyjdeme z rovnic pro proudy v jednotlivých uzlech a získáme soustavu rovnic:

$$\begin{aligned} I_1 &= I_3 + I_4 \\ I_2 + I_3 &= I_5 \end{aligned} \tag{60}$$

Dosazením vzorce $I = \frac{U}{R}$ dostaneme soustavu.

$$\begin{aligned} \frac{U_0 - U_A}{R_1} &= \frac{U_A - U_B}{R_3} + \frac{U_A}{R_4} \\ \frac{U_0 - U_B}{R_2} + \frac{U_A - U_B}{R_3} &= \frac{U_B}{R_5} \end{aligned} \tag{61}$$

Pro výpočet s použitím diferenciálních rovnic zaneseme do obvodu parazitní prvky v podobě kondenzátorů $C_A = C_B = 100\text{Q}$, které zaručí ustálený stav dle Obrázku 16. Ustálený stav obvodu nastane, jakmile jsou kondenzátory ve stavu nabití nebo vybití.



Obrázek 16: Zadání příkladu 2 se zavedením ustalující složky

Nyní bude základní výchozí rovnicí pro korektní zpracování v závislosti na rovnicích:

$$\begin{aligned} I_1 - I_3 - I_4 &= I_A \\ I_2 + I_3 - I_5 &= I_B \end{aligned} \tag{62}$$

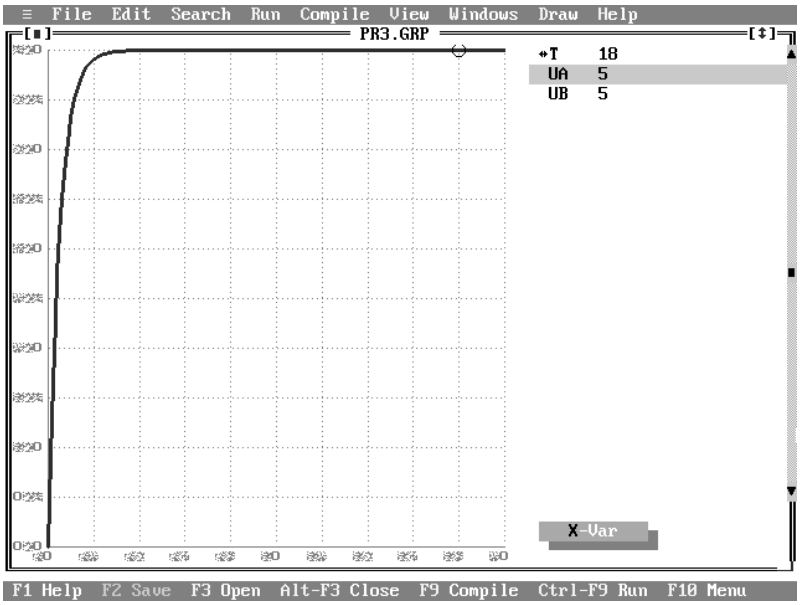
a použitím vzorce pro proud na kondenzátoru $I = C \cdot U'$ přepíšeme soustavu (62) s parazitními proudy na soustavu:

$$\begin{aligned} \frac{1}{C} \cdot \left(\frac{U_0 - U_A}{R_1} - \frac{U_A - U_B}{R_3} - \frac{U_A}{R_4} \right) &= U'_A \\ \frac{1}{C} \cdot \left(\frac{U_0 - U_B}{R_2} + \frac{U_A - U_B}{R_3} - \frac{U_B}{R_5} \right) &= U'_B \end{aligned} \tag{63}$$

Dosazením hodnot a upravením dostaneme soustavu rovnic (64), kterou řešíme pomocí programu TKSL.

$$\begin{aligned} -3\cdot U_A \quad + U_B + 10 &= U'_A \\ U_A - 3\cdot U_B + 10 &= U'_B \end{aligned} \tag{64}$$

Výsledná soustava má stabilní řešení znázorněné na Obrázku 17. Opět je nutno připomenout, že uvedený postup je třeba dodržet. Při jakémkoli zásahu do soustavy se stabilita řešení naruší.



Obrázek 17: Řešení příkladu 2

4.4 Experimenty

Řešení numerických úloh lze brát jako postup, kdy vstupním údajům přiřazujeme vstupní data. Je-li toto přiřazení spojitým zobrazením, pak říkáme, že numerická úloha je korektní. Znamená to, že pro tyto úlohy má zásadní význam citlivost výsledku na malé změny ve vstupních parametrech úlohy.

4.4.1 Špatně podmíněné soustavy

Soustava, resp. matice soustavy je dobře podmíněná, pokud malým relativním změnám vstupních údajů odpovídají malé relativní změny výstupních údajů. Číslo C_p nazýváme číslem podmíněnosti úlohy a pro dobře podmíněné úlohy se blíží k 0.

$$C_p = \frac{relativní\_chyba\_výstupních\_údajů}{relativní\_chyba\_vstupních\_údajů}$$

Pro špatně podmíněné matice platí, že i malá relativní změna vstupního údaje nebo údajů, způsobí velkou relativní změnu výstupních údajů. Uvedeme si příklad na soustavách lineárních rovnic (65) a (66).

$$\begin{aligned}
&2\cdot x_1 + x_2 + x_3 + \ x_4 - 2\cdot x_5 + x_6 \ + x_7 = 2\\
&2\cdot x_1 \ - 2\cdot x_3 + 2x_4 + x_5 + 2\cdot x_6 - 2\cdot x_7 = 0\\
&2\cdot x_1 + 3\cdot x_2 + x_3 \ - x_5 \ + 3\cdot x_7 = 1\\
&\quad 2\cdot x_2 + x_3 + x_4 + 2\cdot x_5 + x_6 \ + x_7 = 0\\
&3\cdot x_1 \ + x_4 + 3\cdot x_5 + 2\cdot x_6 + 3\cdot x_7 = -1\\
&2\cdot x_1 + x_2 + 3\cdot x_3 \ + x_5 - x_6 \ = 2\\
&-x_1 + 2\cdot x_2 + x_3 + x_4 \ + 2\cdot x_7 = 1
\end{aligned} \tag{65}$$

$$\begin{aligned}
&1{,}99999\cdot y_1 + y_2 + y_3 + y_4 - 2\cdot y_5 + y_6 + y_7 = 2\\
&2\cdot y_1 \ - 2\cdot y_3 + 2y_4 + y_5 + 2\cdot y_6 - 2\cdot y_7 = 0\\
&2\cdot y_1 + 3\cdot y_2 + y_3 \ - y_5 \ + 3\cdot y_7 = 1\\
&\quad 2\cdot y_2 + y_3 + y_4 + 2\cdot y_5 + y_6 \ + y_7 = 0\\
&3\cdot y_1 \ + y_4 + 3\cdot y_5 + 2\cdot y_6 + 3\cdot y_7 = -1\\
&2\cdot y_1 + y_2 + 3\cdot y_3 \ + y_5 - y_6 \ = 2\\
&-y_1 + 2\cdot y_2 + y_3 + y_4 \ + 2\cdot y_7 = 1
\end{aligned} \tag{66}$$

Všimněme si, že jen první člen je u druhé soustavy rovnic zmenšen o číslo 0,00001, což lze považovat za malou relativní změnu ve vstupních parametrech. Řešení soustav získané pomocí Gauss-Seidlovy metody vypadá:

$$\begin{aligned}
x_1 &= 16{,}38227 & y_1 &= -0{,}3\\
x_2 &= -2{,}66510 & y_2 &= 0{,}99998\\
x_3 &= 14{,}58818 & y_3 &= 0{,}5\\
x_4 &= 13{,}27345 & y_4 &= 0{,}9\\
x_5 &= 8{,}357411 & y_5 &= -0{,}4\\
x_6 &= -29{,}24625 & y_6 &= -0{,}5\\
x_7 &= -10 & y_7 &= -0{,}1
\end{aligned} \tag{67}$$

Vidíme, že výsledky se liší až o dva řády.

