

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INTELIGENTNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INTELLIGENT SYSTEMS

PARALELNÍ NUMERICKÉ ŘEŠENÍ PARCIÁLNÍCH DIFERENCIÁLNÍCH ROVNIC

DIPLOMOVÁ PRÁCE

MASTER'S THESIS

AUTOR PRÁCE

AUTHOR

Bc. GABRIELA NEČASOVÁ

BRNO 2014



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INTELIGENTNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INTELLIGENT SYSTEMS

PARALELNÍ NUMERICKÉ ŘEŠENÍ PARCIÁLNÍCH DIFERENCIÁLNÍCH ROVNIC

PARTIAL DIFFERENTIAL EQUATIONS PARALLEL SOLUTIONS

DIPLOMOVÁ PRÁCE

MASTER'S THESIS

AUTOR PRÁCE

AUTHOR

Bc. GABRIELA NEČASOVÁ

VEDOUCÍ PRÁCE

SUPERVISOR

doc. Ing. JIŘÍ KUNOVSKÝ, CSc.

BRNO 2014

Abstrakt

Práce se zabývá tématem paralelního numerického řešení parciálních diferenciálních rovnic. Práce se nejprve zaměřuje na obyčejné parciální diferenciální rovnice (ODR) a jejich metody řešení pomocí Taylorova polynomu. Další část je věnována parciálním diferenciálním rovnicím (PDR). Jsou zde popsány typy PDR, jedná se o parabolické, hyperbolické a eliptické PDR. Také je vysvětleno, jakým způsobem používat systém TKSL při výpočtu PDR. Další část práce je zaměřena na metody řešení PDR, mezi tyto metody patří dopředná, zpětná a kombinovaná metoda. Bylo vysvětleno, jakým způsobem lze tyto metody řešit v systémech TKSL a Matlab. Dále je diskutována přesnost a časová náročnost výpočtu. Další součástí je paralelní řešení PDR. Díky možnosti převodu PDR na soustavu ODR lze jednotlivé rovnice reprezentovat nezávislými operačními jednotkami, které umožňují paralelní výpočet. Poslední kapitola je věnována implementaci. Aplikace umožňuje vygenerovat soustavy ODR pro systém TKSL, které reprezentují zadanou hyperbolickou PDR.

Abstract

This thesis deals with the topic of partial differential equations parallel solutions. First, it focuses on ordinary differential equations (ODE) and their solution methods using Taylor polynomial. Another part is devoted to partial differential equations (PDE). There are several types of PDE, there are parabolic, hyperbolic and elliptic PDE. There is also explained how to use TKSL system for PDE computing. Another part focuses on solution methods of PDE, these methods are forward, backward and combined methods. There was explained, how to solve these methods in TKSL and Matlab systems. Computing accuracy and time complexity are also discussed. Another part of thesis is PDE parallel solutions. Thanks to the possibility of PDE conversion to ODE systems it is possible to represent each ODE equation by independent operation unit. These units enable parallel computing. The last chapter is devoted to implementation. Application enables generation of ODE systems for TKSL system. These ODE systems represent given hyperbolic PDE.

Klíčová slova

Parciální diferenciální rovnice, obyčejné diferenciální rovnice, Taylorův polynom, Taylorova řada, RC článek, TKSL, dopředná metoda, zpětná metoda, kombinovaná metoda.

Keywords

Partial differential equations, ordinary differential equations, Taylor polynomial, Taylor series, RC circuit, TKSL, forward method, backward method, combined method.

Citace

Gabriela Nečasová: Paralelní numerické řešení parciálních diferenciálních rovnic, diplomová práce, Brno, FIT VUT v Brně, 2014

Paralelní numerické řešení parciálních diferenciálních rovnic

Prohlášení

Prohlašuji, že jsem tuto diplomovou práci vypracovala samostatně, pod vedením pana doc. Ing. Jiřího Kunovského, CSc.

Uvedla jsem všechny literární prameny a publikace, ze kterých jsem čerpala.

.....

Gabriela Nečasová

26. května 2014

Poděkování

Velice děkuji panu doc. Ing. Jiřímu Kunovskému, CSc., za poskytnuté rady a podnětné připomínky při řešení této diplomové práce.

© Gabriela Nečasová, 2014.

Tato práce vznikla jako školní dílo na Vysokém učení technickém v Brně, Fakultě informačních technologií. Práce je chráněna autorským zákonem a její užití bez udělení oprávnění autorem je nezákonné, s výjimkou zákonem definovaných případů.

Obsah

1	Úvod	5
2	Obyčejné diferenciální rovnice	7
2.1	Metody řešení diferenciálních rovnic	7
2.2	ODR řešené pomocí Taylorovy řady	8
2.2.1	RC článek	8
2.2.2	Příčkový RC článek	11
2.3	Experimentální výpočty	15
2.3.1	Systém TKSL a proměnný integrační krok	16
3	Parciální diferenciální rovnice	25
3.1	Typy parciálních diferenciálních rovnic	25
3.1.1	Parabolická PDR	26
3.1.2	Hyperbolická PDR	26
3.1.3	Eliptická PDR	26
3.2	Převod PDR na ODR	27
3.2.1	Metoda přímek	27
4	Parciální diferenciální rovnice v systému TKSL	29
4.1	n-bodové aproximace	29
4.2	Hyperbolická PDR	30
4.3	Parabolická PDR	32
4.4	Eliptická PDR	33
5	Metody řešení PDR	35
5.1	Dopředná metoda	37
5.2	Zpětná metoda	40
5.3	Kombinovaná metoda	43
5.4	Metody řešení PDR v systému Matlab	46
5.5	Metody řešení PDR v systému TKSL	50
5.5.1	Dopředná metoda v systému TKSL	50
5.5.2	Zpětná metoda v systému TKSL	52
5.5.3	Kombinovaná metoda v systému TKSL	55
5.6	Přesnost výpočtu	57
5.7	Časová náročnost výpočtů	68
6	Paralelní řešení PDR	70

7 Implementace	78
8 Závěr	82
Přílohy	84
A Obsah přiloženého CD	85

Seznam obrázků

2.1	RC článek	8
2.2	RC obvod	9
2.3	Příčkový RC článek	11
2.4	Graf závislosti napětí na čase na kapacitách	12
2.5	Zdrojový kód pro příčkový RC článek o pěti kapacitách	16
2.6	Graf znázorňující řád ORD pro čas $t = 0, 1$	18
2.7	Graf znázorňující řád ORD pro čas $t = 0, 3$	18
2.8	Graf znázorňující řád ORD pro čas $t = 0, 4$	19
2.9	Zdrojový kód, integrační krok $DT = 1$	19
2.10	Graf znázorňující hodnotu řádu ORD v závislosti na čase t ($DT = 1$)	20
2.11	Graf znázorňující hodnotu řádu ORD v čase $t = 2$ ($DT = 1$)	20
2.12	Zdrojový kód, integrační krok $DT = 2$	21
2.13	Graf znázorňující hodnotu řádu ORD v čase $t = 2$ ($DT = 2$)	21
2.14	Graf znázorňující hodnotu řádu ORD v čase $t = 4$ ($DT = 2$)	22
2.15	Graf znázorňující hodnotu řádu ORD v závislosti na čase t pro proměnnou U_{CA}	22
2.16	Graf znázorňující hodnotu řádu ORD v závislosti na čase t pro proměnnou U_{CB}	23
2.17	Graf znázorňující hodnotu řádu ORD v závislosti na čase t pro proměnnou U_{CC}	23
2.18	Graf znázorňující hodnotu řádu ORD v závislosti na čase t pro proměnnou U_{CD}	24
2.19	Graf znázorňující hodnotu řádu ORD v závislosti na čase t pro proměnnou U_{CE}	24
3.1	Princip metody přímek	28
4.1	Pětibodová nesymetrická aproximace – bod u_1	29
4.2	Pětibodová nesymetrická aproximace – bod u_8	30
4.3	Kmitající struna – aproximace pro bod u_2	30
4.4	Tříbodová aproximace – bod u_1	32
4.5	Tříbodová aproximace – bod u_2	32
4.6	Tříbodová aproximace – bod u_3	32
4.7	Náhrada derivací ve směru osy x i y	33
4.8	Náhrada derivací pro bod A_{22}	34
5.1	Výpočet pomocí klasické TŘ	35
5.2	Výpočet pomocí TŘ pro pětibodovou aproximaci	35
5.3	Dopředná metoda – bod u_0	39

5.4	Dopředná metoda – bod u_1	39
5.5	Dopředná metoda – bod u_2	39
5.6	Dopředná metoda – nedostatek bodů pro další výpočet	40
5.7	Dopředná metoda – další bod u_{11}	40
5.8	Zpětná metoda – bod u_{10}	42
5.9	Zpětná metoda – bod u_9	42
5.10	Zpětná metoda – bod u_8	42
5.11	Kombinovaná metoda – bod u_0	44
5.12	Kombinovaná metoda – bod u_1	44
5.13	Kombinovaná metoda – bod u_2	44
5.14	Kombinovaná metoda – bod u_9	44
5.15	Vysvětlení pojmu <i>okno</i>	48
5.16	Pětibodová aproximace – bod u_1	50
5.17	Pětibodová aproximace – bod u_2	50
5.18	Pětibodová aproximace – bod u_{11}	53
5.19	Pětibodová aproximace – bod u_{10}	53
5.20	Pětibodová aproximace, krok $h = 0,7$	58
5.21	Pětibodová aproximace, krok $h = 0,6$	58
5.22	Pětibodová aproximace, krok $h = 0,5$	59
5.23	Pětibodová aproximace, krok $h = 0,4$	59
5.24	Pětibodová aproximace, krok $h = 0,3$	60
5.25	Pětibodová aproximace, krok $h = 0,2$	60
5.26	Pětibodová aproximace, krok $h = 0,1$	61
5.27	Chyba výpočtu Dopředné metody – $n = 5, h = 0.1$	62
5.28	Chyba výpočtu Dopředné metody – $n = 5, h = 0.01$	62
5.29	Chyba výpočtu Dopředné metody – $n = 5, h = 0.001$	63
5.30	Chyba výpočtu Dopředné metody – $n = 3, h = 0.01$	63
5.31	Chyba výpočtu Dopředné metody – $n = 7, h = 0.01$	64
5.32	Chyba výpočtu Dopředné metody – $n = 9, h = 0.01$	64
5.33	Chyba výpočtu Kombinované metody – $n = 5, h = 0.1$	65
5.34	Chyba výpočtu Kombinované metody – $n = 5, h = 0.01$	65
5.35	Chyba výpočtu Kombinované metody – $n = 3, h = 0.1$	66
5.36	Chyba výpočtu Kombinované metody – $n = 3, h = 0.01$	66
5.37	Chyba výpočtu Kombinované metody – $n = 7, h = 0.01$	67
5.38	Chyba výpočtu Kombinované metody – $n = 9, h = 0.01$	67
6.1	Integrátor	71
6.2	Sumátor	71
6.3	Invertor	71
6.4	Konstanta	71
6.5	Sériové zapojení integrátorů	72
6.6	Programové schéma pro příklad 6.1 — rovnice $v1'$	76
6.7	Celkové programové schéma pro příklad 6.1	76
6.8	Programové schéma pro příklad 6.2 — rovnice $v1'$	77
6.9	Celkové programové schéma pro příklad 6.2	77
7.1	Grafické uživatelské rozhraní	79
7.2	Sekvenční diagram činnosti aplikace	80

Kapitola 1

Úvod

Parciální diferenciální rovnice (PDR) mají velký význam v různých oblastech. Tyto rovnice nabízí uplatnění v mnoha průmyslových oblastech jako jsou například jaderné reaktory, vesmírné projekty, řízení dopravy, výroba a distribuce elektrické energie, atd. Soustava PDR poté slouží jako matematický model, který zachycuje podstatné vlastnosti vytvořených systémů za účelem jejich popisu, případě předpovědi jejich chování a řízení jejich časového vývoje. Cílem diplomové práce je seznámit se s paralelním numerickým řešením parciálních diferenciálních rovnic (PDR).

Kapitola 2 slouží k uvedení do problematiky obyčejných diferenciálních rovnic (ODR). Podkapitola 2.1 nás stručně seznamuje s možnými metodami řešení ODR. Další podkapitola 2.2 se zaměřuje zejména na metody jejich řešení pomocí Taylorovy řady. Dále je zde popsáno, jak lze využít RC příčkový článek k výpočtům členů Taylorova polynomu. Podkapitola 2.3 se zaměřuje na volitelný řád numerické integrační metody.

Kapitola 3 pojednává o typech PDR a jejich metodách řešení. Podkapitola 3.1 se zaměřuje na parabolické, hyperbolické a eliptické PDR. V další části (podkapitola 3.2) je nastíněn způsob převodu PDR na ODR.

Kapitola 4 je zaměřena na řešení PDR v systému TKSL. Systém TKSL dokáže řešit pouze ODR. Proto jsou zde vysvětleny principy, jak se jednotlivé typy PDR převádí na soustavu ODR. K tomuto účelu se využívají n -bodové aproximace (viz podkapitola 4.1). Další podkapitoly 4.2, 4.3 a 4.4 se věnují převodu hyperbolických, parabolických a eliptických PDR na soustavy ODR.

V Kapitole 5 jsou uvedeny různé metody řešení PDR, jedná se o dopřednou, zpětnou a kombinovanou metodu (viz podkapitoly 5.1, 5.2, 5.3). Všechny tyto metody jsou spjaty s Taylorovou řadou a je vysvětleno, jakým způsobem je možné řešit PDR právě pomocí Taylorovy řady. V textu se konkrétně zaměřujeme na hyperbolickou PDR, která popisuje kmitání struny. V další části (podkapitola 5.4) je vysvětleno, jakým způsobem lze řešit PDR v systému Matlab, výpočty byly opět inspirovány Taylorovou řadou. Následně se v podkapitole 5.5 seznamujeme s řešením PDR v systému TKSL pro jednotlivé metody. Další podkapitola (5.6) se věnuje přesnosti výpočtu. Jsou zde analyzovány faktory, které významným způsobem ovlivňují přesnost výpočtu, a tedy i kvalitu výsledků. Poslední část této kapitoly je zaměřena na časovou náročnost výpočtu, jsou zde srovnány systémy Matlab a TKSL (viz podkapitola 5.7).

Kapitola 6 se zaměřuje na paralelní řešení PDR. Je zde ukázáno, jakým způsobem a pomocí jakých komponent vytvořit příslušné programové schéma pro libovolné parametry výpočtu (tedy pro libovolný počet řezů na struně a pro libovolnou n -bodovou aproximaci). Také je diskutováno zrychlení výpočtu, které paralelní řešení přináší.

Kapitola 7 se věnuje implementaci grafického uživatelského rozhraní, které je schopno generovat soustavu ODR, která reprezentuje převáděnou hyperbolickou PDR. Dále je zde vysvětlen princip výpočtu a spojitosti jednotlivých částí aplikace, které se podílejí na tvorbě výsledků.

V závěru práce (kapitola 8) jsou zhodnoceny dosažené výsledky a je nastíněna možnost dalšího vývoje.

Kapitola 2

Obyčejné diferenciální rovnice

Nejprve se seznámíme s pojmem diferenciální rovnice. Diferenciální rovnice (dále DR) je rovnice, kde neznámou je funkce a rovnice obsahuje i derivaci této funkce. Na začátek uvádíme definici 2.3. Podrobnější informace jsou uvedeny v [6, 3, 11, 10, 8, 7]. Diferenciální rovnice můžeme rozdělit podle derivací následovně:

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

Rovnice uvedené v definici 2.3, se nazývají *obyčejné diferenciální rovnice*.

Definice 2.1. *Rovnice ve tvaru $F(y^n, y^{n-1}, \dots, y', y, x) = 0$ se nazývá **diferenciální rovnice n -tého řádu** pro funkci $y = y(t)$. Speciálně je*

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

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

Rovnice uvedené v definici 3.1 se nazývají *parciální diferenciální rovnice*. Abychom získali jednoznačné řešení soustavy, je nutné specifikovat pro každou rovnici počáteční podmínku.

Definice 2.2. *Nechť $x_0, y_0 \in \mathbb{R}$. Úloha najít řešení DR, které splňuje tzv. **počáteční podmínku***

$$y(x_0) = y_0,$$

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

2.1 Metody řešení diferenciálních rovnic

Diferenciální rovnice lze řešit:

- analyticky,
- numericky.

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

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

Metody lze rozdělit na

- jednokrokové,
- vícekrokové.

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

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

2.2 ODR řešené pomocí Taylorovy řady

Nejdříve uvedeme definice pro Taylorovu řadu 2.3 a Taylorův polynom 3.1.

Definice 2.3. *Nechť f je funkce a a nějaký bod z vnitřku definičního oboru. Předpokládejme, že f má derivaci všech vyšších řádů v a . Pak definujeme **Taylorovu řadu** v a vzorcem*

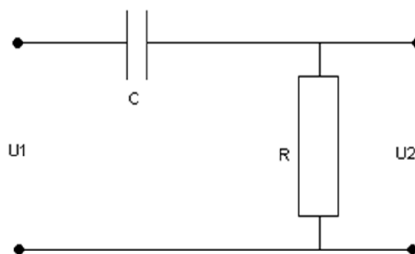
$$T(x) = \sum_{k=0}^{\infty} \frac{f^{(k)}(a)}{k!} (x - a)^k.$$

Definice 2.4. *Nechť funkce f má všechny derivace až po řád n v a . Pak definujeme **Taylorův polynom** f stupně n se středem a jako*

$$T_n(x) = f(a) + f'(a)(x - a) + \frac{1}{2!} f''(a)(x - a)^2 + \dots + \frac{1}{n!} f^{(n)}(a)(x - a)^n = \sum_{k=0}^n \frac{f^{(k)}(a)}{k!} (x - a)^k.$$

2.2.1 RC článek

Při výpočtu členů Taylorova polynomu lze využít derivační RC článek (viz obrázek 2.1). Tento obvod vznikne vytvořením děliče napětí jednoduchým spojením rezistoru a kondenzátoru. Derivační RC článek má vlastnost hornofrekvenční propusti (horní propust). To tedy znamená, že derivačním článkem projdou pouze složky časově proměnného elektrického proudu, které mají frekvenci vyšší než je určitá mezní frekvence.



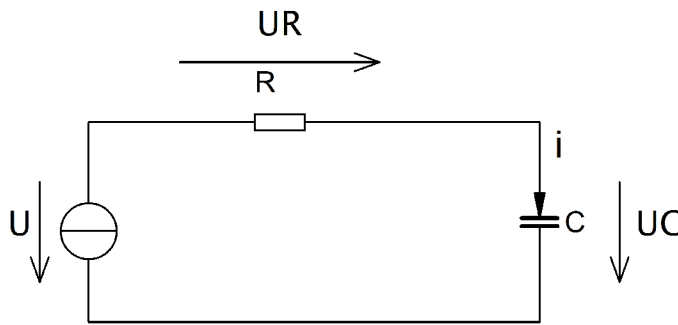
Obrázek 2.1: RC článek

Při přechodu elektrického obvodu z jednoho energetického stavu do druhého dochází v elektrickém obvodu k přechodnému ději. Přechodné děje nastávají při náhlých změnách v elektrickém obvodu, například připojení či odpojení zdrojů elektrického napětí, zkraty, odpojení některé z větví nebo náhlé změny parametrů některých prvků, atd. Prvky v elektrických obvodech můžeme rozdělit následovně:

- nesetrvačné prvky,
- setrvačné prvky.

U nesetrvačných prvků jakákoliv změna napětí vyvolá okamžitou změnu elektrického proudu. Mezi tento typ prvků patří rezistory. Naopak setrvačné prvky akumulují energii a časové průběhy jsou spojité. Patří sem kapacitory (časové průběhy na U a C jsou spojité) a také indukctory (časový průběh I na L je spojitý).

Příklad 2.1. Mějme následující RC obvod (viz obrázek 2.2). V obvodech, kde se vyskytují kapacity, sestavujeme diferenciální rovnice pro napětí na kapacitě. Chceme zjistit vztah pro napětí u'_C .



Obrázek 2.2: RC obvod

Výchozí vztah pro výpočet bude následující.

$$u'_C = \frac{1}{C}i \quad (2.1)$$

Vyjádříme proud i .

$$u = u_R + u_C = Ri + u_C i = \frac{u - u_C}{R} \quad (2.2)$$

Nyní dosadíme do výchozího vztahu $u'_C = \frac{1}{C}i$ a dostáváme níže uvedenou rovnici.

$$u'_C = \frac{1}{RC}(u - u_C) \quad u_{C(0)} = 0 \quad (2.3)$$

□

Nyní si ukážeme, jak můžeme vypočítat členy Taylorovy řady efektivně. Pro ilustraci využijeme následující příklad 2.2.

Příklad 2.2. Uvažujme jednoduchou diferenciální rovnici

$$y' = y \quad y(0) = y_0 \quad (2.4)$$

Naším úkolem je sestavit Taylorovu řadu pro bod $y(1)$ a vyjádřit prvních pět členů Taylorovy řady.

Taylorova řada pro bod $y(1)$ je uvedena níže, vycházíme z definice 2.3.

$$y_{(1)} = y_{(0)} + hy'_{(0)} + \frac{h^2}{2!}y''_{(0)} + \frac{h^3}{3!}y'''_{(0)} + \frac{h^4}{4!}y''''_{(0)} \quad (2.5)$$

Nyní postupujeme následovně. První člen Taylorovy řady je $y_{(0)}$. Druhý člen je $hy'_{(0)}$. V tomto okamžiku ihned dosadíme ze zadání. Tím se zbavíme derivace a tento člen označíme jako DY1.

$$DY1 = hy'_{(0)} = hy_0 \quad (2.6)$$

Poznamenejme, že dosazovat ihned do zadání lze z toho důvodu, že platí následující vztahy.

$$y' = y \quad (2.7)$$

$$y'' = y' \quad (2.8)$$

$$y''' = y'' \quad (2.9)$$

$$y'''' = y''' \quad (2.10)$$

Ve druhém členu Taylorovy řady vidíme, že se zde již vyskytuje faktoriál, umocňování a také druhá derivace. Ukážeme si ovšem, že druhý člen lze vypočítat pouze použitím operací násobení a dělení. Druhý člen Taylorovy řady označíme jako DY2. Pro výpočet členu DY2 postačí, pokud předchozí člen DY1 vynásobíme zlomkem $\frac{h}{2}$:

$$DY2 = \frac{h^2}{2!}y''_{(0)} \quad (2.11)$$

$$DY2 = \frac{h}{2}DY1 \quad (2.12)$$

$$DY2 = \frac{h}{2}hy_{(0)} \quad (2.13)$$

Podíváme se na rovnici 2.13 a zkontrolujeme, zda opravdu koresponduje s rovnicí 2.11. V čitateli zlomku získáváme skutečně h^2 . Ve jmenovateli najdeme číslo 2, což odpovídá, protože $2! = 2 \cdot 1 = 2$. Konečně, celý zlomek je násoben y_0 , což odpovídá y'' . Třetí člen Taylorovy řady označíme jako DY3. Postupujeme obdobným způsobem a vycházíme z předchozího členu DY2.

$$DY3 = \frac{h^3}{3!}y'''_{(0)} \quad (2.14)$$

$$DY3 = \frac{h}{3}DY2 \quad (2.15)$$

$$DY3 = \frac{h}{3} \left(\frac{h}{2}hy_{(0)} \right) \quad (2.16)$$

Opět je zřejmé, že rovnice 2.16 odpovídá rovnici 2.14. Čitatel obsahuje člen h^3 , jmenovatel pak $3! = 3 \cdot 2 \cdot 1 = 6$. Celý zlomek je násoben y_0 , což odpovídá y''' . Poslední člen Taylorovy řady vyjádříme způsobem uvedeným níže.

$$DY4 = \frac{h^4}{4!} y_{(0)}''' \quad (2.17)$$

$$DY4 = \frac{h}{4} DY3 \quad (2.18)$$

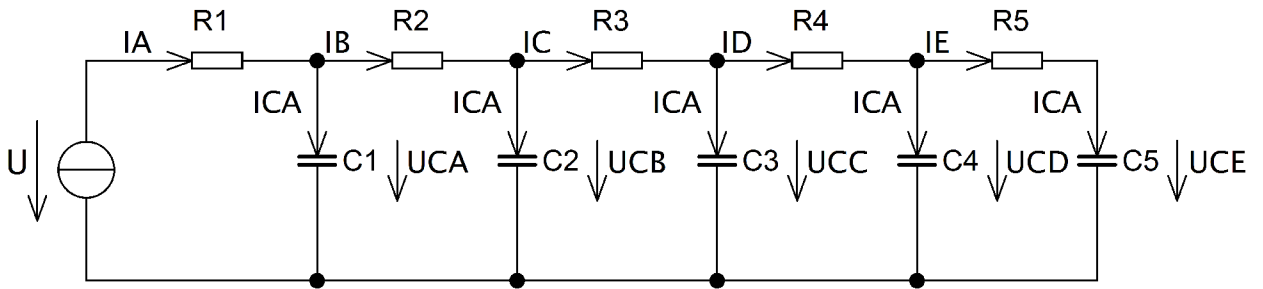
$$DY4 = \frac{h}{4} \left(\frac{h}{3} \left(\frac{h}{2} h y_{(0)} \right) \right) \quad (2.19)$$

□

2.2.2 Příčkový RC článek

Příčkový RC článek je obvod, který je složený z rezistorů a kapacit. Příklad příčkového článku znázorňuje obrázek 2.3. Níže uvedené výpočty platí právě pro tento příčkový článek. Předpokládáme následující hodnoty.

$$\begin{aligned} U &= 1V \\ C &= 1F \\ R &= 1\Omega \\ h &= 0,1 \\ TMAX &= 10 \end{aligned}$$



Obrázek 2.3: Příčkový RC článek

Kapacity v obvodu se postupně nabíjí. Nejdříve se nabije kapacita C1, poté C2, C3 až C5. Průběh napětí v závislosti na čase může vypadat tak, jak je ukázáno v grafu 2.4. Dle prvního Kirchhoffova zákona o proudech (uzlech) platí rovnice 2.20 až 2.24 uvedené níže. Pro připomenutí poznamenáme, že první Kirchhoffův zákon říká, že v každém bodě (uzlu) elektrického obvodu platí, že součet proudů vstupujících do uzlu se rovná součtu proudů vystupujících z uzlu.

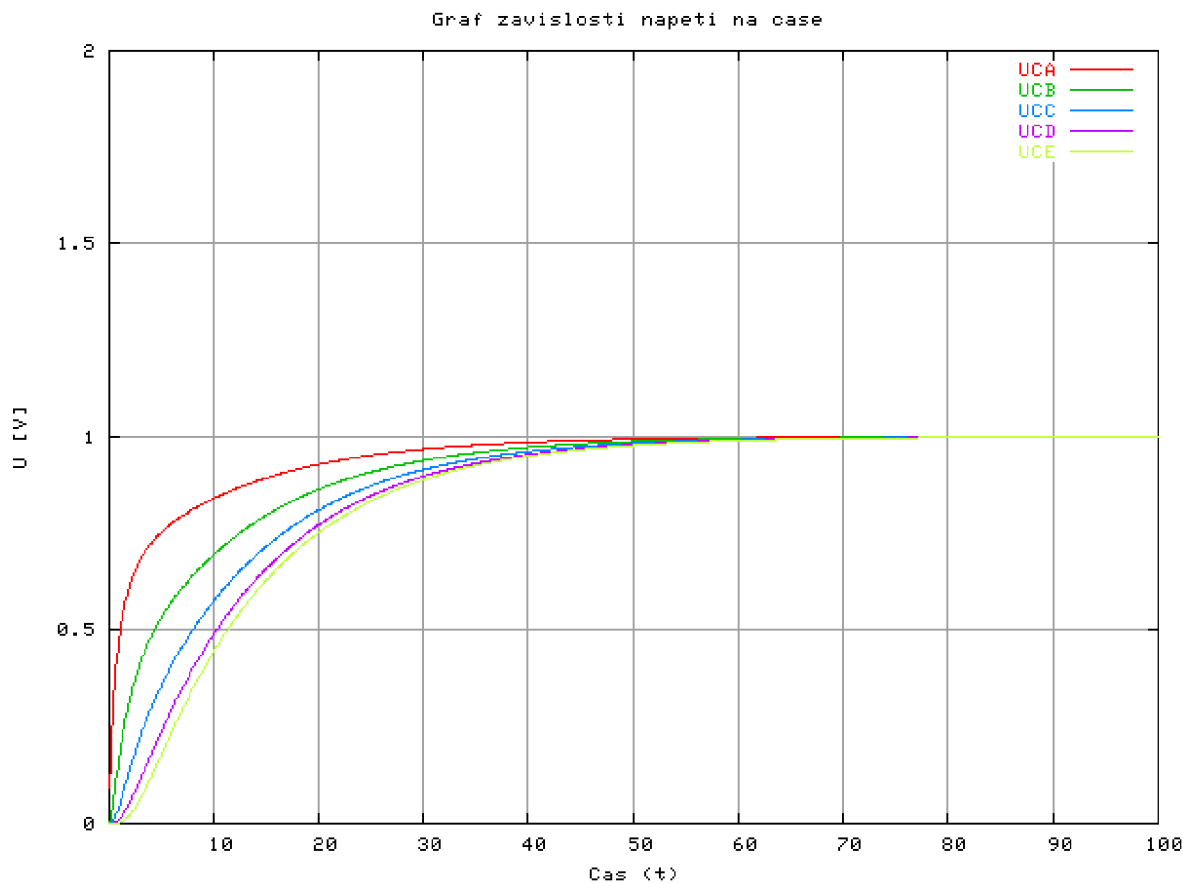
$$I_A = I_B + I_{CA} \quad (2.20)$$

$$I_B = I_C + I_{CB} \quad (2.21)$$

$$I_C = I_D + I_{CC} \quad (2.22)$$

$$I_D = I_E + I_{CD} \quad (2.23)$$

$$I_E = I_{CE} \quad (2.24)$$



Obrázek 2.4: Graf závislosti napětí na čase na kapacitách

Podle Ohmova zákona vyjádříme vztah pro proud (I).

$$U = R \cdot I \quad (2.25)$$

$$I = \frac{U}{R} \quad (2.26)$$

Nyní tedy přepíšeme rovnici pro uzel A.

$$I_A = I_B + I_{CA} \quad (2.27)$$

$$\frac{U - U_{CA}}{R} = \frac{U_{CA} - U_{CB}}{R} + I_{CA} \quad (2.28)$$

Pro napětí U_{CA} na kapacitě C_1 platí vztah znázorněný níže.

$$U'_{CA} = \frac{I}{C} I_{CA} \quad (2.29)$$

Víme, že platí následující rovnice.

$$I_{CA} = I_A - I_B \quad (2.30)$$

Můžeme tedy psát vztah uvedený níže.

$$I_{CA} = I_A - I_B \quad (2.31)$$

$$U'_{CA} = \frac{I}{C} (I_A - I_B) \quad (2.32)$$

Zlomek $\frac{1}{C}$ vytkneme a proudy I_A a I_B prepíšeme dle Ohmova zákona $I = \frac{U}{R}$.

$$U'_{CA} = \frac{I}{C} \left(\frac{U - U_{CA}}{R} - \frac{U_{CA} - U_{CB}}{R} \right) \quad (2.33)$$

Vytkneme ještě zlomek $\frac{1}{R}$ a získáme výsledný vztah pro napětí na kapacitě C_1 . Napětí $U'_{CA(0)}$ v bodě 0 je poté vyjádřeno rovnicí uvedenou níže.

$$U'_{CA(0)} = \frac{I}{C} \frac{1}{R} (U - 2U_{CA(0)} + U_{CB(0)}) \quad (2.34)$$

Obdobným způsobem můžeme vyjádřit i napětí pro ostatní kapacity C_2, C_3, C_4 a C_5 . Počáteční podmínky uvažujeme nulové.

$$U'_{CB(0)} = \frac{I}{C} \frac{1}{R} (U_{CA(0)} - 2U_{CB(0)} + U_{CC(0)}) \quad (2.35)$$

$$U'_{CC(0)} = \frac{I}{C} \frac{1}{R} (U_{CB(0)} - 2U_{CC(0)} + U_{CD(0)}) \quad (2.36)$$

$$U'_{CD(0)} = \frac{I}{C} \frac{1}{R} (U_{CC(0)} - 2U_{CD(0)} + U_{CE(0)}) \quad (2.37)$$

$$U'_{CE(0)} = \frac{I}{C} \frac{1}{R} (U_{CD(0)} - U_{CE(0)}) \quad (2.38)$$

Pro výpočet v bodě 0 jsme využili počáteční podmínky. Pro výpočet v dalších bodech již musíme využít Taylorovu řadu. Ukážeme sestavení Taylorovy řady pro $U_{CA(1)}$, což odpovídá simulačnímu času $t = h = 0, 1$.

$$U_{CA(1)} = U_{CA(0)} + h \cdot U'_{CA(0)} + \frac{h^2}{2!} \cdot U''_{CA(0)} + \frac{h^3}{3!} \cdot U'''_{CA(0)} + \frac{h^4}{4!} \cdot U''''_{CA(0)} \quad (2.39)$$

Na ukázkou vyjádříme prvních pět členů Taylorovy řady pro $U_{CA(1)}$. První člen Taylorovy řady je $U_{CA(0)}$. Ostatní členy označíme postupně jako $DUCA1_{(0)}, DUCA2_{(0)}, DUCA3_{(0)}, DUCA4_{(0)}$. Nejsou zde uvedeny zlomky $\frac{1}{C}, \frac{1}{R}$, které jsme mohli vidět v rovnicích 2.34 až 2.38. Je to tak proto, že uvažujeme hodnoty $R = 1\Omega$ a $C = 1F$.

$$DUCA1_{(0)} = h \cdot U'_{CA(0)} = h(U - 2U_{CA(0)} + U_{CB(0)}) \quad (2.40)$$

$$DUCA2_{(0)} = \frac{h^2}{2!} \cdot U''_{CA(0)} = \frac{h}{2} \cdot (0 - 2DUCA1_{(0)} + DUCA1_{(0)}) \quad (2.41)$$

$$DUCA3_{(0)} = \frac{h^3}{3!} \cdot U'''_{CA(0)} = \frac{h}{3} \cdot (0 - 2DUCA2_{(0)} + DUCA2_{(0)}) \quad (2.42)$$

$$DUCA4_{(0)} = \frac{h^4}{4!} \cdot U''''_{CA(0)} = \frac{h}{4} \cdot (0 - 2DUCA3_{(0)} + DUCA3_{(0)}) \quad (2.43)$$

Máme tedy Taylorovu řadu pro $U_{CA(1)}$. Máme pět uzových bodů, což znamená, že pro každý krok výpočtu sestavíme pět Taylorových řad. Je tedy zapotřebí vytvořit i další Taylorovy řady pro $U_{CB(1)}, U_{CC(1)}, U_{CD(1)}$ a $U_{CE(1)}$. V rámci výpočtu každé Taylorovy řady musíme počítat jednotlivé členy Taylorovy řady – členy $DUCXY$, kde X označuje uzel v elektrickém obvodu, tedy X označuje uzly A až F, písmeno Y označuje pořadí členu v Taylorově řadě, tedy $Y \in \langle 1, \infty \rangle, Y \in \mathbb{Z}$.

Všimněme si, že v rovnici 2.41 již pro výpočet potřebujeme člen $DUCA1_{(0)}$, který ovšem ještě neznáme. Obdobná situace nastává i v dalších rovnicích 2.42 a 2.43. Z této skutečnosti

vyplývá, že musíme při výpočtu postupovat tak, že souběžně získávat nejdříve první členy Taylorových řad, tedy členy $DUCA1_{(0)}$, $DUCB1_{(0)}$, $DUCC1_{(0)}$, $DUCD1_{(0)}$, $DUCE1_{(0)}$. Potom získáme současně druhé členy Taylorových řad $DUCA2_{(0)}$, $DUCB2_{(0)}$, $DUCC2_{(0)}$, $DUCD2_{(0)}$, $DUCE2_{(0)}$, atd. Uvedeme nyní ostatní čtyři Taylorovy řady pro $UCB(1)$, $UCC(1)$, $UCD(1)$ a $UCE(1)$. Pro $UCB(1)$ sestavíme níže uvedenou Taylorovu řadu.

$$UCB(1) = UCB(0) + h \cdot U'_{CB(0)} + \frac{h^2}{2!} \cdot U''_{CB(0)} + \frac{h^3}{3!} \cdot U'''_{CB(0)} + \frac{h^4}{4!} \cdot U''''_{CB(0)} \quad (2.44)$$

Výpočet členů Taylorovy řady můžeme vidět níže.

$$DUCB1_{(0)} = h \cdot (UCA(0) - 2UCB(0) + UCC(0)) \quad (2.45)$$

$$DUCB2_{(0)} = \frac{h}{2} \cdot (DUCA1_{(0)} - 2DUCB1_{(0)} + DUCC1_{(0)}) \quad (2.46)$$

$$DUCB3_{(0)} = \frac{h}{3} \cdot (DUCA2_{(0)} - 2DUCB2_{(0)} + DUCC2_{(0)}) \quad (2.47)$$

$$DUCB4_{(0)} = \frac{h}{4} \cdot (DUCA3_{(0)} - 2DUCB3_{(0)} + DUCC3_{(0)}) \quad (2.48)$$

Taylorovu řadu pro $UCC(1)$ je znázorněna níže.

$$UCC(1) = UCC(0) + h \cdot U'_{CC(0)} + \frac{h^2}{2!} \cdot U''_{CC(0)} + \frac{h^3}{3!} \cdot U'''_{CC(0)} + \frac{h^4}{4!} \cdot U''''_{CC(0)} \quad (2.49)$$

Členy Taylorovy řady jsou vypočteny následujícím způsobem.

$$DUCC1_{(0)} = h \cdot (UCB(0) - 2UCC(0) + UCD(0)) \quad (2.50)$$

$$DUCC2_{(0)} = \frac{h}{2} \cdot (DUCB1_{(0)} - 2DUCC1_{(0)} + DUCD1_{(0)}) \quad (2.51)$$

$$DUCC3_{(0)} = \frac{h}{3} \cdot (DUCB2_{(0)} - 2DUCC2_{(0)} + DUCD2_{(0)}) \quad (2.52)$$

$$DUCC4_{(0)} = \frac{h}{4} \cdot (DUCB3_{(0)} - 2DUCC3_{(0)} + DUCD3_{(0)}) \quad (2.53)$$

Taylorova řada pro $UCD(1)$ má následující tvar.

$$UCD(1) = UCD(0) + h \cdot U'_{CD(0)} + \frac{h^2}{2!} \cdot U''_{CD(0)} + \frac{h^3}{3!} \cdot U'''_{CD(0)} + \frac{h^4}{4!} \cdot U''''_{CD(0)} \quad (2.54)$$

Členy Taylorovy řady vypočteme pomocí níže uvedených vztahů.

$$DUCD1_{(0)} = h \cdot (UCC(0) - 2UCD(0) + UCE(0)) \quad (2.55)$$

$$DUCD2_{(0)} = \frac{h}{2} \cdot (DUCC1_{(0)} - 2DUCD1_{(0)} + DUCE1_{(0)}) \quad (2.56)$$

$$DUCD3_{(0)} = \frac{h}{3} \cdot (DUCC2_{(0)} - 2DUCD2_{(0)} + DUCE2_{(0)}) \quad (2.57)$$

$$DUCD4_{(0)} = \frac{h}{4} \cdot (DUCC3_{(0)} - 2DUCD3_{(0)} + DUCE3_{(0)}) \quad (2.58)$$

Následuje Taylorova řada pro $UCE(1)$.

$$UCE(1) = UCE(0) + h \cdot U'_{CE(0)} + \frac{h^2}{2!} \cdot U''_{CE(0)} + \frac{h^3}{3!} \cdot U'''_{CE(0)} + \frac{h^4}{4!} \cdot U''''_{CE(0)} \quad (2.59)$$

Členy Taylorovy řady jsou vypočteny níže.

$$DUCE1_{(0)} = h \cdot (U_{CD(0)} - U_{CE(0)}) \quad (2.60)$$

$$DUCE2_{(0)} = \frac{h}{2} \cdot (DUCD1_{(0)} - DUCE1_{(0)}) \quad (2.61)$$

$$DUCE3_{(0)} = \frac{h}{3} \cdot (DUCD2_{(0)} - DUCE2_{(0)}) \quad (2.62)$$

$$DUCE4_{(0)} = \frac{h}{4} \cdot (DUCD3_{(0)} - DUCE3_{(0)}) \quad (2.63)$$

Taylorova řada je potenciálně nekonečná a proto je tedy nutné stanovit podmínky, kdy bude výpočet ukončen. Podmínkou pro ukončení výpočtu může být splnění přesnosti ε nebo dosažení maximálního simulačního času TMAX. Při výpočtu budeme postupovat podle algoritmu 2.1:

Algoritmus 2.1. 1. Stanovíme přesnost výpočtu ε .