Špatná podmíněnost je vlastnost matice a výsledek se nezlepší, i když k řešení soustavy zvolíme jakýkoliv algoritmus nebo metodu. Pak řekneme, že metoda počítání je bezcenná, jestliže nemůžeme zaručit naprosto přesné vstupní parametry.

Již víme, že stabilitu řešení numerické úlohy můžeme ověřit pomocí vlastních čísel charakteristické rovnice. Dále také platí, že soustava je stabilní, pokud drobná změna vstupních

údajů vyvolá jen drobnou změnu výsledku, což lze ověřit hodnotami inverzní matice. Platí, že mají-li matice soustavy A a matice A^{-1} srovnatelné prvky, potom je soustava stabilní. V opačném případě jde o nestabilní úlohu. U rozsáhlejších soustav se špatně podmíněnými maticemi mohou být změny ve výsledku mnohem výraznější. Změny záleží na tvaru matice koeficientů soustavy. Ověříme tuto vlastnost na soustavě (68).

$$\begin{aligned} -2\cdot x + 6\cdot y &= 8 \\ -2\cdot x + 6{,}000001\cdot y &= 7{,}999999 \end{aligned} \tag{68}$$

Charakteristická rovnice pro soustavu a její kořeny mají tvar:

$$\lambda^2 - 4{,}000001\cdot\lambda - 0{,}0000019 = 0\,,\ \lambda_{1,2} = \begin{cases} -4{,}0000014 \\ -0{,}0000005 \end{cases} \tag{69}$$

Vidíme, že vlastní čísla jsou záporná, takže soustava (68) je nestabilní, což nám potvrdí i velmi rozdílné prvky matice soustavy a inverzní matice:

$$A\cdot E = \left(\begin{array}{cc|cc} -2 & 6 & 1 & 0 \\ -2 & 6{,}000001 & 0 & 1 \end{array}\right) = \left(\begin{array}{cc|cc} -2 & 6 & 1 & 0 \\ 0 & -0{,}000001 & 1 & -1 \end{array}\right) = \left(\begin{array}{cc|cc} -2 & 6 & 1 & 0 \\ 0 & 1 & -1000000 & 1000000 \end{array}\right) =$$

$$= \left(\begin{array}{cc|cc} -2 & 0 & 6000001 & -6000000 \\ 0 & 1 & -1000000 & 1000000 \end{array}\right) = \left(\begin{array}{cc|cc} 1 & 0 & -3000000{,}5 & 3000000 \\ 0 & 1 & -1000000 & 1000000 \end{array}\right)$$

$$A = \begin{pmatrix} -2 & 6 \\ -2 & 6{,}000001 \end{pmatrix},\ A^{-1} = \begin{pmatrix} -3000000{,}5 & 3000000 \\ -1000000 & 1000000 \end{pmatrix} \tag{70}$$

Špatná podmíněnost matic se projevuje také u stabilních soustav a je většinou spjata s problémem rychlého získání výsledků soustavy. Příkladem soustavy, jejíž řešení je i elementárním převodem stabilní, může být:

$$\begin{aligned} 2\cdot x + 6\cdot y &= 8 \\ 2\cdot x + 6{,}000001\cdot y &= 7{,}999999 \end{aligned} \tag{71}$$

Charakteristická rovnice pro soustavu a její kořeny mají tvar:

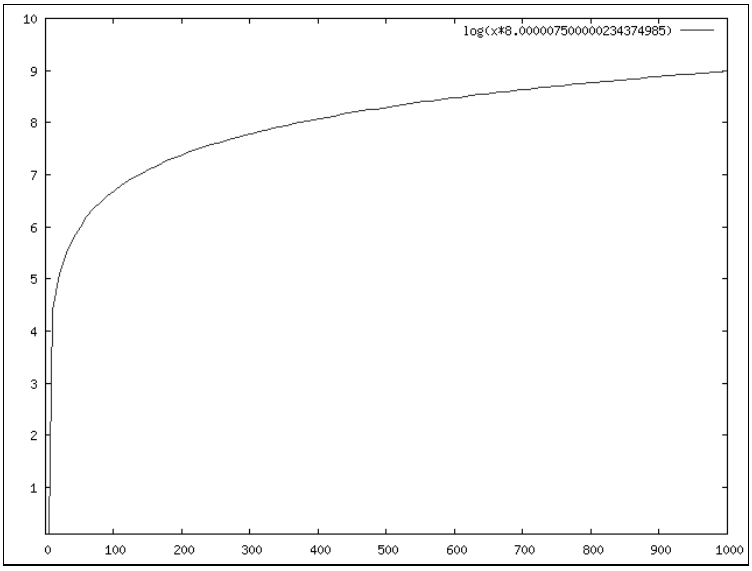
$$\lambda^2 - 8{,}000001\cdot\lambda + 0{,}0000019 = 0\,,\ \lambda_{1,2} = \begin{cases} 8{,}0000087 \\ 2{,}24999\cdot 10^{-7} \end{cases} \tag{72}$$

Napíšeme-li pro soustavu (71) analytické řešení, jehož obecný tvar (33) byl popsán v kapitole 4, dostaneme rovnici s partikulárními výsledky:

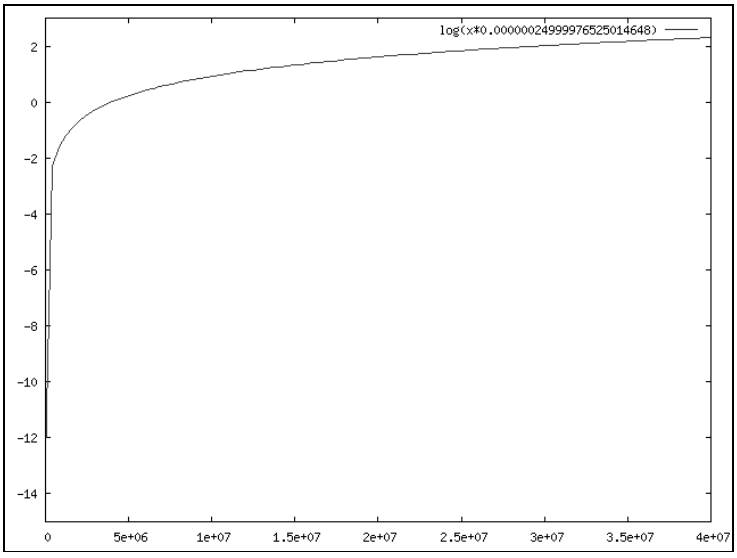
$$y = c_1\cdot e^{8{,}0000087\ \cdot x} + c_2\cdot e^{2{,}24999\ \cdot 10^{-7}\cdot x} \tag{73}$$

Nyní se vrátíme k vlastnostem exponenciální a logaritmické funkce a partikulární výsledky si pomocí funkcí zobrazíme. Protože se při řešení SLAR snažíme dosáhnout ustáleného řešení, řešíme soustavu tak dlouho, dokud se všechny exponenciální křivky (exponenciály) nedostanou z přechodového stavu do stabilního stavu.