2. Počítáme k -té aproximace, $k \in \langle 1, \infty \rangle, k \in \mathbb{Z}$. Získáváme aproximaci $U_{CX(k)}$, kde X označuje uzly A až F . V rámci k -té aproximace postupně počítáme členy $DUCEXY$, kde $Y \in \langle 1, \infty \rangle$.

3. Výpočet končí, až je v jednom kroku k splněna přesnost pro všechny nově vypočtené členy $DUCEXY$. Tedy jakmile platí, že všechny členy $|DUCEXY| < \varepsilon$.

4. Posuneme se v simulačním čase $t = t + h$.

5. Pokud $t > TMAX$, výpočet končí.

6. Výsledná hodnota pro $U_{CX(k)}$ je suma všech vypočtených členů $DUCEXY$.

Pomocí tohoto algoritmu získáme postupně hodnoty napětí na kapacitách C_1, C_2 až C_5 v intervalu $\langle t, TMAX \rangle$, s daným krokem h .

2.3 Experimentální výpočty

V této podkapitole se zaměříme na simulační systém TKSL. Tento systém byl vytvořen pro testování algoritmů, které využívají Taylorovu řadu pro řešení diferenciálních rovnic. Výpočet probíhá s proměnným integračním krokem. K dispozici jsou dvě verze. Starší verze systému nese jméno TKSL/386¹. Pro práci s tímto systémem je potřeba mít nainstalovaný emulátor DOSBox². Novější verzí simulačního nástroje TKSL/386 je systém TKSL/C³, který je napsán v jazyce C++. Oproti staršímu systému přináší řadu výhod, například: je možné řešit libovolný počet rovnic, systém TKSL/C je přenositelný, disponuje jednodušší syntaxí, lze využívat víceslovnou aritmetiku a také ho lze začlenit do jiných projektů.

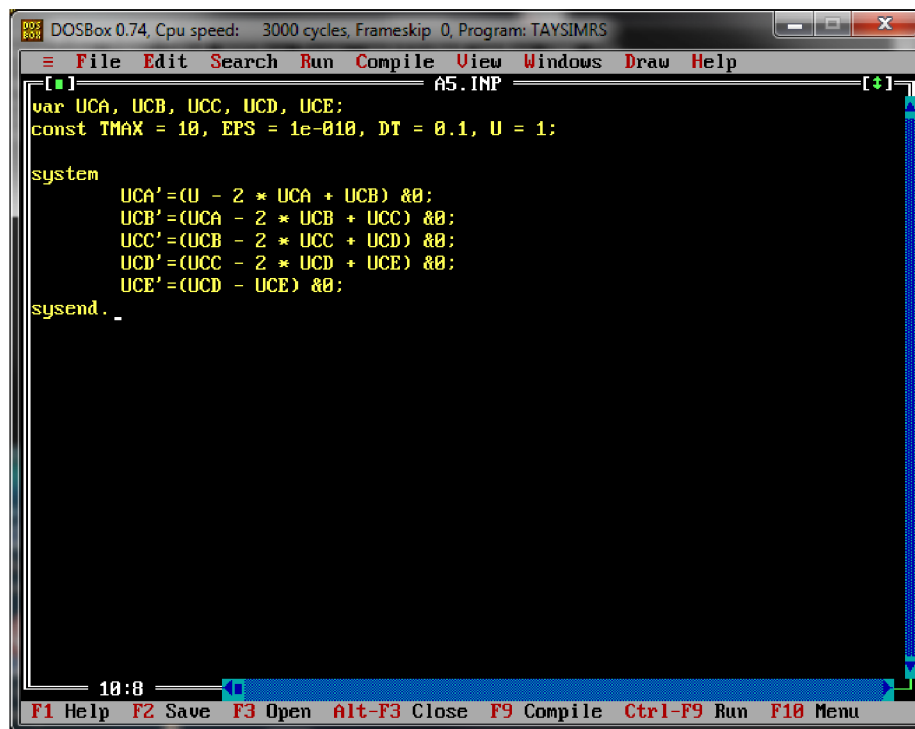
¹Více informací na <http://www.fit.vutbr.cz/~kunovsky/TKSL/tksl386.html.cs>.

²Dostupný zdarma na <http://www.dosbox.com/>.

³Více informací na <http://www.fit.vutbr.cz/~kunovsky/TKSL/tkslc.html.cs>.

2.3.1 Systém TKSL a proměnný integrační krok

Systém TKSL využívá pro výpočet volitelný řád numerické integrační metody. Problematiku si budeme opět vysvětlovat na příčkovém RC článku s pěti kapacitami C_1, C_2 až C_5 . Zdrojový kód můžeme vidět na obrázku 2.5 a vychází z rovnic 2.34 až 2.38. Pro zápis počátečních podmínek se používá znak $\&$. Všechny počáteční podmínky uvažujeme nulové. Na obrázcích 2.6, 2.7 a 2.8 jsou vyjádřeny závislosti napětí jednotlivých kapacit na čase.



```
DOSBox 0.74, Cpu speed: 3000 cycles, Frameskip 0, Program: TAYSIMRS
File Edit Search Run Compile View Windows Draw Help
[ ] AS.INP [ + ]
var UCA, UCB, UCC, UCD, UCE;
const TMAX = 10, EPS = 1e-010, DT = 0.1, U = 1;

system
    UCA'=(U - 2 * UCA + UCB) &0;
    UCB'=(UCA - 2 * UCB + UCC) &0;
    UCC'=(UCB - 2 * UCC + UCD) &0;
    UCD'=(UCC - 2 * UCD + UCE) &0;
    UCE'=(UCD - UCE) &0;
sysend .
10:8
F1 Help F2 Save F3 Open Alt-F3 Close F9 Compile Ctrl-F9 Run F10 Menu
```

Obrázek 2.5: Zdrojový kód pro příčkový RC článek o pěti kapacitách

Všimněme si ale také žluté schodovité přímky v horní části grafu, která vyjadřuje řád Taylorovy řady (ORD). Vidíme, že na grafu 2.6 v čase $t = 0.1$ je řád $ORD = 12$. Na dalším grafu 2.7 v čase $t = 0.3$ je řád $ORD = 11$. Poslední graf 2.8 ukazuje, že v čase $t = 0.4$ je řád $ORD = 10$. Řád potřebný pro výpočet členů Taylorovy řady se tedy postupně snižuje. První členy potřebují pro svůj výpočet vyšší řád. Čím se nacházíme na časové ose dál, tím menší řád je pro výpočet členů potřeba.

Řád metody lze ovlivnit více faktory. Pokud zvýšíme integrační krok h , bude řád Taylorovy řady vyšší. Důvodem je to, že pro dosažení stejné přesnosti musíme využít více členů Taylorovy řady, také potřebujeme delší simulační čas TMAX. Další situací, kdy může dojít ke zvýšení řádu metody je, pokud rozdíl mezi napětími U a U_{CX} (napětí na kapacitách) je příliš velký. Nyní zkusíme zvýšit integrační krok metody, změním ve zdrojovém kódu pro systém TKSL/386 hodnotu konstanty DT. Nastavíme tedy $DT = 1$ (viz obrázek 2.9). Výsledky můžeme vidět v grafech 2.10 a 2.11. V čase $t = 1$ je řád $ORD = 24$, v čase $t = 2$ potom $ORD = 21$. Hodnota řádu je tedy výrazně větší, než když jsme počítali s krokem $DT = 0.1$. V dalších grafech 2.13 a 2.14 můžeme vidět výstupy pro případ, kdy zvýšíme řád metody na hodnotu 2, tedy $DT = 2$ (viz obrázek 2.12). Můžeme si všimnout, že v čase $t = 2$ je řád $ORD = 35$, v čase $t = 4$ je řád $ORD = 29$. Opět došlo k výraznému zvýšení řádu metody.

Abychom snížili řád metody, můžeme se pokusit vhodným způsobem modifikovat algoritmus 2.1. Níže uvádíme nový algoritmus 2.2. Kontrolujeme přesnost ε u každého nově vypočteného členu $DUCXY$. Pokud zjistíme, že je daný člen $|DUCXY| < \varepsilon$, znamená to, že přírůstek je tak malý, že již příliš neovlivní výsledek. Tento člen nastavíme na hodnotu 0. V dalších krocích pak není zahrnut do výpočtů, zůstává tedy nulový. Výpočet končí v případě, že všechny členy $DUCXY$ mají nastavenou nulovou hodnotu.

Algoritmus 2.2. 1. Stanovíme přesnost výpočtu ε .

2. Počítáme k -té aproximace, $k \in \langle 1, \infty \rangle, k \in \mathbb{Z}$. Získáváme aproximaci $U_{CX(k)}$, kde X označuje uzly A až F . V rámci k -té aproximace postupně počítáme členy $DUCXY$, kde $Y \in \langle 1, \infty \rangle$.

3. Pokud je absolutní hodnota aktuálně počítaného členu menší než přesnost ε , $|DUCXY| < \varepsilon$, potom hodnota tohoto členu $DUCXY = 0$. V dalších krocích s tímto členem neprovádíme žádné výpočty.

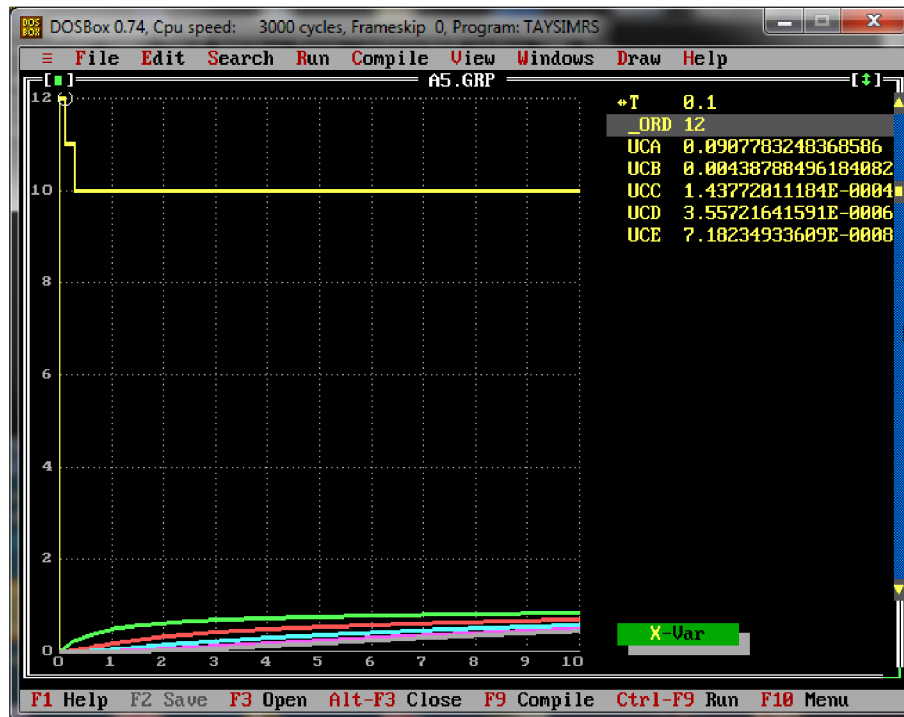
4. Výpočet končí, pokud všechny členy $DUCXY$ mají nastavenou nulovou hodnotu.

5. Posuneme se v simulačním čase $t = t + h$.

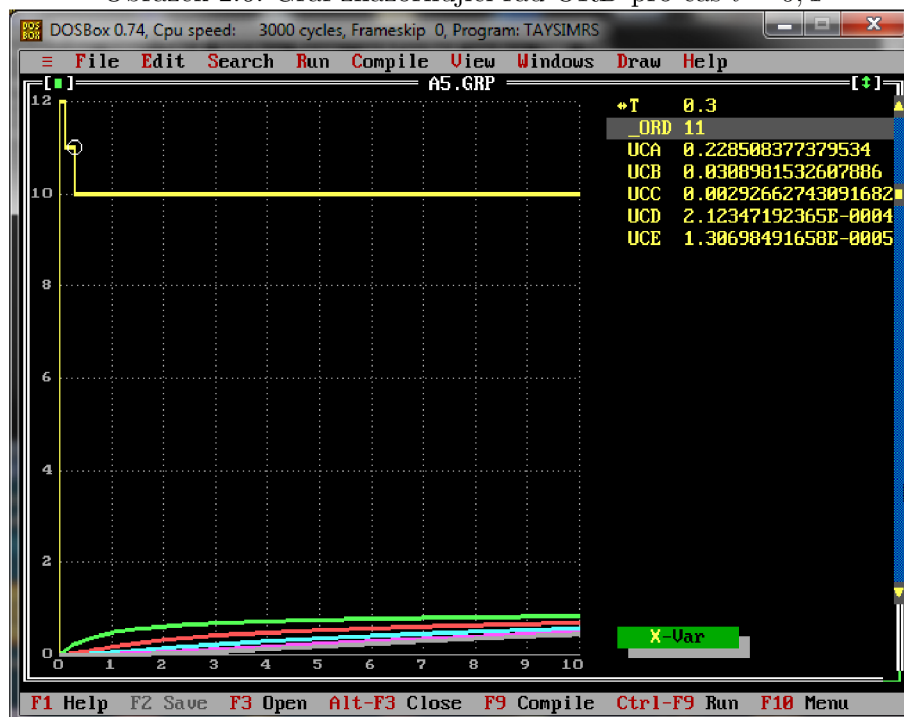
6. Pokud $t > TMAX$, výpočet končí.

7. Výsledná hodnota pro $U_{CX(k)}$ je suma všech vypočtených členů $DUCXY$.

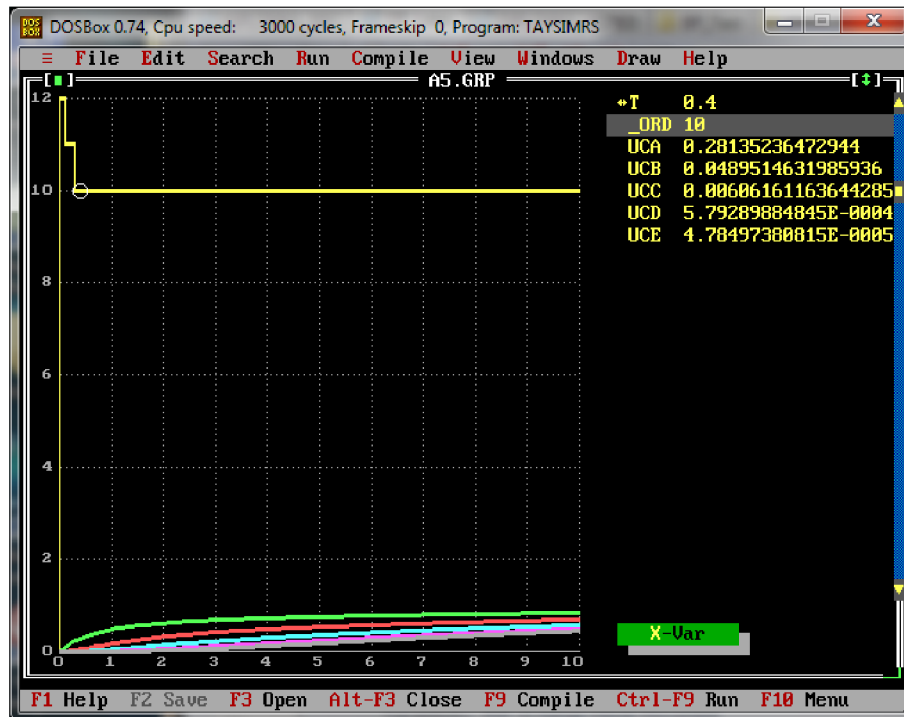
V rámci diplomové práce byl napsán zkušební program, který implementuje výše popsaný algoritmus. Program je napsán v jazyce C++. Generuje také skripty pro systém Gnuplot, je tedy možné vytvářet grafické výstupy. Grafy 2.15, 2.16, 2.17, 2.18 a 2.19 ukazují hodnotu řádu ORD v závislosti na čase t pro každou proměnnou – tedy pro proměnné U_{CA} , U_{CB} až U_{CE} . Oproti grafu 2.6 jsou hodnoty řádu ORD menší.



Obrázek 2.6: Graf znázorňující řád ORD pro čas $t = 0,1$



Obrázek 2.7: Graf znázorňující řád ORD pro čas $t = 0,3$



Obrázek 2.8: Graf znázorňující řád ORD pro čas $t = 0,4$

DOSBox 0.74, Cpu speed: 3000 cycles, Frameskip 0, Program: TAYSIMRS

File Edit Search Run Compile View Windows Draw Help

A5_STEP.INP

```

var UCA, UCB, UCC, UCD, UCE;
const TMAX = 10, EPS = 1e-010, DT = 1, U = 1;

system
  UCA'=(U - 2 * UCA + UCB) &0;
  UCB'=(UCA - 2 * UCB + UCC) &0;
  UCC'=(UCB - 2 * UCC + UCD) &0;
  UCD'=(UCC - 2 * UCD + UCE) &0;
  UCE'=(UCD - UCE) &0;

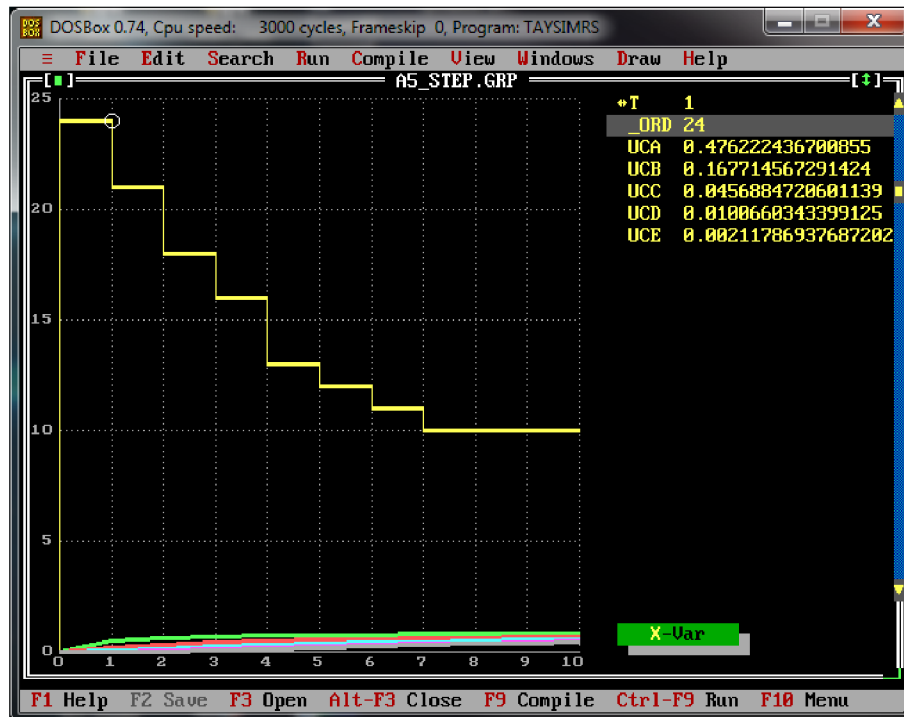
sysend .

```

10:8

F1 Help F2 Save F3 Open Alt-F3 Close F9 Compile Ctrl-F9 Run F10 Menu

Obrázek 2.9: Zdrojový kód, integrační krok $DT = 1$



Obrázek 2.10: Graf znázorňující hodnotu řádu ORD v závislosti na čase t ($DT = 1$)



Obrázek 2.11: Graf znázorňující hodnotu řádu ORD v čase $t = 2$ ($DT = 1$)


```

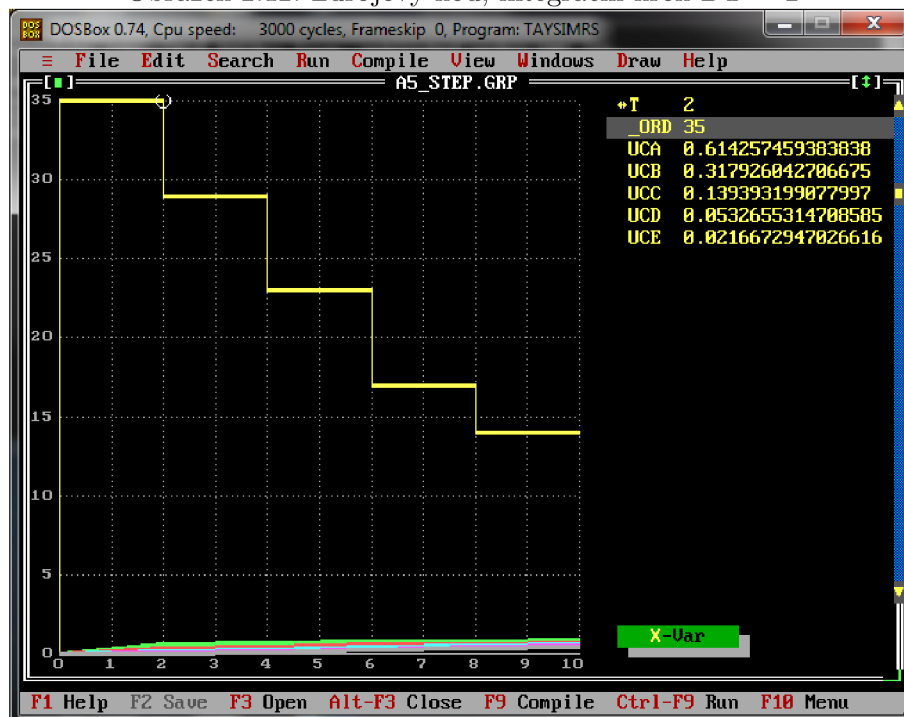
DOSBox 0.74, Cpu speed: 3000 cycles, Frameskip 0, Program: TAYSIMRS
File Edit Search Run Compile View Windows Draw Help
A5_STEP.INP
var UCA, UCB, UCC, UCD, UCE:
const TMAX = 10, EPS = 1e-010, DT = 2, U = 1:

system
UCA'=(U - 2 * UCA + UCB) &0;
UCB'=(UCA - 2 * UCB + UCC) &0;
UCC'=(UCB - 2 * UCC + UCD) &0;
UCD'=(UCC - 2 * UCD + UCE) &0;
UCE'=(UCD - UCE) &0;

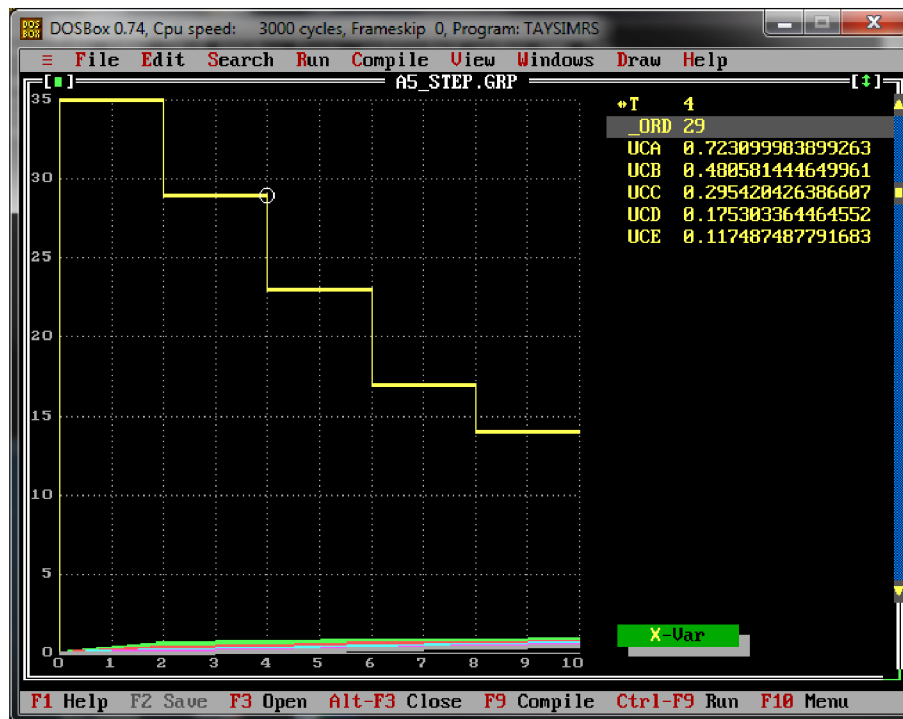
sysend _
10:8
F1 Help F2 Save F3 Open Alt-F3 Close F9 Compile Ctrl-F9 Run F10 Menu

```

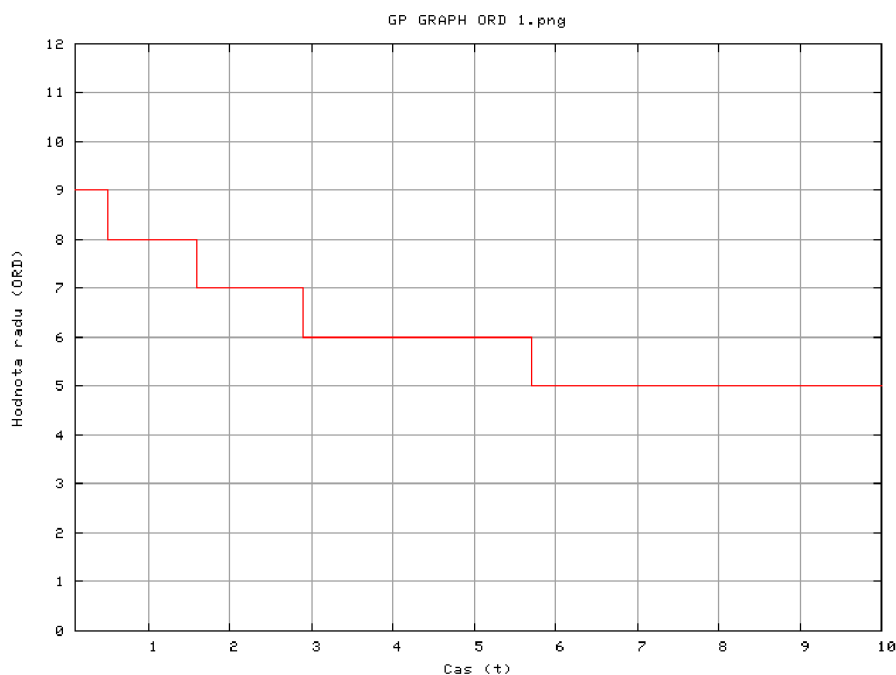
Obrázek 2.12: Zdrojový kód, integrační krok $DT = 2$



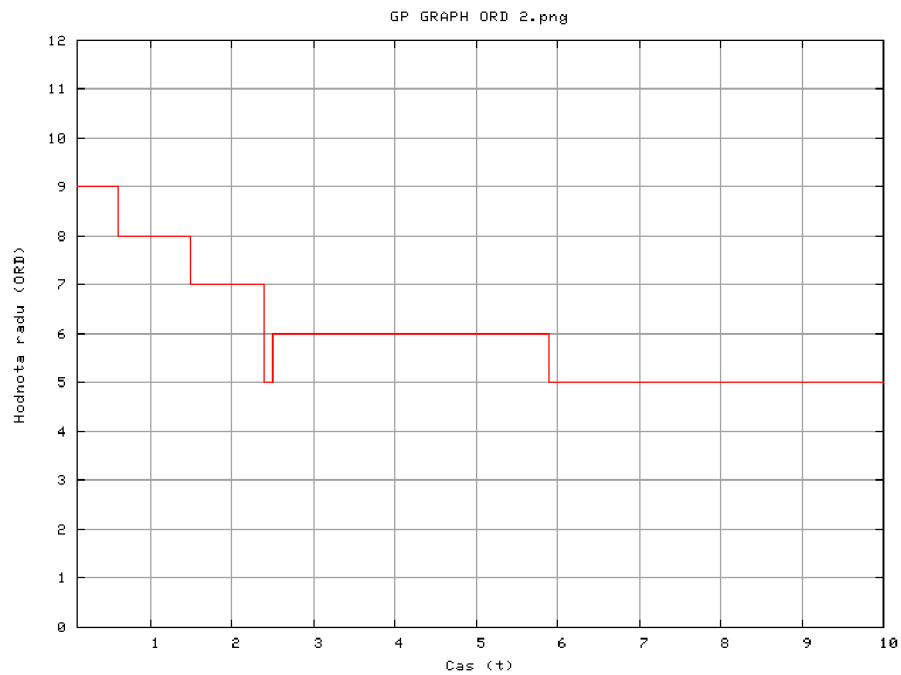
Obrázek 2.13: Graf znázorňující hodnotu řádu ORD v čase $t = 2$ ($DT = 2$)



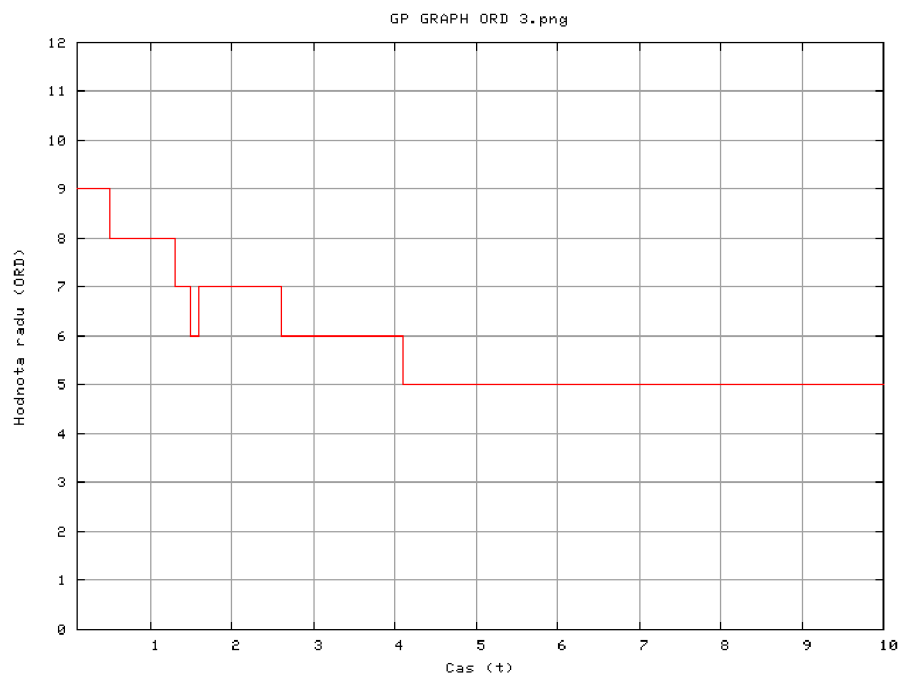
Obrázek 2.14: Graf znázorňující hodnotu řádu ORD v čase $t = 4$ ($DT = 2$)



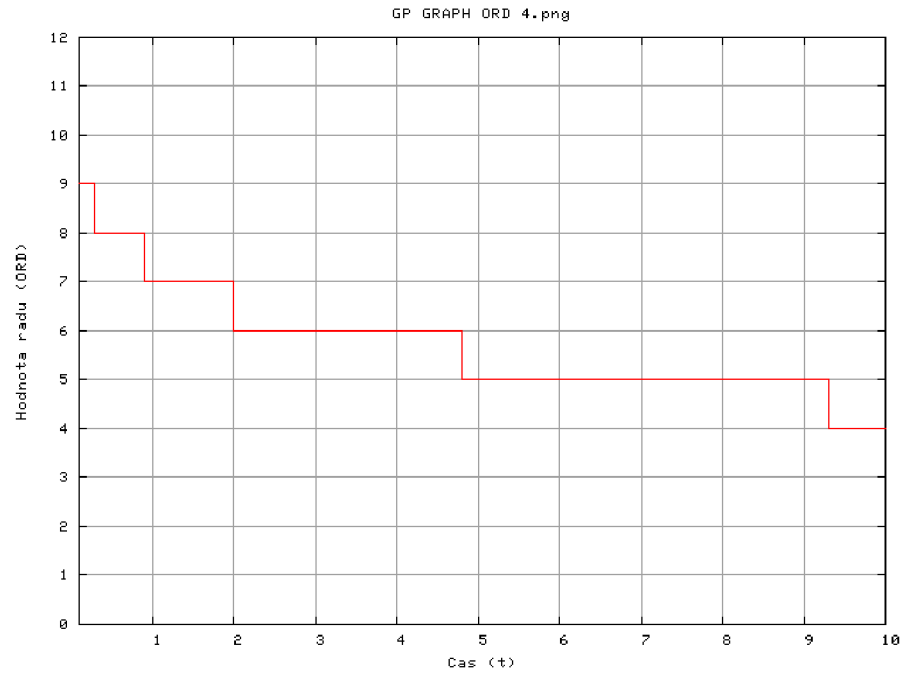
Obrázek 2.15: Graf znázorňující hodnotu řádu ORD v závislosti na čase t pro proměnnou U_{CA}



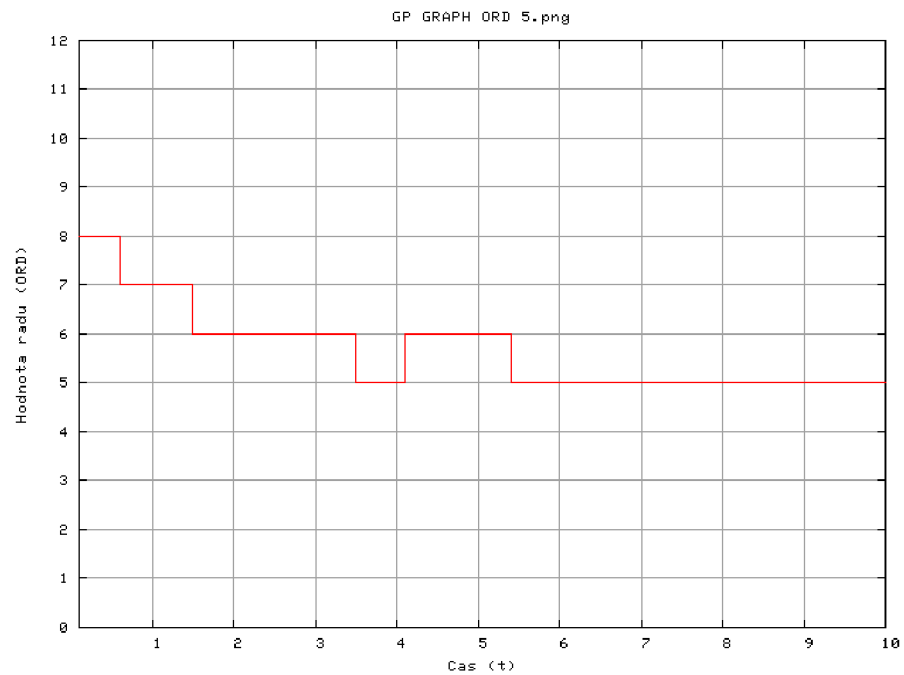
Obrázek 2.16: Graf znázorňující hodnotu řádu ORD v závislosti na čase t pro proměnnou U_{CB}



Obrázek 2.17: Graf znázorňující hodnotu řádu ORD v závislosti na čase t pro proměnnou U_{CC}



Obrázek 2.18: Graf znázorňující hodnotu řádu ORD v závislosti na čase t pro proměnnou U_{CD}



Obrázek 2.19: Graf znázorňující hodnotu řádu ORD v závislosti na čase t pro proměnnou U_{CE}

Kapitola 3

Parciální diferenciální rovnice

Rovnice uvedené v definici 3.1, se nazývají *parciální diferenciální rovnice (PDR)*. Podrobněji se parciálními diferenciálními rovnicemi zabývají publikace [11, 1, 4].

Definice 3.1. *Parciální diferenciální rovnice k-tého řádu je rovnice ve tvaru*

$$F\left(x_1, x_2, \dots, x_n, z, \frac{\partial z}{\partial x_1}, \dots, \frac{\partial z}{\partial x_n}, \frac{\partial^2 z}{\partial x_1^2}, \frac{\partial^2 z}{\partial x_1 \partial x_2}, \dots, \frac{\partial^2 z}{\partial x_1 \partial x_n}, \frac{\partial^2 z}{\partial x_2^2}, \dots, \frac{\partial^k z}{\partial x_n^k}\right) = 0$$

kde $z(x_1, x_2, \dots, x_n)$ je neznámá funkce n proměnných. **Řád** parciální diferenciální rovnice je určen řádem nejvyšší derivace, která se v rovnici vyskytuje.

3.1 Typy parciálních diferenciálních rovnic

Nyní se budeme zabývat parciálními diferenciálními rovnicemi druhého řádu, viz definice 3.2.

Definice 3.2. *Rovnice ve tvaru*

$$F\left(x, y, u, \frac{\partial u}{\partial x}, \frac{\partial u}{\partial y}, \frac{\partial^2 u}{\partial x^2}, \frac{\partial^2 u}{\partial x \partial y}, \frac{\partial^2 u}{\partial y^2}\right) = 0$$

se nazývá **parciální diferenciální rovnicí druhého řádu** pro neznámou funkci u dvou proměnných, kde $F(x, y, u, p, q, r, s, t)$ je funkce 8 proměnných.

Pomocí parciálních diferenciálních rovnic lze řešit celou řadu technických i fyzikálních problémů. Některé parciální diferenciální rovnice se proto nazývají *rovnice matematické fyziky* a popisují některé fyzikální děje (v určitém rozsahu a s určitou přesností). Podle dimenze prostoru, ve kterém zkoumaný jev probíhá, je neznámou funkce $u(x, t)$, $u(x, y, t)$, $u(x, y, z, t)$, případně pak $u(x_1, x_2, \dots, x_N, t)$ ve vyšších dimenzích.

Mezi nejznámější typy PDR patří:

- Parabolická PDR (difuzní rovnice) $\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2}$,
- Hyperbolická PDR (vlnová rovnice) $\frac{\partial^2 u}{\partial t^2} = \frac{\partial^2 u}{\partial x^2}$,
- Eliptická PDR (Laplaceova rovnice) $\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = 0$.