Využijeme zobrazení výsledků pomocí logaritmické funkce První složka s větším kladným číslem je zobrazena na Obrázku 18 a druhá složka na Obrázku 19.



Obrázek 18: Partikulární řešení $e^{8,0000087x}$



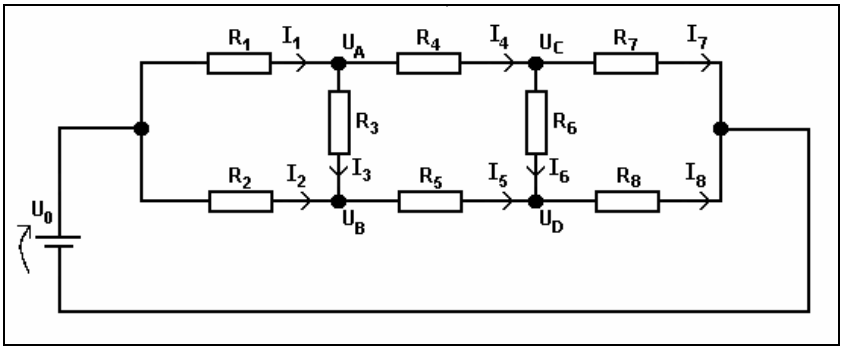
Obrázek 19: Partikulární řešení $e^{2,24999 \cdot 10^{-7} x}$

Porovnáme-li oba obrázky, vidíme, že v grafu zobrazující první složku na Obrázku 18 se partikulární řešení blíží do ustáleného stavu mnohem dříve než řešení druhé části na Obrázku 19. Hodnotu druhé složky řešení budeme schopni přesně říci až za velmi dlouhý časový interval. Podle experimentů nezáleží u špatně podmíněných soustav na metodě řešení, čas ustálení výsledku záleží na rychlosti ustálení pomalejší složky.

Vlastnost, o které se musíme zmínit v souvislosti se špatně podmíněnými soustavami, je singularita a regulárnost matic soustav. Jestliže špatně podmíněná soustava bude zadaná jako téměř singulární, pak platí, že bude mít nejméně dvě téměř lineárně závislé rovnice. Z experimentů vyplývá, že řešení jakékoli soustavy, která má lineárně závislé řádky, záleží na počátečních podmínkách. Pak je-li soustava „správně“ zapsaná, má regulární matici a řešení na jiných parametrech nezáleží.

Znalosti o chování špatně podmíněných soustav lze v praxi velice dobře využít, např. v návrhu elektrických obvodů. Představme si rozsáhlou síť, ve které návrhář změní některý z členů, a pak při počítání s mnoha rovnicemi obvod nepracuje správně. Podle výsledků můžeme zpětně určit, proč se tak stalo.

Demonstrujme si zjednodušenou síť na příkladě, který je zadán Obrázkem 20 a parametry: $U_0 = 32\text{V},\ R_1 = 200\Omega,\ R_2 = 300\Omega,\ R_3 = 200,\ R_4 = 0{,}0002\Omega,\ R_5 = 500\Omega,$ $R_6 = 100\Omega,\ R_7 = 200\Omega,\ R_8 = 100\Omega$. Úkolem je vypočítat napětí $U_A,\ U_B,\ U_C,\ U_D$.



Obrázek 20: Zadání příkladu

Podobně jako jsme postupovali v příkladě Elektrického obvodu 2 v části 4.3.1.2, nyní vneseme do obvodu kondenzátory o velikosti $C_A = C_B = C_C = C_D = 10^{-3}\,\text{Q}$ a pomocí Kirchhofových zákonů vytvoříme rovnice soustavy:

$$\begin{aligned}
\frac{U_0 - U_A}{R_1} - \frac{U_A - U_B}{R_3} - \frac{U_A - U_C}{R_4} &= C \cdot U'_A \\
\frac{U_0 - U_B}{R_2} + \frac{U_A - U_B}{R_3} - \frac{U_B - U_D}{R_5} &= C \cdot U'_B \\
\frac{U_A - U_C}{R_4} - \frac{U_C - U_B}{R_6} - \frac{U_C}{R_7} &= C \cdot U'_C \\
\frac{U_B - U_D}{R_5} + \frac{U_C - U_D}{R_6} - \frac{U_D}{R_8} &= C \cdot U'_D
\end{aligned} \tag{74}$$

Po dosazení hodnot a upravení dostaneme SDR v elementární úpravě:

$$\begin{array}{cccccc}
-5000010 \cdot U_A & 5 \cdot U_B & -5000000 \cdot U_C & & +0{,}8 & = U'_A \\
5 \cdot U_A & -10{,}\overline{3} \cdot U_B & & 2 \cdot U_D & +106{,}\overline{6} & = U'_B \\
5000010 \cdot U_A & -5 \cdot U_B & -5000005 \cdot U_C & & & = U'_C \\
 & 2 \cdot U_B & 10 \cdot U_C & -22 \cdot U_D & & = U'_D
\end{array} \tag{75}$$

Již v zadání parametrů odporů jsme si mohli všimnout velice rozdílných hodnot mezi rezistory R_1, R_7 a R_4. Hodnoty odlišné řádově desetitisíce se projeví i v rovnicích. Všimneme si, že soustava (75) je špatně podmíněná, můžeme říci téměř singulární, protože první a třetí rovnice jsou si velice podobné.

Špatná podmíněnost matic má za následek špatnou konvergenci řešení soustavy. Hodnoty výsledků zobrazené v (76) byly vypočítány programem TKSL až po velice dlouhém časovém intervalu. Lze z nich vyčíst, že napětí mezi rozdílně zadanými hodnotami rezistorů, jsou téměř stejná čísla a až tisícem řádů se odlišují od ostatních výsledků soustavy. Z částečných výsledků bylo vysledováno, že řešení konvergují velice pomalu.

$$\begin{aligned}
U'_A &= 8{,}0019167688 \cdot 10^{-8} \\
U'_B &= 2{,}9994295326 \cdot 10^{-5} \\
U'_C &= 7{,}9999999962 \cdot 10^{-8} \\
U'_D &= -5{,}8054819097 \cdot 10^{-6}
\end{aligned} \tag{76}$$

Z praktického hlediska můžeme z výsledků určit, která napětí jsou podobná a v návrhu tato místa podrobně prozkoumat, zda-li se v návrhu nestala chyba. Špatně podmíněné soustavy jsou jedním z případů soustav se silným tlumením, které si představíme nyní.

4.4.2 Soustavy se silným tlumením

Soustavy se silným tlumením jsou také nazývány tuhými systémy či stiff. Jsou to soustavy, které obsahují útlum s charakteristickým časem τ mnohem menším než jsou zbylé charakteristické časy úlohy s obvyklým krokem $h \cong 0{,}01$. V čase větším než τ je rychle tlumící se složka zanedbatelně malá a na výsledku se neprojevuje. Tlumící složka má za následek, že řešení konverguje ke správnému výsledku velice pomalu. Dalším z důvodů, proč je výhodné používat SDR pro řešení SLAR, je sledování právě této konvergence řešení. Abychom soustavy a jejich řešení, které konvergují pomalu, mohli urychlit, musíme je detekovat a rozlišit.