3.1.1 Parabolická PDR

Tato rovnice slouží pro popis přenosu tepla. Máme tepelně izolovanou tyč délky l , její průřez zanedbáme. Tyč je umístěna na ose x a její levý konec leží v počátku souřadnicového systému. Předpokládáme, že proměnná $u = u(x, t)$ popisuje teplotu tyče v místě x a čase t . Lze dokázat, že tato funkce splňuje parciální diferenciální rovnici 3.12.

$$\frac{\partial u}{\partial t} = a^2 \frac{\partial^2 u}{\partial x^2} + f(x, t), \quad (x, t) \in (0, l) \times (0, T) \quad (3.1)$$

kde $a^2 = \frac{k}{\rho c}$ je tepelná difuzivita, k je koeficient tepelné vodivosti, ρ je měrná hmotnost, c je měrné teplo. Funkce $f(x, t)$ charakterizuje intenzitu vnitřních zdrojů (např. pokud tyčí prochází elektrický proud, vyvíjí se v ní teplo), T je doba trvání zkoumaného děje.

Rovnice 3.12 má nekonečně mnoho řešení. Aby bylo možné určit teplotu tyče jednoznačně v libovolném místě a čase, je nutné k rovnici připojit počáteční podmínky. Jednu počáteční podmínku popisující teplotu tyče na počátku děje a potom dvě okrajové podmínky, které charakterizují situaci na obou koncích tyče v průběhu celého děje. Tyto podmínky lze popsat vztahy:

$$u(x, 0) = g(x), \quad 0 < x < l, \quad (3.2)$$

$$u(0, t) = h_1(t), \quad 0 < t < T, \quad (3.3)$$

$$u(l, t) = h_2(t), \quad 0 < t < T. \quad (3.4)$$

Podmínky 3.2, 3.3 a 3.4 popisují situaci, kdy levý konec tyče je udržován na teplotě h_1 , druhý konec na teplotě h_2 . Funkce g_x popisuje teplotu tyče na počátku děje.

3.1.2 Hyperbolická PDR

Hyperbolická parciální diferenciální rovnice se často používá při popisu šíření vln. Mějme dokonale pružnou strunu délky l , která je ukotvena na ose x a napínána konstantní silou F . Kmitání popisuje rovnice 3.5.

$$\frac{\partial^2 u}{\partial t^2} = a^2 \frac{\partial^2 u}{\partial x^2} + f(x, t), \quad (x, t) \in (0, l) \times (0, T) \quad (3.5)$$

kde $a^2 = \frac{F}{\rho}$, ρ je délková měrná hmotnost struny, F je napínací síla a $u(x, t)$ je vertikální výchylka od rovnovážné polohy struny v bodě x a čase t a $f(x, t)$ znamená případné další vnější zatížení (například gravitace). K rovnici 3.5 doplníme počáteční podmínky (viz rovnice 3.6).

$$u(x, 0) = g_1(x), \quad \frac{\partial u}{\partial t}(x, 0) = g_2(x), \quad 0 < x < l \quad (3.6)$$

Oproti rovnici vedení tepla je nutné uvažovat počáteční výchylku struny a také počáteční rychlost struny. Okrajové podmínky charakterizující chování struny v místech uchycení lze opět popsat vztahy 3.3 a 3.4.

3.1.3 Eliptická PDR

Eliptická parciální diferenciální rovnice se využívá v časově stálých modelech. V závislosti na dimenzi můžeme rovnice vedení tepla a vlnovou rovnici vyjádřit ve tvaru 3.7.

$$\frac{\partial u}{\partial t} = a^2 \Delta u + f, \quad \frac{\partial^2 u}{\partial t^2} = a^2 \Delta u + f \quad (3.7)$$

Symbol Δ se nazývá *Laplaceův operátor*, který je dán definicí 3.3.

Definice 3.3. Platí, že obecně v prostoru proměnných x_1, \dots, x_N má Laplaceův operátor tvar

$$\Delta = \frac{\partial^2}{\partial x_1^2} + \dots + \frac{\partial^2}{\partial x_N^2}$$

V případě časově stálých modelů mají rovnice vedení tepla a vlnová rovnice formálně stejný tvar

$$-\Delta u = f, \quad (3.8)$$

kde Δ je Laplaceův operátor a f je funkce proměnných x_1, \dots, x_n . Rovnice 3.8 se nazývá *Poissonova rovnice*. Ve dvourozměrném prostoru, kdy $N = 2$, můžeme rovnici zapsat ve tvaru 3.9.

$$-\frac{\partial^2 u}{\partial x^2} - \frac{\partial^2 u}{\partial y^2} = f(x, y) \quad (3.9)$$

Zvláštním případem Poissonovy rovnice 3.8 je *Laplaceova rovnice*.

$$\Delta u = 0 \quad (3.10)$$

Rovnici 3.10 můžeme zapsat ve dvourozměrném prostoru ve tvaru 3.11.

$$\frac{\partial^2 u}{\partial x^2} - \frac{\partial^2 u}{\partial y^2} = 0 \quad (3.11)$$

3.2 Převod PDR na ODR

Aby bylo možné řešit tyto rovnice systémem TKSL, je nutné soustavu rovnic upravit do jiného tvaru. Pro řešení PDR se využívají diferenční metody, jejichž základem je aproximace parciálních derivací konečnými diferenčními podíly. Metoda sítí je výsledkem nahrazení všech parciálních derivací v rovnici, naproti tomu metoda přímků ponechává pouze jednu spojitou proměnnou (čas) a derivace ostatních proměnných jsou nahrazeny diferenčními podíly.

3.2.1 Metoda přímků

Tuto metodu lze odvodit z rovnice vedení tepla v jednorozměrné ideální tyči. Časovou proměnnou t necháme měnit spojitě a prostorovou proměnnou x diskretizujeme.

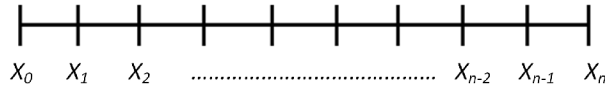
$$\frac{\partial u}{\partial t} = a^2 \frac{\partial^2 u}{\partial x^2} + f(x, t), \quad (x, t) \in (0, l) \times (0, T) \quad (3.12)$$

Interval $\langle 0, l \rangle$ musíme rozdělit na n dílků o délce h , viz vztah 3.13.

$$\Delta x = h = \frac{L}{n} \quad (3.13)$$

Graficky můžeme tuto skutečnost znázornit obrázkem 3.1. Ve směru osy x budeme parciální derivace nahrazovat v každém diskrétním bodě (uzlu) podílem konečných diferencí. První parciální derivace $u = u(x, t)$ podle x má následující tvar.

$$\frac{\partial u(x, t)}{\partial x} = \lim_{x \rightarrow 0} \frac{u(x + \Delta x, t) - u(x, t)}{\Delta x} \quad (3.14)$$



Obrázek 3.1: Princip metody přímek

Máme-li interval rozdělený na stejně velké části, potom můžeme pro konečnou hodnotu $x = h$ stanovit první aproximaci. V případě symetrických diferencí v libovolném bodě intervalu můžeme psát níže uvedený vztah.

$$\left. \frac{\partial u(x, t)}{\partial x} \right|_x \approx \frac{u(x + h/2, t) - u(x - h/2, t)}{h} \quad (3.15)$$

Pro bod $x = x_i$ platí následující vztah.

$$\left. \frac{\partial u(x, t)}{\partial x} \right|_{x_i} \approx \frac{u(x_{i+1/2}, t) - u(x_{i-1/2}, t)}{\Delta x} \approx \frac{u_{i+1/2}(t) - u_{i-1/2}(t)}{\Delta x} \quad (3.16)$$

Vztah 3.16 lze přepsat do následujícího tvaru.

$$\left. \frac{\partial u(x, t)}{\partial x} \right|_{x_i} \approx \frac{u_{i+1}(t) - u_{i-1}(t)}{2\Delta x} \quad (3.17)$$

Pokud využíváme nesymetrické vzorce, potom platí následující vztah.

$$\left. \frac{\partial u(x, t)}{\partial x} \right|_{x_{i+1/2}} \approx \frac{u_{i+1}(t) - u_i(t)}{\Delta x} \quad (3.18)$$

$$\left. \frac{\partial u(x, t)}{\partial x} \right|_{x_{i-1/2}} \approx \frac{u_i(t) - u_{i-1}(t)}{\Delta x} \quad (3.19)$$

Nyní uvedeme vztahy pro aproximaci vyšších derivací.

Pro 2. parciální derivaci platí níže uvedený vztah.

$$\left. \frac{\partial^2 u(x, t)}{\partial x^2} \right|_{x_i} \approx \frac{u_{i+1}(t) - 2u_i(t) + u_{i-1}(t)}{(\Delta x)^2} \quad (3.20)$$

Obdobným způsobem vyjádříme i vztah pro 3. parciální derivaci.

$$\left. \frac{\partial^3 u(x, t)}{\partial x^3} \right|_{x_i} \approx \frac{u_{i+2}(t) - 3u_{i+1}(t) + 3u_{i-1}(t) - u_{i-2}(t)}{2(\Delta x)^3} \quad (3.21)$$

Pro 4. parciální derivaci platí vztah, který je uveden níže.

$$\left. \frac{\partial^4 u(x, t)}{\partial x^4} \right|_{x_i} \approx \frac{u_{i+2}(t) - 4u_{i+1}(t) + 6u_i(t) - 4u_{i-1}(t) - u_{i-2}(t)}{2(\Delta x)^4} \quad (3.22)$$

Pokud potřebujeme vyšší přesnost řešení, můžeme použít větší počet přímek anebo použít vyšší řád diferenčního vzorce. U metody přímek je vždy nutné se rozhodnout, kterou proměnnou ponecháme spojitou a kterou naopak budeme diskretizovat. V případě rovnic matematické fyziky je tomu tak, že čas zachováme jako spojitou proměnnou.

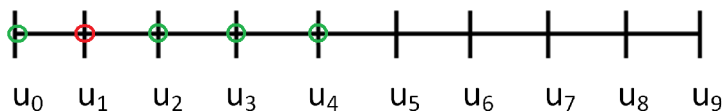
Kapitola 4

Parciální diferenciální rovnice v systému TKSL

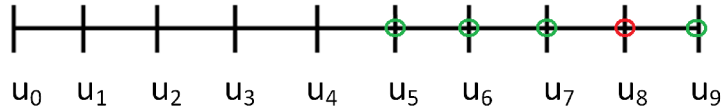
V této kapitole si ukážeme, jak vytvářet zdrojové texty pro systém TKSL. Ukážeme postupně konkrétní příklady pro všechny zmíněné typy PDR, tedy pro parabolickou, hyperbolickou i eliptickou PDR. Informace k systému TKSL jsou uvedeny v [9].

4.1 n-bodové aproximace

Tyč bude rozdělena deseti řezy (máme tedy devět částí) a využijeme tříbodovou aproximaci. Tříbodová aproximace znamená, že při určování přibližných diferencí bereme jeden uzel z levé strany od aktuálního bodu a jeden uzel z pravé strany. V případě pětibodové aproximace bereme dva uzly z levé strany od aktuálního bodu a dva uzly z pravé strany, atd. Je ale zapotřebí si uvědomit, že je nutné ve druhém (u_1) a předposledním bodě (u_8) použít nesymetrické aproximace. Situaci vystihuje obrázek 4.1. Pokud budeme využívat pětibodovou aproximaci a budeme počítat hodnotu v bodě u_1 , potom nelze využít symetrickou aproximaci, tedy dva body zleva a dva body zprava. Je proto nutné použít nesymetrickou aproximaci, kdy vezmeme pouze jeden bod z levé strany (u_0) a pak tři body ze strany pravé (u_2, u_3, u_4). Obdobně je tomu u předposledního bodu (u_8), viz 4.2, kdy naopak vezmeme tři body z levé strany (u_5, u_6, u_7) a jeden bod z pravé strany (u_9). Poznamenáme, že pokud máme hyperbolickou rovnici a uvažujeme tříbodovou aproximaci, potom nesymetrické vzorce nevyužijeme. Uvažujme, že máme strunu rozdělenou deseti řezy, které odpovídají bodům u_0 až u_{10} . Potom body u_0 až u_{10} jsou body uchycení struny a zde je hodnota nulová. Pomocí tříbodové aproximace vyjadřujeme body u_1 až u_9 . Pro bod u_1 máme tedy k dispozici jeden bod zleva, což je u_0 i bod zprava, což je u_2 . Pro bod u_9 máme k dispozici zleva bod u_8 a zprava bod u_{10} .



Obrázek 4.1: Pětibodová nesymetrická aproximace – bod u_1



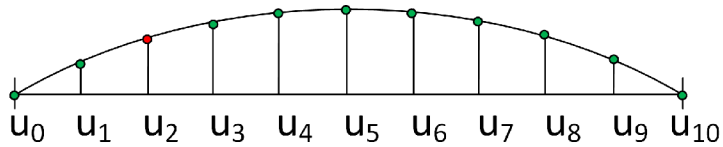
Obrázek 4.2: Pětibodová nesymetrická aproximace – bod u_8

4.2 Hyperbolická PDR

Hyperbolická parciální diferenciální rovnice popisuje kmitání struny, viz 3.1.2. Grafické znázornění můžeme vidět na obrázku 4.3.

$$\frac{\partial^2 u}{\partial t^2} = \frac{\partial^2 u}{\partial x^2} \quad (4.1)$$

U tohoto typu PDR je nutné provádět substituci. Pravou stranu rovnice upravíme pomocí tříbodové aproximace. Ukážeme si, jak probíhá úprava pro konkrétní bod u_2 .



Obrázek 4.3: Kmitající struna – aproximace pro bod u_2

Postup při náhradě druhé derivace probíhá následujícím způsobem.

$$\frac{\partial^2 u}{\partial x^2} \Big|_2 = \frac{\frac{\partial u}{\partial x} \Big|_2 - \frac{\partial u}{\partial x} \Big|_1}{\Delta x} = \frac{u_3 - u_2}{\Delta x} - \frac{u_2 - u_1}{\Delta x} = \frac{u_3 - 2u_2 + u_1}{(\Delta x)^2} \quad (4.2)$$

Hyperbolickou rovnici můžeme nyní vyjádřit pomocí níže uvedeného vztahu.

$$a \frac{\partial^2 u}{\partial t^2} \Big|_2 = \frac{1}{(\Delta x)^2} (u_3 - 2u_2 + u_1) \quad k = \frac{1}{a(\Delta x)^2} \quad (4.3)$$

Náhrada první derivace se provede za pomocí vztahu uvedeného níže.

$$\frac{\partial u}{\partial x} \Big|_2 \approx \frac{\Delta u}{\Delta x} \Big|_2 = \frac{u_2 - u_1}{\Delta x} \frac{\partial u}{\partial x} \Big|_1 \approx \frac{u_2 - u_1}{\Delta x} \quad (4.4)$$

Pravou stranu rovnice nyní převedeme pomocí přibližných diferencí.

$$\frac{\partial^2 u}{\partial x^2} \Big|_2 = k(u_1 - 2u_2 + u_3) \quad (4.5)$$

Celková podoba rovnice je uvedena níže.

$$\frac{\partial^2 u}{\partial t^2} = k(u_1 - 2u_2 + u_3) \quad (4.6)$$

Nyní již máme derivaci pouze podle jedné proměnné, můžeme tedy rovnici upravit na následující tvar.

$$\frac{d^2 u}{dt^2} = k(u_1 - 2u_2 + u_3) \quad (4.7)$$

$$u_2'' = k(u_3 - 2u_2 + u_1) \quad (4.8)$$

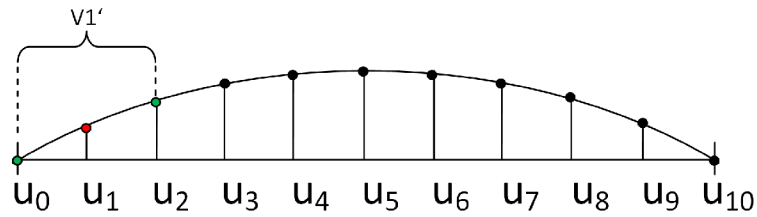
System TKSL ovšem dokáže řešit pouze diferenciální rovnice prvního řádu. Proto je nyní potřeba zavést substituci, díky které získáme dvě rovnice prvního řádu.

$$u_2' = v_2 \quad (4.9)$$

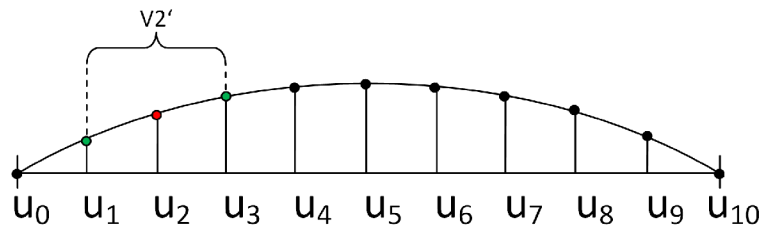
$$v_2' = k(u_3 - 2u_2 + u_1) \quad (4.10)$$

Obdobně lze samozřejmě aproximovat i další body. Kompletní zdrojový kód je uveden níže. Body u_0 a u_{10} jsou nastaveny na nulu, protože se jedná o body, ve kterých je struna ukotvena. V bodech u_1 až u_9 probíhá výpočet tak, že bereme jeden bod z levé strany a jeden bod ze strany pravé. Výpočet pro body u_1, u_2 a u_3 je znázorněn na obrázcích 4.4, 4.5 a 4.6.

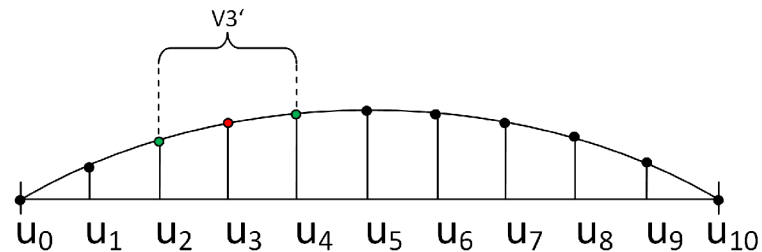
```
var u0, u1, u2, u3, u4, u5, u6, u7, u8, u9, u10,
v1, v2, v3, v4, v5, v6, v7, v8, v9;
Const
  PR= 10,
  eps=1e-20,
  A= PR*PR,
  tmax=20,
  dt=0.1,
  PI= 3.1415926535897932385;
System
  u0= 0;
  u10= 0;
  v1'= A* (u0-2*u1+u2) &Sin(PI*1/PR);
  v2'= A* (u1-2*u2+u3) &Sin(PI*2/PR);
  v3'= A* (u2-2*u3+u4) &Sin(PI*3/PR);
  v4'= A* (u3-2*u4+u5) &Sin(PI*4/PR);
  v5'= A* (u4-2*u5+u6) &Sin(PI*5/PR);
  v6'= A* (u5-2*u6+u7) &Sin(PI*6/PR);
  v7'= A* (u6-2*u7+u8) &Sin(PI*7/PR);
  v8'= A* (u7-2*u8+u9) &Sin(PI*8/PR);
  v9'= A* (u8-2*u9+u10) &Sin(PI*9/PR);
  u1'= v1 &0;
  u2'= v2 &0;
  u3'= v3 &0;
  u4'= v4 &0;
  u5'= v5 &0;
  u6'= v6 &0;
  u7'= v7 &0;
  u8'= v8 &0;
  u9'= v9 &0;
SysEnd.
```



Obrázek 4.4: Tříbodová aproximace – bod u_1



Obrázek 4.5: Tříbodová aproximace – bod u_2



Obrázek 4.6: Tříbodová aproximace – bod u_3

4.3 Parabolická PDR

Jak již bylo řečeno v podkapitole (3.1.1), parabolická PDR popisuje vedení tepla v tenké izolované tyči.

$$\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2} \quad (4.11)$$

Budeme používat tříbodovou aproximaci. U tohoto typu PDR nepotřebujeme zavádět žádné substitute, přímo získáme diferenciální rovnici prvního řádu. Diferenční výraz na pravé straně rovnice pouze nahradíme přibližnými diferencemi, jak jsme již ukázali v předchozí podkapitole 4.2.

```
var u0, u1, u2, u3, u4, u5, u6, u7, u8, u9, u10;
```

```
System
```

```
u0=0;
```

```
u10=0;
```

```
u1'=(u0-2*u1+u2) &0;
```

```
u2'=(u1-2*u2+u3) &0;
```

```
u3'=(u2-2*u3+u4) &0
```

```
u4'=(u3-2*u4+u5) &0;
```

```
u5'=(u4-2*u5+u6) &0;
```

```
u6'=(u5-2*u6+u7) &0;
```

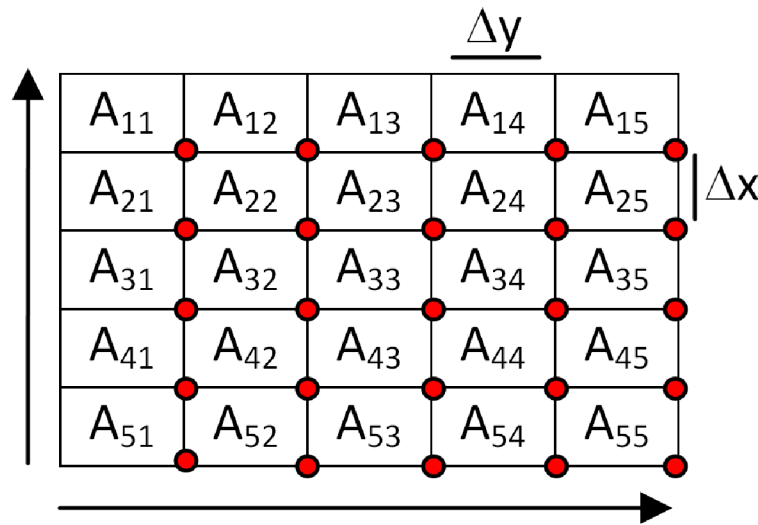
$u_7' = (u_6 - 2u_7 + u_8) \cdot 0;$
 $u_8' = (u_7 - 2u_8 + u_9) \cdot 0;$
 $u_9' = (u_8 - 2u_9 + u_{10}) \cdot 0;$
 Sysend.