Pro detekci stiff SLAR, jejichž řešení bude konvergovat pomalu, využijeme reálné části vlastních čísel, které budou splňovat vztah: $|\mathrm{Re}\,\lambda_{\max}| >> |\mathrm{Re}\,\lambda_{\min}|$. Čím menší je $|\mathrm{Re}\,\lambda_{\min}|$, tím je exponenciála pomalejší a tím větší časový interval potřebujeme k vyřešení soustavy. Na druhé straně musíme dbát, aby integrační krok zvolený vůči $|\mathrm{Re}\,\lambda_{\max}|$ zachovával jeho stabilitu. Proto při volbě integračního kroku s velice rozdílnými vlastními čísly nastane obtížná situace, kdy na velice dlouhém časovém intervalu řešení počítáme s velice malým integračním krokem. Podle vzorce (77) zjistíme koeficient systému se silným tlumením s (koeficient tuhosti či koeficient podmíněnosti). Čím větší je toto číslo, tím hůře se soustava chová.

$$s = \frac{|\lambda_{\max}|}{|\lambda_{\min}|} \tag{77}$$

Má-li koeficient tuhosti hodnotu přesahující stovky řádů, můžeme o soustavě říci, že alespoň jedno její řešení konverguje k výsledku velice pomalu, a soustavu nazveme tuhým systémem.

Vrátíme se k příkladu soustavy (71) a podíváme se, jak se tato soustava bude chovat z hlediska nově zjištěných skutečností. Je vidět, že poměr vlastních čísel přesahuje milion řádů. Soustava je sice stabilní, ale i když použijeme transformaci soustavy do stabilního stavu, koeficient tuhosti nezlepšíme.

Z experimentů bylo zjištěno, že špatně podmíněná soustava je soustavou se silným tlumením vždy, když jsou její rovnice singulární nebo alespoň téměř singulární, přičemž nezáleží na stabilitě řešení. Ani transformační algoritmus konvergenci řešení nezlepší, koeficient tuhosti (podmíněnosti) počítaný pro transformovanou soustavu se dále zvětšuje.

5 Srovnání metod

Srovnáme-li řešení převodu s řešením klasických přímých numerických metod, pak můžeme říci, že rychlost je ovlivněna nejen řádem soustavy, ale také vlastnostmi matice a matice transponované. Platí, že pro některé hůře podmíněné matice z hlediska metody převodu i nízkého řádu trvá řešení dlouho. Ve srovnání s tím je řešení „dobrých matic" vyšších řádů relativně rychlé.

Daleko příznivěji vychází srovnání s iteračními metodami. V případě, že je počet iterací metody převodu na soustavu diferenciálních rovnic přibližně stejný jako počet iterací iterační metody, pak platí, že metoda převodu je rychlejší, neboť iterace mohou být počítány paralelně. Aby klasická iterační metoda byla rychlejší, musela by konvergovat n-krát rychleji, kde n je počet rovnic v soustavě. A to není možné, protože podmínky konvergence klasických iteračních metod a metody převodu jsou srovnatelné.

Numerické metody řešení SLAR lze hodnotit pomocí dvou kritérií, které jsou pro nás důležité: rychlost výpočtu a přesnost výsledku. Závislost těchto aspektů je nepřímo úměrná, chceme-li získat výsledek co nejrychleji, potřebujeme co nejméně výpočetních operací, které zapříčiní vznik nepřesností ve výpočtu. Aby byl výsledek získán co nejefektivněji, je třeba najít kompromis mezi rychlostí a přesností bez újmy na stabilitě daného řešení.

5.1 Problémy numerických metod

Obecně nejsou definována žádná pravidla, která by určovala, jakou metodu pro danou SLAR zvolit. Pro SLAR, které mají matice tzv. plné, ale nepříliš velké, je doporučeno použít některou z přímých metod. Zatímco pro SLAR s mnohdy velmi rozsáhlou řídkou maticí je výhodnější použít iterační metodu, z důvodu případných potíží s kapacitou paměti počítače. Existují doporučení, proč pro řešení SLAR s plnými maticemi nízkého řádu nepoužívat iterační metodu, naopak vyzdvihují efektivnost první metody. Není definováno striktní rozdělení, které SLAR řešit danou metodou, a je třeba pohlížet na další problémy.

Chyba metody, která pochází z nahrazování nekonečného procesu konečným, je typická pro iterační metody. Snažíme se zajistit, aby řešení konvergovalo ke správnému výsledku, pokud SLAR splňuje podmínky konvergence. Rozhodnout, zda podmínky splňuje, nebývá jednoduché a z toho také vyplývá, že některé soustavy mohou být iterační metodou neřešitelné.

Zdrojem chyb u přímých i iteračních metod jsou zaokrouhlovací chyby. V programovacích jazycích jsou implicitně zadány typy čísel (integer, real, float atd.). Problém je, že čísla některých typů (např. čísla v plovoucí desetinné čárce) nejsou uzavřena na základní aritmetické operace. Součin a podíl čísel může vyžadovat více platných cifer a výsledek je zaokrouhlen na *x* platných cifer. Tyto chyby se v průběhu delšího výpočtu kumulují a důsledkem může být, že konečný výsledek se velmi liší od správného řešení.

Žádná obecně použitelná metoda pro vyloučení zaokrouhlovacích chyb neexistuje. Problém zaokrouhlovacích chyb zdánlivě řeší systémy, které používají exaktní aritmetiku. Jejich problém spočívá v menší rychlosti výpočtu kvůli velkému počtu operací. K přesnějším výpočtům se používají systémy s víceslovnou aritmetikou.

Lze říci, že při volbě jedné z metod pro řešení soustav je jedno z hledisek zjištění, je-li možno učinit chybu metody zanedbatelnou při objemu výpočtů ne větším, než si vyžádá metoda přímá nebo naopak. Chyba metody tvoří vždy menší část celkové chyby, kterou je zatíženo vypočtené řešení.

5.2 Doba výpočtu

5.2.1 Přímé metody

U přímých metody musíme brát na vědomí velkou výpočetní náročnost. Např. u metody Cramerových vzorců dostaneme přesné řešení, avšak je zapotřebí vypočítat *(n+1)* determinantů *n*-tého řádu a tím vzniká problém vyčíslování velkého počtu násobení a dělení. Pro rozsáhlejší soustavy je jejich použití problematické, protože ani s pomocí výpočetní techniky nejsme schopni určit přesně hodnoty determinantů. Počet operací, které musíme použít, abychom zjistili všechna řešení soustavy, je naznačen v Tabulce 4.

matice (řádky × sloupce)	počet součinů (podílů)	počet součtů (rozdílů)
2x2	8	3
3x3	51	20
4x4	124	35
5x5	245	54
6x6	426	77
...	…	…
30x30	53 970	1 890

Tabulka 4: Počet operací potřebných k řešení soustavy pomocí Cramerových vzorců

Druhou nejčastější přímou metodou je výpočet pomocí inverzní matice, viz. 2.1.1.5 Pro vyřešení soustavy rovnic musíme provést dva kroky, při kterých počet operací narůstá:

1. výpočet inverzní matice k matici soustavy,
2. dosazení a výpočet řešení.