4.4 Eliptická PDR

Eliptická parciální diferenciální rovnice má následující tvar (viz 3.1.3).

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = 0 \quad (4.12)$$

Zde platí, že musíme nahradit derivace ve směru osy x i y . Situaci si můžeme představit jako dvourozměrnou síť, viz obrázek 4.7.



Obrázek 4.7: Náhrada derivací ve směru osy x i y

Musíme tedy pomocí přibližných diferencí nahradit levou i pravou stranu rovnice. Ukážeme konkrétní situaci, kdy chceme aproximovat hodnotu v bodě A_{22} a budeme využívat tříbodovou aproximaci. Začneme levou stranou. Zde budeme nahrazovat ve směru osy x , při sestavování rovnic pro poměrové difference tedy bereme jeden prvek z levé strany (A_{21}) a jeden prvek z pravé strany (A_{23}). Na pravé straně probíhá nahrazování ve směru osy y , bereme tedy jeden prvek shora (A_{12}) a jeden zespodu (A_{32}). Situaci znázorňuje obrázek 4.8.

Pro levou stranu rovnice provedeme náhradu pomocí přibližných diferencí následovně.

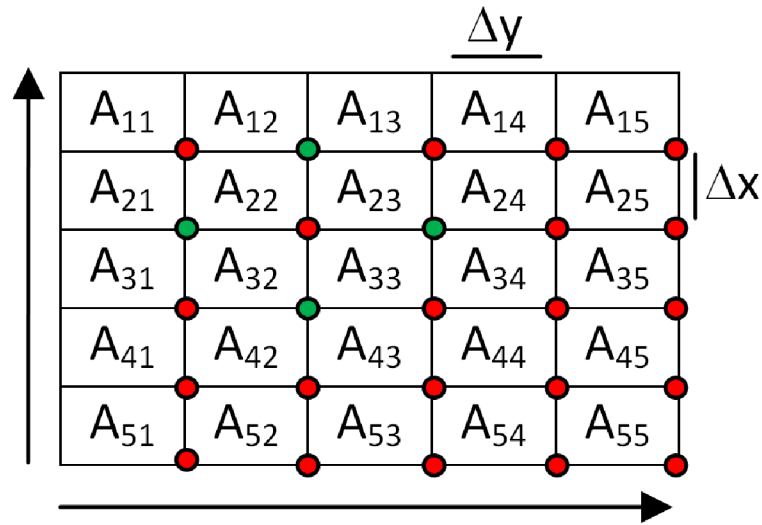
$$\left. \frac{\partial^2 u}{\partial x^2} \right|_{22} \approx \frac{A_{21} - 2A_{22} + A_{23}}{(\Delta x)^2} \quad (4.13)$$

Pro pravou stranu rovnice postupujeme podobným způsobem.

$$\left. \frac{\partial^2 u}{\partial y^2} \right|_{22} \approx \frac{A_{12} - 2A_{22} + A_{32}}{(\Delta y)^2} \quad (4.14)$$

Celkově dostáváme níže uvedený vztah.

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} \approx k(A_{21} + A_{23} + A_{12} + A_{32} - 4A_{22}) = 0 \quad (4.15)$$



Obrázek 4.8: Náhrada derivací pro bod A_{22}

Nyní můžeme získat rovnici pro prvek A_{22} .

$$A_{22} = \frac{A_{21} + A_{23} + A_{12} + A_{32}}{4} \quad (4.16)$$

Tuto rovnici dáme přepíšeme na tvar, který je uvedený níže.

$$u_{21} + u_{23} + u_{12} + u_{32} - 4u_{22} = 0 \quad (4.17)$$

Analogicky bychom postupovali i pro další uzly sítě. Získali bychom tedy soustavu lineárních algebraických rovnic (SLAR). Pro výpočty v systému TKSL je ovšem nutné tuto SLAR transformovat na soustavu diferenciálních rovnic (SDR). Po převodu dostaneme následující rovnici, která je již vhodná pro řešení pomocí TKSL.

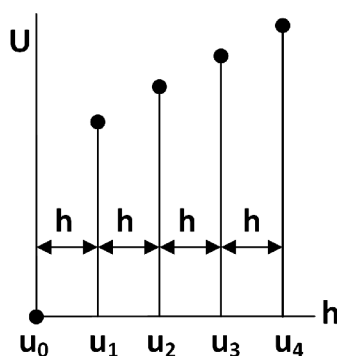
$$u_{21} + u_{23} + u_{12} + u_{32} - 4u_{22} = u'_{22} \quad (4.18)$$

Kapitola 5

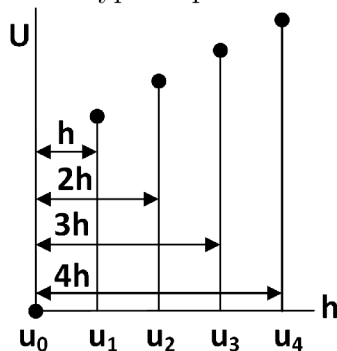
Metody řešení PDR

V této kapitole si ukážeme, jakými metodami lze řešit parciální diferenciální rovnice, bude se jednat o dopřednou metodu, zpětnou metodu a kombinovanou metodu. Setkáme se zde opět s Taylorovou řadou (dále TR) a vysvětlíme si, jak pomocí ní lze numericky řešit parciální diferenciální rovnice. Bližší informace k dané problematice lze najít v publikacích [2]. Pro lepší pochopení problematiky, týkající se Taylorovy řady a vícebodových aproximací, si ukážeme následující příklad.

Příklad 5.1. Víme, že bod $u_0 = 0$ a chceme získat hodnoty (například hodnoty napětí) v bodech u_1, u_2, u_3 a u_4 . Výpočet budeme provádět s daným krokem h . Úlohu budeme nejdříve řešit klasickou TR (viz obrázek 5.1) a poté TR pro pětibodovou aproximaci (viz obrázek 5.2).



Obrázek 5.1: Výpočet pomocí klasické TR



Obrázek 5.2: Výpočet pomocí TR pro pětibodovou aproximaci

Pokud bychom tuto úlohu řešili klasickou Taylorovou řadou, potom bychom získali tyto rovnice. Pro jednoduchost uvažujeme jen prvních pět členů Taylorovy řady.

$$u_1 = u_0 + hu'_0 + \frac{h^2}{2!}u''_0 + \frac{h^3}{3!}u'''_0 + \frac{h^4}{4!}u''''_0 \quad (5.1)$$

$$u_2 = u_1 + hu'_1 + \frac{h^2}{2!}u''_1 + \frac{h^3}{3!}u'''_1 + \frac{h^4}{4!}u''''_1 \quad (5.2)$$

$$u_3 = u_2 + hu'_2 + \frac{h^2}{2!}u''_2 + \frac{h^3}{3!}u'''_2 + \frac{h^4}{4!}u''''_2 \quad (5.3)$$

$$u_4 = u_3 + hu'_3 + \frac{h^2}{2!}u''_3 + \frac{h^3}{3!}u'''_3 + \frac{h^4}{4!}u''''_3 \quad (5.4)$$

Nyní si ukážeme, jak lze řešit zadanou úlohu pomocí vícebodové (v tomto případě pětibodové) aproximace. Opět využijeme Taylorovu řadu.

$$u_1 = u_0 + hu'_0 + \frac{h^2}{2!}u''_0 + \frac{h^3}{3!}u'''_0 + \frac{h^4}{4!}u''''_0 \quad (5.5)$$

$$u_2 = u_0 + 2hu'_0 + \frac{2h^2}{2!}u''_0 + \frac{2h^3}{3!}u'''_0 + \frac{2h^4}{4!}u''''_0 \quad (5.6)$$

$$u_3 = u_0 + 3hu'_0 + \frac{3h^2}{2!}u''_0 + \frac{3h^3}{3!}u'''_0 + \frac{3h^4}{4!}u''''_0 \quad (5.7)$$

$$u_4 = u_0 + 4hu'_0 + \frac{4h^2}{2!}u''_0 + \frac{4h^3}{3!}u'''_0 + \frac{4h^4}{4!}u''''_0 \quad (5.8)$$

□

Srovnáme nyní klasickou TŘ s TŘ pro pětibodovou aproximaci. U klasické TŘ (rovnice 5.1 až 5.4) je jasné vidět, že při výpočtu dalších bodů uvažujeme vždy předcházející bod. Například pokud počítáme hodnotu v bodě u_1 , vycházíme z bodu u_0 , při výpočtu hodnoty v bodě u_2 vycházíme z bodu u_1 , atd. Krok h tedy zůstává stále stejný, protože pro výpočet v daném bodě uvažujeme vždy hodnotu bezprostředně předchozí. Pomocí klasické TŘ získáváme postupně hodnoty prvních derivací pro zadané body u_1, u_2, u_3, u_4 . U TŘ pro pětibodovou aproximaci (rovnice 5.5 až 5.8) si můžeme všimnout hned několika odlišností. Pomocí této modifikace TŘ získáváme první (rovnice 5.5), druhou (rovnice 5.6), třetí (rovnice 5.7) a čtvrtou (rovnice 5.8) derivaci v bodě u_0 . V rovnicích neustále vycházíme z bodu u_0 . Z toho plyne další odlišnost – krok h zde není konstantní, ale postupně se zvětšuje. Krok h tedy nabývá postupně hodnot $h, 2h, 3h, 4h$. Nejdůležitější je, že dostáváme soustavu rovnic $(n-1) \times (n-1)$, kde n označuje, kolikabodovou aproximaci používáme. V našem případě získáme soustavu rovnic o rozměrech 4×4 .

Rovnice 5.5 až 5.8 můžeme přepsat následujícím způsobem. Známou proměnnou u_0 přesuneme na levou stranu rovnice a zároveň jednotlivé členy TŘ ve tvaru $\frac{h^i}{i!}u^{(i)}$, $i \in \{1; 4\}$ označíme postupně $DY1, DY2, DY3$ a $DY4$.

$$u_1 - u_0 = DY1 + DY2 + DY3 + DY4 \quad (5.9)$$

$$u_2 - u_0 = 2DY1 + 2^2DY2 + 2^3DY3 + 2^4DY4 \quad (5.10)$$

$$u_3 - u_0 = 3DY1 + 3^2DY2 + 3^3DY3 + 3^4DY4 \quad (5.11)$$

$$u_4 - u_0 = 4DY1 + 4^2DY2 + 4^3DY3 + 4^4DY4 \quad (5.12)$$

Výše uvedené rovnice lze přepsat do maticového zápisu.

$$\begin{pmatrix} u_1 - u_0 \\ u_2 - u_0 \\ u_3 - u_0 \\ u_4 - u_0 \end{pmatrix} = \begin{pmatrix} 1 & 1^2 & 1^3 & 1^4 \\ 2 & 2^2 & 2^3 & 2^4 \\ 3 & 3^2 & 3^3 & 3^4 \\ 4 & 4^2 & 4^3 & 4^4 \end{pmatrix} \cdot \begin{pmatrix} DY1 \\ DY2 \\ DY3 \\ DY4 \end{pmatrix} \quad (5.13)$$

Nyní chceme získat hodnoty neznámých $DY1, DY2, DY3$ a $DY4$. Pro jejich výpočet platí tato rovnice v maticovém zápisu.

$$\begin{pmatrix} DY1 \\ DY2 \\ DY3 \\ DY4 \end{pmatrix} = \begin{pmatrix} 1 & 1^2 & 1^3 & 1^4 \\ 2 & 2^2 & 2^3 & 2^4 \\ 3 & 3^2 & 3^3 & 3^4 \\ 4 & 4^2 & 4^3 & 4^4 \end{pmatrix}^{-1} \cdot \begin{pmatrix} u_1 - u_0 \\ u_2 - u_0 \\ u_3 - u_0 \\ u_4 - u_0 \end{pmatrix} \quad (5.14)$$

Pokud nyní chceme získat hodnoty derivací u'_0, u''_0, u'''_0 a u''''_0 , musíme je vyjádřit následovně.

$$u'_0 = \frac{DU1}{h} \quad (5.15)$$

$$u''_0 = \frac{DU2}{\frac{h^2}{2!}} \quad (5.16)$$

$$u'''_0 = \frac{DU3}{\frac{h^3}{3!}} \quad (5.17)$$

$$u''''_0 = \frac{DU4}{\frac{h^4}{4!}} \quad (5.18)$$

5.1 Dopředná metoda

Podíváme se blíže na první metodu řešení parciálních diferenciálních rovnic. Princip metody spočívá v tom, že pro výpočet derivace v daném bodě používáme výhradně body následující od aktuálního bodu výpočtu. Metodu budeme ilustrovat na příkladu 5.2. Ještě si zavedeme některé konvence které budeme využívat i u dalších metod. Matice A bude reprezentovat matici koeficientů. Jedná se koeficienty, které jsme získali ze zápisu pomocí Taylorovy řady. Dále zavádíme vektor \vec{b} . Tento vektor reprezentuje hodnoty rozdílů známých vzorků.

Definice 5.1. *Mějme vzorky $f_{-l}, \dots, f_{-1}, f_0, f_1, f_k$ v časech $-lh, \dots, -h, 0, \dots, kh, n = l + k + 1$. Potom je možné použít Taylorův polynom:*

$$\forall i \in -l, -l+1, \dots, -1, 0, 1, \dots, k-1, k : f_i = f_0 + \sum_{m=1}^n \frac{i^m h^m}{m!} f^{(m)}(0). \quad (5.19)$$

Výše uvedený vztah lze zapsat v maticové podobě.

$$\vec{b} = \begin{pmatrix} f_{-l} - f_0 \\ f_{-l+1} - f_0 \\ \dots \\ f_{-1} - f_0 \\ f_1 - f_0 \\ \dots \\ f_{k-1} - f_0 \\ f_k - f_0 \end{pmatrix} \quad (5.20)$$

$$A = \begin{pmatrix} (-l)^1 & (-l)^2 & \dots & (-l)^n \\ (-l+1)^1 & (-l+1)^2 & \dots & (-l+1)^n \\ \dots & \dots & \dots & \dots \\ (-1)^1 & (-1)^2 & \dots & (-1)^n \\ (1)^1 & (1)^2 & \dots & (1)^n \\ \dots & \dots & \dots & \dots \\ (k-1)^1 & (k-1)^2 & \dots & (k-1)^n \\ (k)^1 & (k)^2 & \dots & (k)^n \end{pmatrix} \quad (5.21)$$

Příklad 5.2. Uvažujme strunu, která bude rozdělena řezy u_0 až u_{10} . Naším úkolem je sestavit Taylorovy řady pro první tři body u_0 , u_1 , u_2 . Budeme uvažovat pětibodovou aproximaci.

Při výpočtech postupujeme od začátku struny (bod u_0) směrem ke konci (bod u_{10}). Situaci na struně ilustrují obrázky 5.3, 5.4 a 5.5. Červeně označený bod je bod, ve kterém aktuálně počítáme hodnoty derivací. Zeleně jsou zvýrazněny body, které používáme při výpočtu pomocí dopředné metody. Body, které v aktuálním kroku výpočtu neuvažujeme, jsou označeny černě. Důležité je si uvědomit, že v každém bodě sestavujeme více rovnic, tedy více Taylorových řad. Pro počet rovnic platí, že pokud máme n -bodovou aproximaci, potom počet sestavovaných Taylorových řad v každém bodě bude roven $n - 1$. My jsme se rozhodli využít pětibodovou aproximaci, pro každý bod na struně budeme tedy sestavovat čtyři Taylorovy řady. Z toho plyne, že v každém bodě jsme tedy schopni zjistit první, druhou, třetí i čtvrtou derivaci aktuálně počítaného bodu.