Počet řádků a počet sloupců je stejný, označíme pouze *i*, obecně je počet operací značen *o*. Označení *o1* představuje operaci součinu nebo podílu, *o2* znamená operaci součtu nebo rozdílu. Předpokládejme složitě zadanou matici, tzn. matici, jejíž čísla na hlavní diagonále nejsou rovny jedné, aby se co nevíce odlišovala od jednotkové matice a měla co nejméně nulových prvků. Pro ilustraci vypočítáme příklad soustavy tří rovnic (78). Přehled prováděných operací ukazuje Tabulka 5.

$$\begin{aligned} 3\cdot x-4\cdot y+5\cdot z&=1\\ 2\cdot x-3\cdot y+z&=-3\\ 3\cdot x-5\cdot y-z&=-4 \end{aligned} \qquad (78)$$

Pro matici soustavy vypočteme inverzní matici pomocí rozšířené Gaussovy eliminace:

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

$$\left(\begin{array}{ccc|ccc} 3&-4&0&-4&-15&-5\\ 0&1&0&-5&18&-7\\ 0&0&1&1&-3&1 \end{array}\right)\cong\left(\begin{array}{ccc|ccc} 1&0&0&-8&29&-11\\ 0&1&0&-5&18&-7\\ 0&0&1&1&-3&1 \end{array}\right)\Rightarrow\ A^{-1}=\begin{pmatrix} -8&29&-11\\ -5&18&-7\\ 1&-3&1 \end{pmatrix}$$

Vypočtenou inverzní matici dosadíme a vypočteme řešení:

$$X=\begin{pmatrix} -8&29&-11\\ -5&18&-7\\ 1&-3&1 \end{pmatrix}\cdot\begin{pmatrix} 1\\ -3\\ -4 \end{pmatrix}\Rightarrow\begin{array}{c} x=-51\\ y=-31\\ z=6 \end{array}$$

Typ matice	Inverzní matice		Výpočet řešení
	rozšířená Gaussova eliminace	pomocí determinantů	
2x2	2·2· (2·*o1*+2·*o2*)	8·*o1*+1·*o2*	2·(2·*o1*+1·*o2*)
3x3	3·2·(3·*o1*+3·*o2*)	3·3·(24·*o1*+7·*o2*)	3·(3·*o1*+2·*o2*)
4x4	4·2·(4·*o1*+4·*o2*)	4·4·(32·*o1*+9·*o2*)	4·(4·*o1*+3·*o2*)
…	…	…	…
obecně	o^3	$k\cdot o^2$	o^2

Tabulka 5: Počet operací pro výpočet řešení pomocí inverzní matice

Celková doba výpočtu pro metodu s použitím inverzní matice je v nejhorším případě o^3+o^2.

5.2.2 Iterační metody

Principem všech iteračních metod je nekonečné počítání s počátečními parametry s cílem dostat co nejpřesnější výsledek. Prakticky však nekončený proces nahrazujeme konečným počtem kroků, a proto můžeme určit, za jak dlouhou dobu se má výpočet ukončit. Víme, že prostá iterační metoda konverguje nezávisle na počáteční aproximaci. K přijetí daného řešení a ukončení výsledku slouží chyba iterační metody. Pokud nám stačí přibližné řešení, určíme příslušnou chybu, resp. přesnost výsledku, a řešení dostaneme po mnohem kratší době než při počítání výsledku přesného na deset desetinných míst a malým integračním krokem.

Dobu výpočtu si předvedeme na příkladě. Máme soustavu rovnic (79), kterou budeme řešit Jacobiho metodou pro počáteční hodnoty $x_1^{(0)} = x_2^{(0)} = x_3^{(0)} = 0$ s chybou $\varepsilon = 10^{-1}$, následně $x_1^{(0)} = x_2^{(0)} = x_3^{(0)} = 100$ se stejnou chybou.

$$\begin{aligned} 2 \cdot x_1 + x_2 &= 200 \\ x_1 + 4 \cdot x_2 + x_3 &= 0 \\ x_2 + 3 \cdot x_3 &= -200 \end{aligned} \tag{79}$$

Podle vzorce (13) vypočítáme rekurentní vztah postupných iterací:

$$\begin{aligned} x_1^{(k+1)} &= \frac{1}{2} \cdot \left(200 - x_2^{(k)}\right) \\ x_2^{(k+1)} &= \frac{1}{4} \cdot \left(- x_1^{(k)} - x_3^{(k)}\right) \\ x_3^{(k+1)} &= \frac{1}{3} \cdot \left(-200 - x_2^{(k)}\right) \end{aligned} \tag{80}$$

které vneseme do Tabulky 6 postupným doplňováním hodnot. Porovnáním mohou být výsledky získané vypočítáním soustavy (79) klasickou přímou metodou: $x_1 = 105{,}26315$, $x_2 = -10{,}52631$, $x_3 = -63{,}15789$.

Z Tabulky 6 lze vyčíst, že pro různé počáteční hodnoty dostaneme výsledky v různých iteračních krocích. Při nulových startovacích hodnotách, dostaneme výsledek s přesností na jedno desetinné místo ve dvanáctém kroku, na rozdíl od počátečních hodnot rovnající se číslu 100, kdy dostaneme stejné řešení již v devátém kroku. U výpočtu soustavy (79) bylo e zvoleno veliké číslo, ale kdybychom přesnost snížili, řešení dostaneme mnohem později a opět v odlišných časech. Obecně tedy dobu iteračních metod nelze určit, záleží na konvergenci jednotlivých soustav.

krok / výsledky	x_1		x_2		x_3	
0	0	100	0	100	0	100
1	100	125	-8,33333	-50	-66,66666	
2	100	106,25	-8,33333	-12,5	-66,66666	
3	104,16666	109,375	-10,06944	-18,75	-63,88888	
…	…	…	…	…	…	…
8	104,38368	105,36349	-10,50648	-10,54416	-63,74421	-63,09100
9	105,25324	**105,2**7208	-10,15986	**-10,5**6812	-63,16450	**-63,1**5194
10	105,07993		-10,52218		-63,28004	
11	105,26109		-10,44997		-63,15927	
12	**105,2**2490		**-10,5**2545		**-63,1**8334	

Tabulka 6: Výsledky soustavy (79) řešené Jacobiho metodou

5.2.3 Doba výpočtu převodu

Elementární převod do SDR má prakticky stejnou podobu jako SDR, která je řešená jakoukoli iterační metodou. Řešení programem TKSL používá jednu z iteračních metod Taylorova typu, takže tyto doby výpočtů obou postupů jsou srovnatelné.

Zajímavější porovnání vychází s metodou transformačního algoritmu. I při výpočtu SDR s transponovanou maticí program využívá iterační metodu, avšak doba výpočtu může být vzhledem ke klasickým řešením kratší.

U soustav s malým počtem rovnic, které jsou zadány tak, že splňují podmínku konvergence, je rozdíl času řešení nepatrný. Je-li však podmínka porušena a iterační metoda v konečném čase nedokáže určit výsledky, převod transformačního algoritmu výpočet výsledků zaručí. Doba výpočtu je závislá na ustálení nejpomalejší složky, ale řešení bude dosaženo v konečném čase. Pro řešení soustav s velkým množstvím rovnic je doba výpočtu pomocí převodu vzhledem k přímým i vzhledem k iteračním metodám mnohem rychlejší. Urychlení tohoto postupu je neustále zkoumáno a dalším krokem ve vývoji je použití paralelních výpočtů.

5.3 Konvergence numerického řešení

Základní vlastnost, kterou od použitelné numerické metody požadujeme, je konvergence numerického řešení y_i k teoretickému řešení $y(x_i)$ dané úlohy. Konverguje-li přibližné řešení získané určitou metodou pro všechny počáteční úlohy, řekneme, že metoda je konvergentní. Konvergence vyjadřuje buď závislost chyby iterace na chybě v předchozí iteraci, nebo celkovou rychlost zmenšování chyby při neomezeně rostoucím počtu iterací. Obecně lze podmínku konvergence vyjádřit pomocí tří po sobě jdoucích partikulárních řešeních v posloupnosti vzniklé iterováním:

$$|x_3 - x_2| < |x_2 - x_1| \tag{81}$$

Vhodným zvolením iteračního kroku *h* a řádu metody lze předejít kumulování chyb v procesu iterování. Pro každý iterační krok existuje takový řád Taylorovy řady, který zaručí konvergenci jednotlivých partikulárních řešeních a tedy i celkovou konvergenci metody. Zvolíme-li pro daný krok nižší řád, než jaký je doporučen, pak posloupnosti funkčních hodnot bude divergovat. Problémem tedy je, jak k řádu metody najít vhodný iterační krok a naopak, aby řešení konvergovalo. K hledání parametrů lze přistoupit takto:

1. zvolíme iterační krok *h* a k němu určíme řád Taylorovy řady podle Algoritmu 1,
2. zvolíme řád a k němu hledáme příslušný iterační krok dle Algoritmu 2.

Algoritmus 1

1. Zvolíme počáteční krok.
2. Vypočítáme $x_{j,1}$, $x_{j,2}$, $x_{j,3}$ pro *j=1, 2,… n,* kde *n* je počet rovnic SLAR s nulovými počátečními podmínkami.
3. Ověříme, zda pro všechna x_j platí podmínka (73).
4. Pokud je podmínka splněna, můžeme s použitím vypočteného kroku SLAR řešit bez nebezpečí divergence. Neplatí-li bod 3, snížíme iterační krok a opakujeme algoritmus od bodu 2.

Algoritmus 2

1. Zvolíme první řád Taylorova rozvoje.
2. Vypočítáme $x_{j,1}$, $x_{j,2}$, $x_{j,3}$ pro *j=1, 2,… n,* kde *n* je počet rovnic SLAR s nulovými počátečními podmínkami.

3. Ověříme, zda pro všechna x_j platí podmínka (73).
4. Neplatí-li bod 3, lze řád metody považovat za vhodný se zaručením konvergence. Pokud není podmínka splněna, zvýšíme řád o jedna a vrátíme se na bod 2.

Při hledání vhodného iteračního kroku k řádu metody nelze přesně říci, který je největší přípustný. Může existovat velké množství kroků, které konvergenci řešení splňují. Hledání kroku záleží na výchozí hodnotě kroku a na hodnotě, o kterou v bodu 4 snižujeme. Proto při volbě odpovídajících parametrů metody, určení kroku a hledání příslušného řádu nebo naopak pomocí algoritmu, lze zaručit požadavek konvergence řešení. Na dalších místech se musíme zabývat také přesností a rychlostí, pro něž nalezené hodnoty vyhovovat nemusí.

Přesnost metody převodu lze teoreticky nekonečně zvyšovat. Omezení je dáno pouze použitou metodou vyčíslování a numerického počítání. Otázkou zůstává, za jakých podmínek a omezení jsme schopni vyšší přesnosti dosáhnout. Po experimentování se SDR v programu TKSL lze říci, že velikost chyby závisí na hodnotě determinantu. Čím je soustava hůře podmíněná, tím je chyba výpočtu větší.

Rychlost konvergence metody je charakterizována řádem metody. Tento řád může být maximálně takové přirozené číslo *p*, aby pro danou metodu aplikovanou na libovolnou počáteční úlohu existovalo číslo *c* nezávislé na kroku konvergence *h*. Přičemž při malém konvergenčním kroku platí pro lokální diskretizační chybu: $|d_i| \leq c \cdot h_i^{p+1}$. Abychom nemuseli složitě zjišťovat řád metody, je vhodné použít transformační algoritmus.

Důsledek použití metody transformace k řešení SLAR je zajištění stability a tím i konvergence řešení. Stabilní systém se chová jako systém se silným tlumením a je tedy zaručeno, že analytické řešení SDR konverguje k řešení SLAR.

Pro rychlost konvergence analytického řešení je určující hodnota nejmenšího vlastního čísla $\lambda_{\min}$. Čím menší bude toto číslo, tím pomalejší bude konvergence řešení. Na určení délky integračního kroku má vliv naopak největší vlastní číslo $\lambda_{\max}$, které čím bude větší, tím bude krok metody menší a tím více iterací budeme potřebovat k dosažení výsledku.

Konvergenci iteračních metod lze urychlit tzv. předpodmíněním. Je to obecný způsob, který je realizovaný vlastní iterační metodou a přímou rychlou metodou, která slouží k předpodmínění. Cenou za urychlení konvergence je větší paměťová a výpočetní náročnost prováděného výpočtu. Nejjednodušším způsobem předpodmínění je trojúhelníková faktorizace nebo tzv. cyklická redukce.

6 Rozhraní TKSL/C - SLAR

Program TKSL/C, který je vyvíjen na Fakultě informatiky VUT v Brně a neustále vylepšován, obsahuje všechny přínosné informace jako starší verze TKSL386, která byla také k ověřování vlastností a k experimentům použita. Principem zůstává metoda výpočtu DR využitím Taylorovy řady.

Existuje mnoho programů, které umožňují automatický výpočet přímých i iteračních metod. Pro program TKSL zatím nebylo vytvořeno rozhraní, které by usnadňovalo uživateli zápis soustav rovnic a zadávání parametrů pro elementární převod a ani transformační převod SLAR do SDR. Proto bylo vhodné vytvořit přátelské grafické rozhraní, které by uživateli urychlilo práci při zadávání vstupů. Rozhraní TKSL/C – SLAR, které bylo naprogramováno v rámci této práce, je napsáno v programovacím jazyce C++ s použitím programu Visual Studio C++ a klade důraz na následující požadavky:

- snadná rozšiřitelnost,
- jednoduchá syntaxe vstupu a výstupu,
- přenositelnost systému, nezávislost systému na použité platformě,
- možnost využití v dalších projektech.

Při vytváření rozhraní se ukázalo, že není nutné zasahovat do zdrojových kódů programu TKSL/C, ale vytvořit vhodný program, který jej používá jako nástroj k výpočtu.

6.1 Program TKSL/C

Program TKSL/C má již zabudováno několik vhodně zvolených parametrů, které uživatel může použít k zadání explicitních hodnot. Příkaz, kterým se volá spuštění výpočtu může mít následující parametry.

```
cltksl [-hHELP] [-VVERSION] [-sSTEP] [-tTMAX] [-WNUMWIDTH] [-OORDER]
      [-AACCURACY] [-ZZEROS] [-wTEXTWIDTH] [-pPRECISION] [-fFORMAT]
      [- | vstup >> výstup]
```

Význam jednotlivých parametrů je zobrazen v Tabulce 7. V rozhraní byla většina parametrů zachována a ponechána s implicitními hodnotami. Zvýrazněné položky v Tabulce 7 je v rozhraní možno měnit podle potřeb uživatele.

h	nápověda programu
V	info o verzi programu
s	**inicializace velikosti kroku, implicitně 0,1**
t	**inicializace hodnoty horní meze výpočtu, implicitně 0,1**
W	velikost čísel [bits], implicitně 80
O	maximální order, implicitně 50
A	nastavení přesnosti výpočtu, 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**
-	načítání rovnic ze standardního vstupu
vstup/výstup	**načítání rovnic ze souboru vstup/ řešení se zapisuje do souboru výstup**

Tabulka 7: Seznam parametrů pro program TKSL/C

6.1.1 Vstupní soubor

Nejjednodušším vstupem v programu TKSL/C je přesměrování souboru, ve kterém je zápis rovnic. Rozhraní také dovoluje uživateli vložit hodnoty jednotlivých proměnných a stisknutím tlačítka mu automaticky vygeneruje soustavu rovnic. Možnosti zobrazení jsou dvojí, soustava bude ve tvaru použitím elementárního převodu, nebo vytvoří SDR s použitím transformačního algoritmu.