Taylorovy řady pro bod u_0 mají následující tvar, situaci vystihuje obrázek 5.3. Aktuálním bodem výpočtu je tedy bod u_0 označený červeně. V případě dopředné metody a pětibodové aproximace tedy bereme čtyři následující body na struně. Jedná se o zeleně vyznačené body u_1, u_2, u_3, u_4 .

$$u_1 = u_0 + hu'_0 + \frac{h^2}{2!}u''_0 + \frac{h^3}{3!}u'''_0 + \frac{h^4}{4!}u''''_0 \quad (5.22)$$

$$u_2 = u_0 + 2hu'_0 + \frac{2h^2}{2!}u''_0 + \frac{2h^3}{3!}u'''_0 + \frac{2h^4}{4!}u''''_0 \quad (5.23)$$

$$u_3 = u_0 + 3hu'_0 + \frac{3h^2}{2!}u''_0 + \frac{3h^3}{3!}u'''_0 + \frac{3h^4}{4!}u''''_0 \quad (5.24)$$

$$u_4 = u_0 + 4hu'_0 + \frac{4h^2}{2!}u''_0 + \frac{4h^3}{3!}u'''_0 + \frac{4h^4}{4!}u''''_0 \quad (5.25)$$

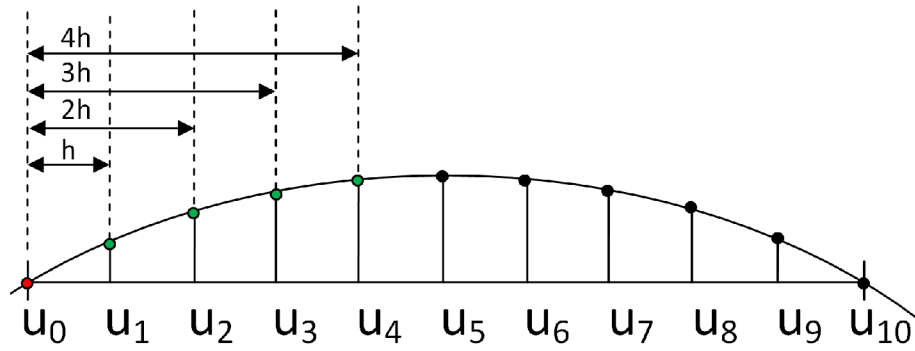
Dále uvedeme Taylorovy řady pro bod u_1 . K těmto rovnicím se váže obrázek 5.4. Analogicky postupujeme dále. Aktuálním bodem se nyní stává bod u_1 , pro výpočet opět vezmeme čtyři následující body u_2, u_3, u_4, u_5 .

$$u_2 = u_1 + hu'_1 + \frac{h^2}{2!}u''_1 + \frac{h^3}{3!}u'''_1 + \frac{h^4}{4!}u''''_1 \quad (5.26)$$

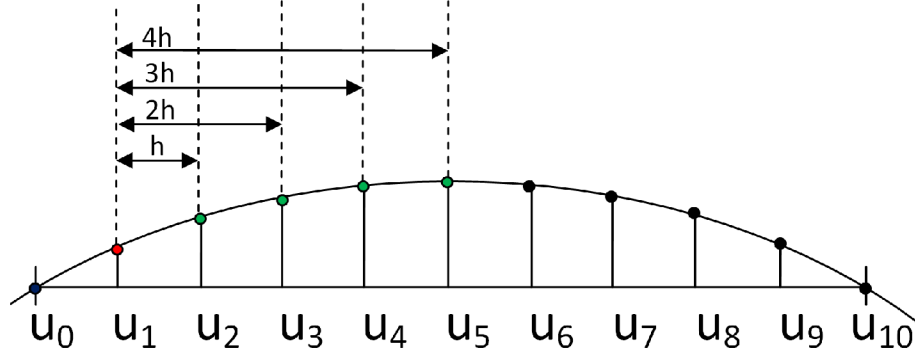
$$u_3 = u_1 + 2hu'_1 + \frac{2h^2}{2!}u''_1 + \frac{2h^3}{3!}u'''_1 + \frac{2h^4}{4!}u''''_1 \quad (5.27)$$

$$u_4 = u_1 + 3hu'_1 + \frac{3h^2}{2!}u''_1 + \frac{3h^3}{3!}u'''_1 + \frac{3h^4}{4!}u''''_1 \quad (5.28)$$

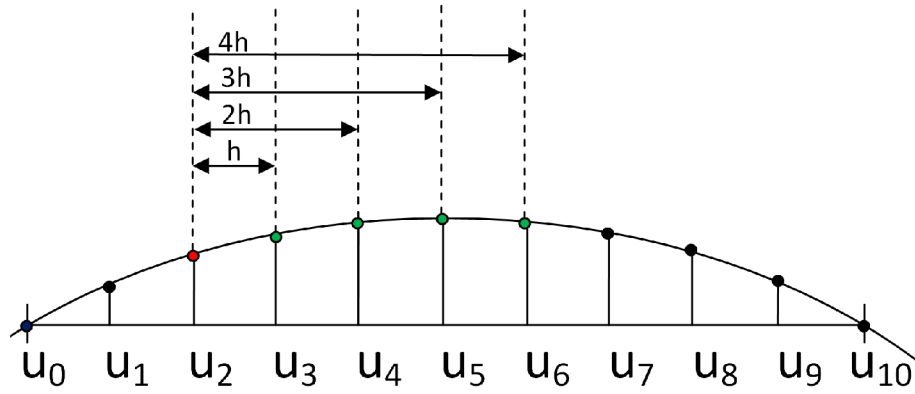
$$u_5 = u_1 + 4hu'_1 + \frac{4h^2}{2!}u''_1 + \frac{4h^3}{3!}u'''_1 + \frac{4h^4}{4!}u''''_1 \quad (5.29)$$



Obrázek 5.3: Dopředná metoda – bod u_0



Obrázek 5.4: Dopředná metoda – bod u_1



Obrázek 5.5: Dopředná metoda – bod u_2

Následují Taylorovy řady pro bod u_2 . Situace je zobrazena na obázku 5.5. V posledním kroku, který si zde ukážeme, je aktuální bod u_2 a následující body jsou u_3, u_4, u_5 a u_6 .

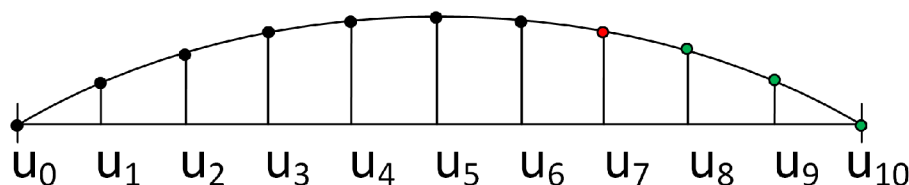
$$u_3 = u_2 + hu'_2 + \frac{h^2}{2!}u''_2 + \frac{h^3}{3!}u'''_2 + \frac{h^4}{4!}u''''_2 \quad (5.30)$$

$$u_4 = u_2 + 2hu'_2 + \frac{2h^2}{2!}u''_2 + \frac{2h^3}{3!}u'''_2 + \frac{2h^4}{4!}u''''_2 \quad (5.31)$$

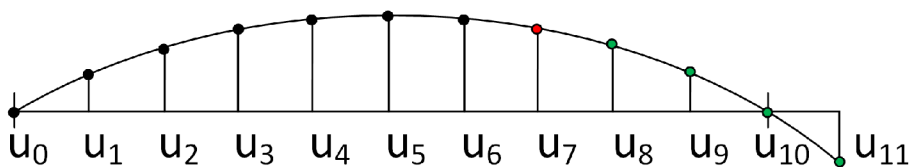
$$u_5 = u_2 + 3hu'_2 + \frac{3h^2}{2!}u''_2 + \frac{3h^3}{3!}u'''_2 + \frac{3h^4}{4!}u''''_2 \quad (5.32)$$

$$u_6 = u_2 + 4hu'_2 + \frac{4h^2}{2!}u''_2 + \frac{4h^3}{3!}u'''_2 + \frac{4h^4}{4!}u''''_2 \quad (5.33)$$

Obdobným způsobem lze samozřejmě postupovat dále. Pokud se dostaneme do situace, kdy již nemáme na pravé straně dostatek bodů pro výpočet, lze zvolit jednu z následujících možností. Buď výpočet ukončíme, protože k dispozici nemáme další body (viz obrázek 5.6) anebo můžeme pro výpočet použít další body za koncovým bodem u_{10} (viz obrázek 5.7).



Obrázek 5.6: Dopředná metoda – nedostatek bodů pro další výpočet



Obrázek 5.7: Dopředná metoda – další bod u_{11}

□

5.2 Zpětná metoda

Představíme si nyní druhou metodu řešení parciálních diferenciálních rovnic. Tato metoda je obdobou dopředné metody, kterou jsme si ukázali výše. Modifikace spočívá v tom, že při výpočtu postupujeme opačným směrem. Pro výpočet derivací v daném bodě se využívají předchozí body od aktuálního bodu výpočtu. Pro snazší pochopení začneme opět příkladem (viz příklad 5.3). Všimněme si, že postupujeme od konce struny (bod u_{10}) směrem k začátku (bod u_0).

Příklad 5.3. Uvažujme strunu, která bude rozdělena opět řezy u_0 až u_{10} . Úkolem bude sestavit Taylorovy řady pro poslední tři body u_{10}, u_9, u_8 , používáme pětibodovou aproximaci.

Pro ilustraci uvádíme obrázky 5.8, 5.9 a 5.10. V každém bodě výpočtu sestavujeme více Taylorových řad. Opět si ukážeme Taylorovy řady pro pětibodovou aproximaci. Významnou odlišností je ovšem to, že se pohybujeme po struně se záporným krokem $-h$.

Taylorovy řady pro bod u_{10} mají následující tvar, situaci pro bod u_{10} vystihuje obrázek 5.8. Aktuálně počítaný bod u_{10} je znázorněn červeně. V případě zpětné metody využíváme k aproximaci derivací předcházející body u_9, u_8, u_7, u_6 , které jsou znázorněny zeleně. Kvůli tomu, že postupujeme se záporným krokem $-h$, znaménka před jednotlivými členy Taylorových řad alternují.

$$u_9 = u_{10} + (-h)u'_{10} + \frac{(-h)^2}{2!}u''_{10} + \frac{(-h)^3}{3!}u'''_{10} + \frac{(-h)^4}{4!}u''''_{10} \quad (5.34)$$

$$u_8 = u_{10} + (-2h)u'_{10} + \frac{(-2h)^2}{2!}u''_{10} + \frac{(-2h)^3}{3!}u'''_{10} + \frac{(-2h)^4}{4!}u''''_{10} \quad (5.35)$$

$$u_7 = u_{10} + (-3h)u'_{10} + \frac{(-3h)^2}{2!}u''_{10} + \frac{(-3h)^3}{3!}u'''_{10} + \frac{(-3h)^4}{4!}u''''_{10} \quad (5.36)$$

$$u_6 = u_{10} + (-4h)u'_{10} + \frac{(-4h)^2}{2!}u''_{10} + \frac{(-4h)^3}{3!}u'''_{10} + \frac{(-4h)^4}{4!}u''''_{10} \quad (5.37)$$

Nyní přistoupíme k bodu u_9 . K aproximaci derivací využíváme v tomto případě body u_8, u_7, u_6 a u_5 . Pro lepší představu uvádíme obrázek 5.9. Taylorovy řady sestavíme následovně.

$$u_8 = u_9 + (-h)u'_9 + \frac{(-h)^2}{2!}u''_9 + \frac{(-h)^3}{3!}u'''_9 + \frac{(-h)^4}{4!}u''''_9 \quad (5.38)$$

$$u_7 = u_9 + (-2h)u'_9 + \frac{(-2h)^2}{2!}u''_9 + \frac{(-2h)^3}{3!}u'''_9 + \frac{(-2h)^4}{4!}u''''_9 \quad (5.39)$$

$$u_6 = u_9 + (-3h)u'_9 + \frac{(-3h)^2}{2!}u''_9 + \frac{(-3h)^3}{3!}u'''_9 + \frac{(-3h)^4}{4!}u''''_9 \quad (5.40)$$

$$u_5 = u_9 + (-4h)u'_9 + \frac{(-4h)^2}{2!}u''_9 + \frac{(-4h)^3}{3!}u'''_9 + \frac{(-4h)^4}{4!}u''''_9 \quad (5.41)$$

Na závěr uvádíme Taylorovy řady pro bod u_8 , vše ukazuje obrázek 5.10. Obdobným způsobem můžeme postupovat dále. Podobně jako u dopředné metody zde platí, že v případě nedostatku bodů z levé strany můžeme buď výpočet ukončit nebo uvažovat pro výpočet i další body před bodem u_0 .

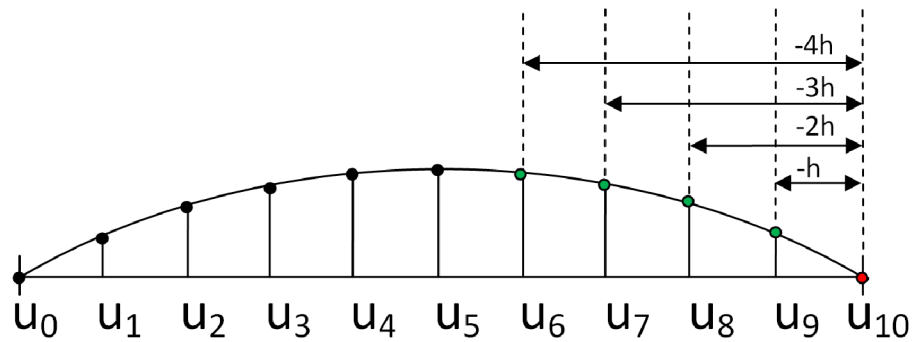
$$u_7 = u_8 + (-h)u'_8 + \frac{(-h)^2}{2!}u''_8 + \frac{(-h)^3}{3!}u'''_8 + \frac{(-h)^4}{4!}u''''_8 \quad (5.42)$$

$$u_6 = u_8 + (-2h)u'_8 + \frac{(-2h)^2}{2!}u''_8 + \frac{(-2h)^3}{3!}u'''_8 + \frac{(-2h)^4}{4!}u''''_8 \quad (5.43)$$

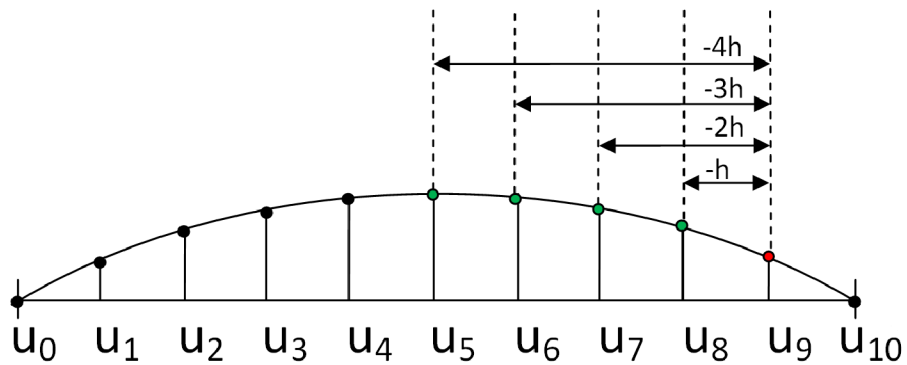
$$u_5 = u_8 + (-3h)u'_8 + \frac{(-3h)^2}{2!}u''_8 + \frac{(-3h)^3}{3!}u'''_8 + \frac{(-3h)^4}{4!}u''''_8 \quad (5.44)$$

$$u_4 = u_8 + (-4h)u'_8 + \frac{(-4h)^2}{2!}u''_8 + \frac{(-4h)^3}{3!}u'''_8 + \frac{(-4h)^4}{4!}u''''_8 \quad (5.45)$$

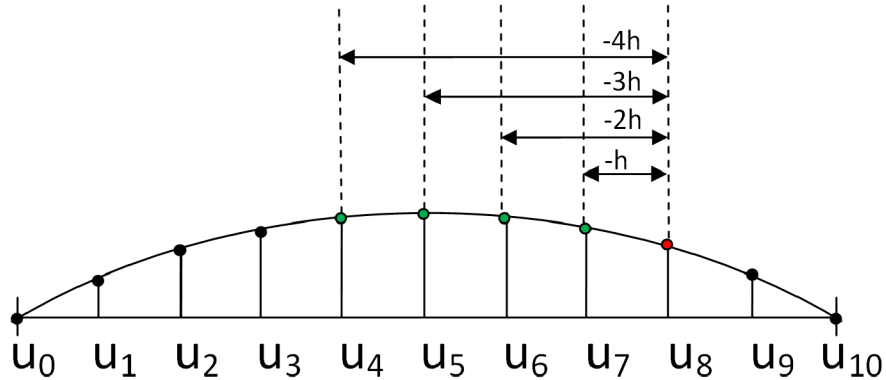
□



Obrázek 5.8: Zpětná metoda – bod u_{10}



Obrázek 5.9: Zpětná metoda – bod u_9



Obrázek 5.10: Zpětná metoda – bod u_8

5.3 Kombinovaná metoda

Třetí metodou řešení PDR je kombinovaná metoda. Jak již název napovídá, bude se jednat o kombinaci předchozích dvou metod. Kombinovaná metoda využívá principu symetrických i nesymetrických diferencí. Pro symetrické difference to znamená, že pro výpočet využíváme stejný počet bodů z levé i pravé strany od aktuálního bodu výpočtu. Změna ovšem nastává v okolí počátečního a koncového bodu struny, kde musíme využít nesymetrické difference. Představme si, že chceme provádět výpočet pomocí pětibodové aproximace. Kombinovaná metoda tedy znamená, že k výpočtu derivací v daném bodě budeme využívat dva body z levé strany a dva body ze strany pravé. Pokud se ovšem nacházíme na počátku struny (bod u_0), potom na levé straně od tohoto bodu nemáme další body k dispozici. V tomto případě bude výpočet probíhat pomocí Dopředné metody a budeme používat čtyři body z pravé strany – body u_1, u_2, u_3, u_4 (viz obrázek 5.11). Důležité je si také uvědomit, že pokud bereme body pro výpočet z pravé strany, je krok h kladný, pokud používáme body ze strany levé, krok je $-h$, tedy záporný.

Podíváme se, jak bude vypadat výpočet v bodě u_1 . Nyní máme k dispozici z levé strany bod u_0 a z pravé strany je nutné vzít body u_2, u_3, u_4 . Dalším bodem je bod u_2 , ve kterém můžeme konečně využívat symetrické difference. Pro výpočet použijeme dva body z levé strany (body u_0, u_1) a dva body z pravé strany (body u_3, u_4). Obdobně tomu bude pro bod u_3 . Z levé strany využijeme body u_1, u_2 , z pravé strany potom body u_4, u_5 . Změna nastane v bodě u_9 , kde je nutné vzít tři body z levé strany (body u_6, u_7, u_8). Z pravé strany pro výpočet použijeme pouze bod u_{10} . Pro poslední bod u_{10} použijeme Zpětnou metodu (viz obrázek 5.8), budeme tedy používat pouze body z levé strany od aktuálně počítaného bodu (body u_9, u_8, u_7, u_6). Obrázek 5.12 ukazuje způsob výpočtu v bodě u_1 , obrázek 5.13 znázorňuje výpočet v bodě u_2 (symetrický vzorec), konečně obrázek 5.14 ilustruje situaci v bodě u_9 .