Příklad kódu pro soustavu o čtyřech neznámých bude zápis po transformaci do programu TKSL vypadat následovně :

```
q1=x+2*y+3*z+u-2;
q2=y+2*z+u-1;
q3=2*x+y+z;
q4=3*x+2*y+4*z+3*u-1;
x'=-(q1+2*q3+3*q4) &0;
y'=-(2*q1+q2+q3+2*q4) &0;
z'=-(3*q1+2*q2+q3+4*q4) &0;
u'=-(q1+q2+3*q4 &0;
```

6.2 Uživatelské rozhraní

Hlavní výhodou rozhraní TKSL/C – SLAR je ve vložení číslic a rychlé vygenerování kódu programem, který vytvoří soustavu elementárním převodem nebo zapíše rovnice pomocí transformačního algoritmu, přičemž sám vytvoří transponovanou matici.

6.2.1 Požadavky

Požadavky pro grafické uživatelské rozhraní pro výpočet SLAR elementární nebo transformační metodou byly stanoveny takto:

- bude spustitelné v operačním systému MS Windows,
- uživatel může jako vstup zadávat reálná čísla určující hodnoty proměnných v jednotlivých rovnicích soustavy, může zadat přesnost zobrazovaných čísel,
- převede zadaná vstupní čísla elementární nebo transformační metodou,
- nabízí možnost načítat vstupní soubor se SLAR,
- poskytuje možnost změny parametrů pro výpočet,
- zobrazuje výsledky podle zadaného přepínače,
- výsledky zobrazí v textovém Editu a nabízí možnost uložení do souboru.

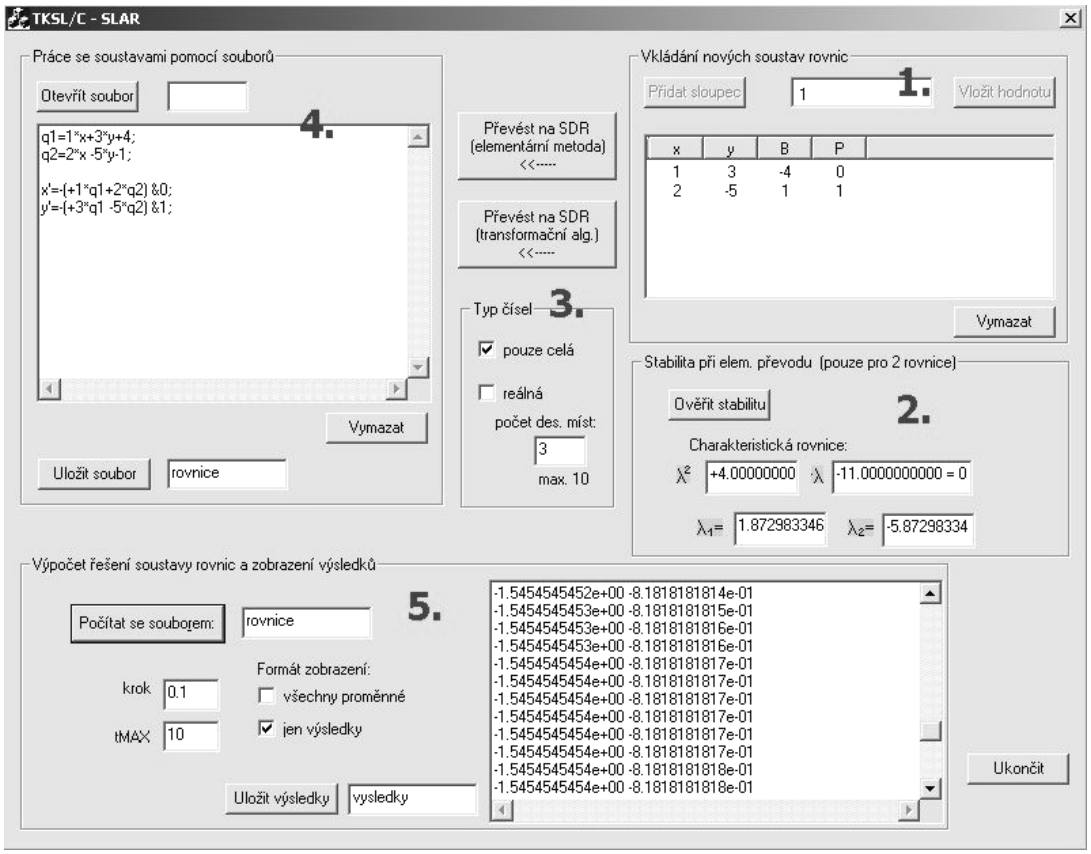
6.2.2 Vzhled programu

Okno programu je základním a jediných editačním prvkem aplikace, kde je možné provádět veškeré operace týkající se vstupů, výpočtů a výstupů programu. Aplikace, zobrazena na Obrázku 21, komunikuje s uživatelem v českém jazyce. Okno je rozděleno na pět částí, hodnoty a parametry jsou předávány tak, aby uživateli umožnily co nejrychlejší výpočet dané soustavy.

První část v pravém horním rohu aplikace slouží k zadávání čísel jako hodnot do rovnic. Umístění v pravém horním rohu bylo vybráno záměrně, protože po seznámení s elementární a transformační metodou si uživatel jistě oblíbí načítání souborů a přepisování jednotlivých hodnot než vkládání celých soustav. Uživatel má možnost přidat řádky a vytvořit až šest rovnic o šesti neznámých.

Zajímavé bylo rychle ověřit, zda-li je zadaná soustava stabilní i elementárním převodem, jak jsme se dozvěděli v kapitole 4.3. Proto byla vytvořena sekce 2, kde lze ověřit stabilitu soustavy o dvou rovnicích.

Třetí část programu obsahuje tlačítka pro jednotlivé převody a uživatel má možnost před vkládáním hodnot do rovnic určit, s jakými čísly chce pracovat a na kolik desetinných míst se mají čísla při vkládání a převodu zaokrouhlovat.



Obrázek 21: Rozhraní TKSL/C - SLAR

V levém horním rohu je umístěno zobrazení původně zadané SLAR v sekci 1 po převodu na SDR. Uživatel má možnost místo vkládání hodnot načíst předem uložený soubor, zobrazit jej a případně ručně upravit. Před samotným počítáním řešení je nutno soubor uložit. K tomu slouží další tlačítko v této části spolu se zadáním názvu pro uložený soubor.

Poslední, páté umístění je věnováno samotnému počítání výsledků. Uživatel musí zadat název souboru, kde je zapsán zdrojový text, a vybrat parametry k počítání. Implicitně jsou mu nabídnuty běžně používané hodnoty, takže po stisknutí tlačítka pro počítání, pošle programu TKSL/C příkaz k vyhodnocení dané soustavy s parametry. Výsledky jsou automaticky uloženy do souboru „Vysl“, který je zobrazen v editační části. Uživatel má možnost uložit si tyto výsledky do vlastního souboru, který si logicky pojmenuje.

7 Závěr

Cílem této práce bylo seznámení s metodami řešení soustav lineárních rovnic, zpracování metod převodu výpočtu soustav lineárních rovnic na výpočet diferenciálních rovnic a analýza tohoto algoritmu vzhledem ke klasickým metodám řešení. Největší pozornost byla věnována metodám převodu, zjišťování závislostí a chování soustav při změnách parametrů. Experimentálně byly sledovány změny v řešení a zaznamenávány výsledky kritických průběhů. Pomocí zjištěných poznatků bylo vytvořeno rozhraní k programu TKSL, které umožňuje jiným uživatelům bez nutnosti vytváření zdrojových textů pro metody převodu přiblížit pohled na danou problematiku. Toto rozšíření zkracuje dobu přípravy dat a od uživatele nevyžaduje hlubší znalost systému. Vytvořené rozhraní je vhodné pro využití ve výuce Moderní aplikace počítačů nebo Vysoce náročné výpočty na Fakultě informačních technologií VUT v Brně a v navazujícím studiu se předpokládá jeho další rozšíření.

Popsaná experimentální metoda transformačního algoritmu je vzhledem ke známým metodám řešení nová a odstraňuje většinu problémů, které mohou nastat při řešení klasicky. Největší výhodou je rychlost výpočtu soustav o mnoha rovnicích a jejich konvergence ke správnému výsledku. Metoda případné problémy nejen detekuje, ale nabízí možnosti odstranění. V oblasti převodu soustav a vlastnostech řešení je však stále mnoho nepopsaných souvislostí, např. závislosti konvergence výsledku na parametrech soustavy, stabilita řešení a urychlování konvergence.

Pro řešení soustav jsou díky moderní výpočetní technice otevřeny možnosti paralelního počítání a tím urychlování výpočtů soustav obsahujících až stovky rovnic. V souvislosti uvedených příkladů z elektrotechniky by řešení soustav o mnoha rovnicích mohlo najít propojení s hardwarovým vybavením, např. Petriho sítí. S tématem převodu SLAR na SDR a celou přilehlou problematikou bych ráda pokračovala v navazujícím doktorandském studiu.

Literatura

[1] Kubíček, M., Dubcová, M., Janovská, D., Numerické metody a algoritmy. Praha, VŠCHT Praha, 2005. Dokument dostupný na URL http://vydavatelstvi.vscht.cz/knihy/uid_isbn-80-7080-558-7/pages-img/070.html (duben 2007).

[2] Černá, R., Majlingová, O., Vogel, J., Základy algoritmizace a programování. Praha, ČVUT Praha. Dokument dostupný na URL http://www.fsid.cvut.cz/cz/U201/zapg.html (květen 2007).

[3] Limpouch, J., Numerické metody lineární algebry. Praha, ČVUT Praha, 2000. Dokument dostupný na URL http://kfe.fjfi.cvut.cz/~limpouch/numet/linalg/linalg.html (květen 2007).

[4] Mošová, V., Numerické metody. Olomouc, Univerzita Palackého v Olomouci. Dokument dostupný na URL http://www.inf.upol.cz/skripta/texty/numericke%20metody.pdf (květen 2007).

[5] Přispěvatelé Wikipedie, Wikipedie: Otevřená encyklopedie, c2007. Dokument dostupný na URL http://cs.wikipedia.org/wiki/ (květen 2007).

[6] Sehnalová, P., Řešení soustav lineárních algebraických rovnic [ročníkový projekt], VUT Brno, 2005.

Seznam příloh

Příloha A

Obsah přiloženého média

Na disku CD-ROM naleznete v adresářích elektronický dokument práce, popisovaný systém se zdrojovými texty a ukázkami příkladů a vstupní soubory pro programy TKSL386. Adresář SLAR je nutno celý nakopírovat na místo v počítači. Zdrojové soubory jsou spustitelné pouze z adresáře SLAR\Files\. Přehled obsahu jednotlivých adresářů:

vase_umisteni: \readme.txt	stručný popis obsahu média
vase_umisteni: \zprava\	elektronická forma této práce spolu s použitými obrázky
vase_umisteni: \SLAR\	spustitelný program slar.exe spolu s nutnou součástí cltksl.exe a demonstračním příkladem
vase_umisteni: \tksl\	zdrojové vstupní soubory použité v práci pro program TKSL386

Příloha B

Uživatelská příručka TKSL/C - SLAR

B.1 Výpočet zadáním SLAR do editačního pole

1. Spusťte aplikaci slar.exe.
2. Podle toho, s jakými čísly si přejete pracovat, zaškrtněte příslušné políčko: Pouze reálná nebo políčko Pouze celá. U počítání s reálnými čísly lze zadat přesnost desetinných míst, se kterou se budou hodnoty ukládat.
3. V pravém horním rohu jsou jako sloupce editačního pole popsány jednotlivé proměnné rovnic. Proměnné na levé straně jsou postupně označeny jako x, y. Po kliknutí na tlačítko Přidat sloupec, lze vytvořit soustavu o šesti rovnicích. Další proměnné jsou z, u, v a w. Následující sloupec označený B obsahuje hodnoty pravých stran rovnic. Poslední sloupec P určuje počáteční podmínky výpočtu.
4. U soustavy rovnic o dvou neznámých máte možnost ověřit, zda řešení této soustavy pomocí elementárního převodu vyjde v ustáleném tvaru. Ověření provedete stiskem tlačítka Ověřit stabilitu.
5. Následuje proces převodu, zvolte elementární převod nebo převod pomocí transformačního algoritmu. Z pravé strany editačního pole se SLAR upraví do SDR na levé straně v editačním poli.
6. Následuje část B.3.

B.2 Výpočet pomocí vstupního souboru

1. Otevřeme existující soubor, který musí být uložen ve složce vase_umisteni: \SLAR\Files\ Obsah souboru se zobrazí v editačním poli v levém horním rohu aplikace. Soubor můžeme přepsat nebo jej upravit.
2. Soubor, který jsme změnili nebo nově vytvořili, je nutné před spuštěním výpočtu uložit kliknutím na tlačítko Uložit soubor. Pokud již soubor stejného názvu existuje, jste tázáni, zda si jej přejete přepsat. Aplikace nabízí možnost uložit soubor pod jiným názvem.
3. Pro výpočet následuje část B.3.

B.3 Výpočet SDR

1. Před samotným výpočtem je nutno zadat název souboru vedle tlačítka Počítat se souborem.
2. Pokud chcete změnit parametry pro krok a maximální čas výpočtu, změňte hodnoty v příslušných polích krok a tMAX.
3. Aplikace nabízí vypočítání a zobrazení dvou formátů výstupu. Zvolte možnost Všechny proměnné, pokud chcete zkoumat čas, pomocné proměnné i výsledky. Pokud si přejete získat jen řešení soustavy zatrhněte možnost Jen výsledky.
4. Po stisknutí tlačítka Počítat se souborem je spuštěno další okno s programem cltksl.exe, který se nachází ve složce vase_umisteni: \SLAR\Files\. Při rychlém výpočtu a správné konfiguraci vstupního souboru se samo ukončí, aplikace oznámí konec výpočtu a v editační části se zobrazí výsledky. Mohou nastat případy, že se okno programu cltksl samo neukončí. Jedná se o příliš dlouhý zadaný čas tMAX, v tomto případě programu trvá než dokončí výpočet a okno uzavře. Nebo došlo k zacyklení v důsledku zadání špatně podmíněné soustavy nebo soustavy, která nekonverguje. V obou případech je možné okno uzavřít, ale výsledky nebudou korektní.
5. Soubor s výsledky je automaticky uložen do výstupního souboru s názvem Vysl do složky spolu se vstupnímu soubory. Aplikace nabízí možnost uložení do souboru s jiným názvem, který zadáte.
6. Po ukončení výpočtů okno aplikace zavřete tlačítkem Zavřít.