Nyní si ukážeme, jak se sestavují Taylorovy řady pro kombinovanou metodu. Začneme Taylorovou řadou pro bod u_0 , což odpovídá dopředné metodě.

$$u_1 = u_0 + hu'_0 + \frac{h^2}{2!}u''_0 + \frac{h^3}{3!}u'''_0 + \frac{h^4}{4!}u''''_0 \quad (5.46)$$

$$u_2 = u_0 + 2hu'_0 + \frac{2h^2}{2!}u''_0 + \frac{2h^3}{3!}u'''_0 + \frac{2h^4}{4!}u''''_0 \quad (5.47)$$

$$u_3 = u_0 + 3hu'_0 + \frac{3h^2}{2!}u''_0 + \frac{3h^3}{3!}u'''_0 + \frac{3h^4}{4!}u''''_0 \quad (5.48)$$

$$u_4 = u_0 + 4hu'_0 + \frac{4h^2}{2!}u''_0 + \frac{4h^3}{3!}u'''_0 + \frac{4h^4}{4!}u''''_0 \quad (5.49)$$

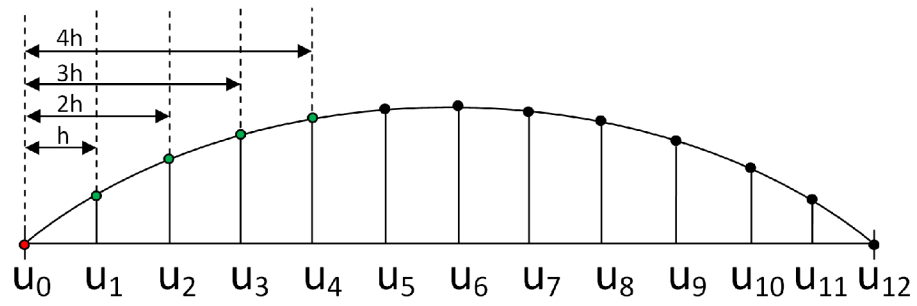
Pro bod u_1 již musíme využít nesymetrické difference. Všimněme si, že u poslední uvedené Taylorovy řady (viz 5.53) používáme krok $-h$, protože bod u_0 se od aktuálního bodu výpočtu u_1 nachází nalevo.

$$u_2 = u_1 + hu'_1 + \frac{h^2}{2!}u''_1 + \frac{h^3}{3!}u'''_1 + \frac{h^4}{4!}u''''_1 \quad (5.50)$$

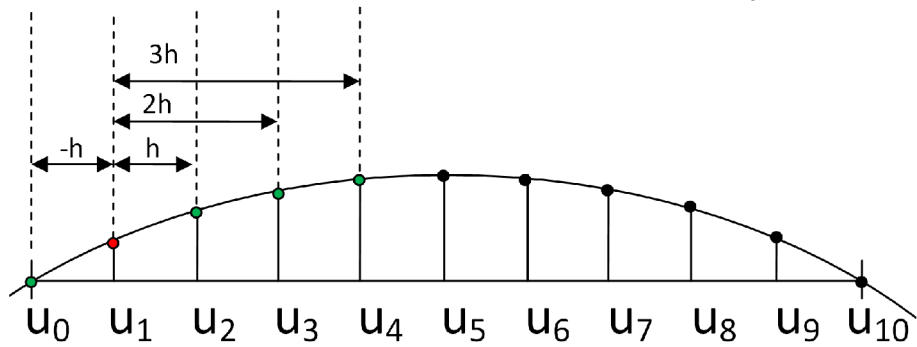
$$u_3 = u_1 + 2hu'_1 + \frac{2h^2}{2!}u''_1 + \frac{2h^3}{3!}u'''_1 + \frac{2h^4}{4!}u''''_1 \quad (5.51)$$

$$u_4 = u_1 + 3hu'_1 + \frac{3h^2}{2!}u''_1 + \frac{3h^3}{3!}u'''_1 + \frac{3h^4}{4!}u''''_1 \quad (5.52)$$

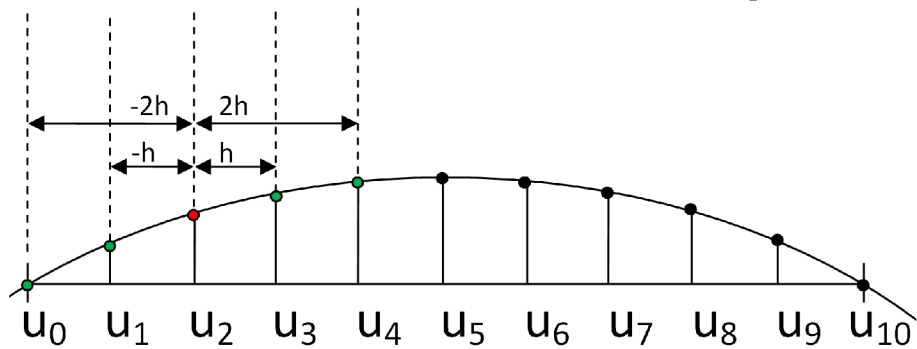
$$u_0 = u_1 + (-h)u'_1 + \frac{(-h)^2}{2!}u''_1 + \frac{(-h)^3}{3!}u'''_1 + \frac{(-h)^4}{4!}u''''_1 \quad (5.53)$$



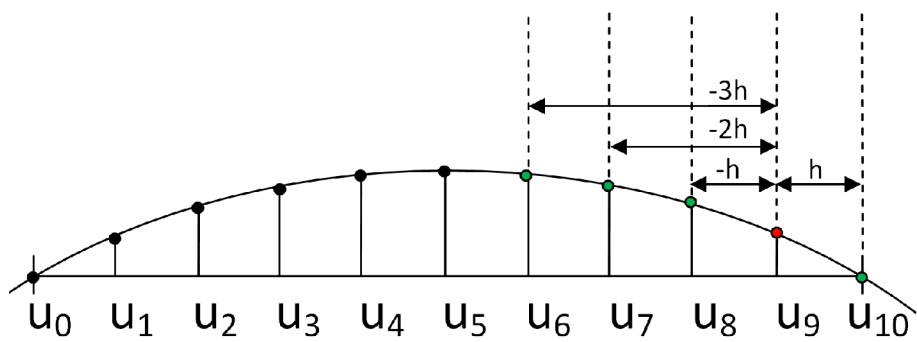
Obrázek 5.11: Kombinovaná metoda – bod u_0



Obrázek 5.12: Kombinovaná metoda – bod u_1



Obrázek 5.13: Kombinovaná metoda – bod u_2



Obrázek 5.14: Kombinovaná metoda – bod u_9

Přejdeme k bodu u_2 , ve kterém využijeme symetrické difference. Zde bude tvar Taylorových řad takový, že dvě z nich budou postupovat s krokem h (viz 5.54 a 5.55) a další dvě s krokem $-h$ (viz 5.56 a 5.57).

$$u_3 = u_2 + hu'_2 + \frac{h^2}{2!}u''_2 + \frac{h^3}{3!}u'''_2 + \frac{h^4}{4!}u''''_2 \quad (5.54)$$

$$u_4 = u_2 + 2hu'_2 + \frac{2h^2}{2!}u''_2 + \frac{2h^3}{3!}u'''_2 + \frac{2h^4}{4!}u''''_2 \quad (5.55)$$

$$u_1 = u_2 + (-h)u'_2 + \frac{(-h)^2}{2!}u''_2 + \frac{(-h)^3}{3!}u'''_2 + \frac{(-h)^4}{4!}u''''_2 \quad (5.56)$$

$$u_0 = u_2 + (-2h)u'_2 + \frac{(-2h)^2}{2!}u''_2 + \frac{(-2h)^3}{3!}u'''_2 + \frac{(-2h)^4}{4!}u''''_2 \quad (5.57)$$

Taylorovy řady pro body u_3 až u_8 by se sestavovaly stejným způsobem, jako Taylorovy řady pro výše uvedený bod u_2 . Přesuneme se nyní do bodu u_9 , kde budeme, podobně jako v bodě u_1 , využívat nesymetrické difference. Body u_6, u_7, u_8 se nachází nalevo od aktuálního bodu u_9 . Proto zde najdeme tři Taylorovy řady s krokem $-h$ (viz 5.59, 5.60, 5.61), poslední bod u_{10} se nachází na pravé straně, takže postupujeme s krokem h (viz 5.58).

$$u_{10} = u_9 + hu'_9 + \frac{h^2}{2!}u''_9 + \frac{h^3}{3!}u'''_9 + \frac{h^4}{4!}u''''_9 \quad (5.58)$$

$$u_8 = u_9 + (-h)u'_9 + \frac{(-h)^2}{2!}u''_9 + \frac{(-h)^3}{3!}u'''_9 + \frac{(-h)^4}{4!}u''''_9 \quad (5.59)$$

$$u_7 = u_9 + (-2h)hu'_9 + \frac{(-2h)h^2}{2!}u''_9 + \frac{(-2h)^3}{3!}u'''_9 + \frac{(-2h)^4}{4!}u''''_9 \quad (5.60)$$

$$u_6 = u_9 + (-3h)u'_9 + \frac{(-3h)^2}{2!}u''_9 + \frac{(-3h)^3}{3!}u'''_9 + \frac{(-3h)^4}{4!}u''''_9 \quad (5.61)$$

Na závěr si ukážeme Taylorovy řady pro bod u_{10} . Jak jsme již zmínili dříve, Taylorovy řady budou mít stejný tvar jako jsme již uváděli u zpětné metody.

$$u_9 = u_{10} + (-h)u'_{10} + \frac{(-h)^2}{2!}u''_{10} + \frac{(-h)^3}{3!}u'''_{10} + \frac{(-h)^4}{4!}u''''_{10} \quad (5.62)$$

$$u_8 = u_{10} + (-2h)u'_{10} + \frac{(-2h)^2}{2!}u''_{10} + \frac{(-2h)^3}{3!}u'''_{10} + \frac{(-2h)^4}{4!}u''''_{10} \quad (5.63)$$

$$u_7 = u_{10} + (-3h)u'_{10} + \frac{(-3h)^2}{2!}u''_{10} + \frac{(-3h)^3}{3!}u'''_{10} + \frac{(-3h)^4}{4!}u''''_{10} \quad (5.64)$$

$$u_6 = u_{10} + (-4h)u'_{10} + \frac{(-4h)^2}{2!}u''_{10} + \frac{(-4h)^3}{3!}u'''_{10} + \frac{(-4h)^4}{4!}u''''_{10} \quad (5.65)$$

5.4 Metody řešení PDR v systému Matlab

V této části si popíšeme, jak se provádí výpočet derivací pomocí systému Matlab. Protože chceme provádět výpočty s vysokou přesností, je vhodné pro tento účel použít *Symbolic Math Toolbox*. Pokud máme tuto knihovnu k dispozici, lze Matlab využívat pro symbolické výpočty (tedy přesné, analytické výpočty). Pro přesné výpočty budeme využívat funkci `vpa()` – což znamená *Variable Precision Arithmetic*. Syntaxe funkce je:

```
R = vpa(A)
R = vpa(A, d)
```

Funkce `vpa()` vypočítá každou položku `A` s přesností na `d` číslic. Požadovanou přesnost lze nastavit příkazem `digits(n)`. Každá položka výsledku je potom symbolickým výrazem. Níže můžeme vidět rozdíl mezi tím, zda použijeme či nepoužijeme funkci `vpa()`.

Příklad 5.4. *Vypočítáme hodnotu funkce sinus v bodě 0,1 nejdříve s použitím funkce `vpa()` a poté bez ní.*

```
>> vpa(sin(0.1))

ans =

0.099833416646828154750181738563696853816509246826171875

>> sin(0.1)

ans =

0.0998
```

□

Budeme uvažovat hyperbolickou PDR, budeme tedy modelovat strunu. Pro tento účel byly vytvořeny dva skripty. První z nich je `diffNum.m`. Tento skript obsahuje funkci `function y = diffNum(l, k, Imin, Imax, h, precision, fname, option)`. Význam parametrů je následující:

- `l` – počet bodů z levé strany od aktuálně počítaného bodu,
- `k` – počet bodů z pravé strany od aktuálně počítaného bodu,
- `Imin` – spodní hranice okna,
- `Imax` – horní hranice okna,
- `h` – krok výpočtu,
- `precision` – přesnost (počet platných číslic, lze nastavit pomocí funkce `digits(precision)`),
- `fname` – jméno souboru pro uložení výsledků (druhých derivací),
- `option` – volba metody,
 - 'f' – Dopředná metoda,

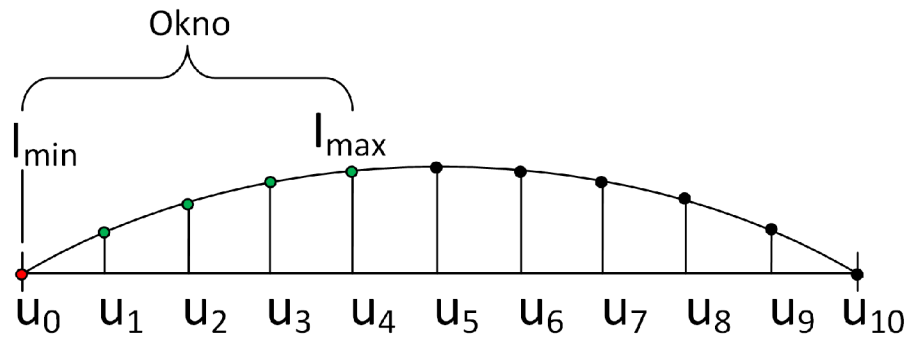
- 'b' – Zpětná metoda,
- 'c' – Kombinovaná metoda.

Činnost skriptu můžeme shrnout do následujících kroků:

- vygeneruje hodnoty funkce sinus v intervalu $\langle 0; \pi \rangle$ s velkou přesností (za pomoci funkce `vpa()`, `precision = 100`) a s daným krokem `h`,
- připraví matici koeficientů A ,
- připraví vektor \vec{b} , jedná se o vektor rozdílů funkčních hodnot funkce sinus,
- vypočte inverzní matici A^{-1} ,
- vypočítá jednotlivé členy Taylorovy řady, $DY = A^{-1} \cdot \vec{b}$,
- zajímají nás druhé derivace, takže získáme druhou derivaci pro aktuálně počítaný bod $y = \frac{DY(2)}{\frac{h^2}{2!}}$.
- vypočítanou hodnotu druhé derivace v daném bodě zapíšeme do výstupního souboru

Druhý skript nese název `diffRun.m`. Tento skript volá výše uvedený skript `diffNum.m` a zajišťuje posun po struně (funkce sinus). Uvnitř skriptu se nachází funkce `function diffRun(approx, h, precision, fname, option, draw, result)`, parametry skriptu jsou následující.

- `approx` – volba n-bodové aproximace,
- `h` – krok výpočtu,
- `precision` – přesnost (počet platných číslic, lze nastavit pomocí funkce `digits(precision)`),
- `fname` – jméno souboru pro uložení výsledků (druhých derivací),
- `option` – volba metody,
- `draw` – nastavení formy grafického výstupu,
 - 0 – žádný grafický výstup,
 - 1 – zobrazí se analytické a numerické řešení,
 - 2 – zobrazí se pouze chyba výpočtu,
 - 3 – zobrazí se analytické řešení, numerické řešení a také chyba výpočtu,
- `result` – nastavení výpisů výsledků,
 - 0 – žádný výpis výsledků,
 - 1 – zobrazí se hodnoty druhých derivací,
 - 2 – zobrazí se hodnoty chyby výpočtu (odchylka numerického a analytického řešení).



Obrázek 5.15: Vysvětlení pojmu *okno*

V tomto skriptu tedy probíhá především nastavování mezí okna pro výpočet – parametry I_{\min} a I_{\max} a také nastavování parametrů l a k , které udávají počet bodů z levé a pravé strany. Pro jistotu uvádíme obrázek 5.15, který vysvětluje, co je pojmem *okno* myšleno. Skripty jsou tedy společné pro všechny tři metody, volbu metody zajišťuje parametr `option`. Důležité je si uvědomit, že vektor \vec{b} se při každém posunutí okna mění, protože počítáme rozdíly funkčních hodnot funkce sinus v různých bodech. Naoproti tomu inverzní matice se vždy měnit nemusí. Nyní si blíže vysvětlíme, jak probíhá výpočet v systému Matlab pro jednotlivé metody. Zaměříme se na tvar matice A a inverzní matice A^{-1} .

V případě dopředné metody spustíme výpočet voláním skriptu `diffRun(5, vpa(0.01), 100, 'out.txt', 'f', 0, 1)`, využíváme pětibodovou aproximaci, krok $h=0.01$. V tomto případě budeme volat skript `diffNum.m` s parametry $l=0$ a $k=4$, protože uvažujeme pouze body z pravé strany od aktuálního bodu (viz obrázky 5.3, 5.4 a 5.5). Inverzní matice k matici A_1 bude mít následující podobu, označíme ji A_1^{-1} .

$$A_1 = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 2 & 4 & 8 & 16 \\ 3 & 9 & 27 & 81 \\ 4 & 16 & 64 & 256 \end{pmatrix} \quad (5.66)$$

$$A_1^{-1} = \begin{pmatrix} 4,0 & -3,0 & 1,333 & -0,25 \\ -4,333 & 4,75 & -2,333 & 0,458 \\ 1,5 & -2,0 & 1,166 & -0,25 \\ -0,166 & 0,25 & -0,166 & 0,042 \end{pmatrix} \quad (5.67)$$

Pro zpětnou metodu provedeme výpočet pomocí volání skriptu `diffRun(5, vpa(0.01), 100, 'out.txt', 'b', 0, 1)`. Skript `diffNum.m` je volán s parametry $l=4$, $k=0$. Inverzní matice k matici A_2 je následující (označíme ji A_2^{-1}).

$$A_2 = \begin{pmatrix} -4 & 16 & -64 & 256 \\ -3 & 9 & -27 & 81 \\ -2 & 4 & -8 & 16 \\ -1 & 1 & -1 & 1 \end{pmatrix} \quad (5.68)$$

$$A_2^{-1} = \begin{pmatrix} 0,25 & -1,333 & 3,0 & -4,0 \\ 0,458 & -2,333 & 4,75 & -4,333 \\ 0,25 & -1,166 & 2,0 & -1,5 \\ 0,042 & -0,166 & 0,25 & -0,166 \end{pmatrix} \quad (5.69)$$

V případě kombinované metody využijeme všechny možné tvary inverzní matice, které lze pro pětibodovou aproximaci sestavit. Jedná se o kombinace parametrů l a k . Pro přehlednost uvádíme následující tabulku 5.1.

Tabulka 5.1: Dostupné kombinace parametrů l a k pro tvorbu inverzních matic

parametr l	parametr k	poznámka
0	4	dopředná/kombinovaná metoda
1	3	kombinovaná metoda
2	2	kombinovaná metoda
3	1	kombinovaná metoda
4	0	zpětná/kombinovaná metoda

Pro úplnost uvádíme zbývající inverzní matice. Inverzní matice A_3^{-1} k matici A_3 pro hodnoty parametrů $l=1$, $k=3$, inverzní matice A_4^{-1} k matici A_4 pro $l=2$, $k=2$ a inverzní matice A_5^{-1} k matici A_5 pro $l=3$, $k=1$.

$$A_3 = \begin{pmatrix} -1 & 1 & -1 & 1 \\ 2 & 4 & 8 & 16 \\ 3 & 9 & 27 & 81 \\ 4 & 16 & 64 & 256 \end{pmatrix} \quad (5.70)$$

$$A_3^{-1} = \begin{pmatrix} -0,25 & 1,5 & -0,5 & 0,083 \\ 0,458 & 0,25 & 0,166 & -0,042 \\ -0,25 & -1,0 & 0,5 & -0,083 \\ 0,042 & 0,25 & -0,166 & 0,042 \end{pmatrix} \quad (5.71)$$

$$A_4 = \begin{pmatrix} -2 & 4 & -8 & 16 \\ -1 & 1 & -1 & 1 \\ 3 & 9 & 27 & 81 \\ 4 & 16 & 64 & 256 \end{pmatrix} \quad (5.72)$$

$$A_4^{-1} = \begin{pmatrix} 0,083 & -0,666 & 0,666 & -0,083 \\ -0,042 & 0,666 & 0,666 & -0,042 \\ 0,083 & -0,5 & 1,0 & 0,25 \\ -0,083 & -0,166 & -0,166 & 0,042 \end{pmatrix} \quad (5.73)$$

$$A_5 = \begin{pmatrix} -3 & 9 & -27 & 81 \\ -2 & 4 & -8 & 16 \\ -1 & 1 & -1 & 1 \\ 4 & 16 & 64 & 256 \end{pmatrix} \quad (5.74)$$

$$A_5^{-1} = \begin{pmatrix} 0,25 & -1,333 & 3,0 & -4,0 \\ 0,458 & -2,333 & 4,75 & -4,333 \\ 0,25 & -1,166 & 2,0 & -1,5 \\ 0,042 & -0,166 & 0,25 & -0,166 \end{pmatrix} \quad (5.75)$$

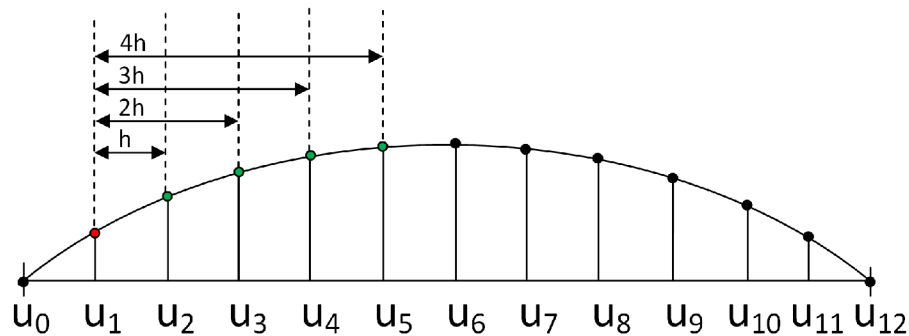
5.5 Metody řešení PDR v systému TKSL

V této podkapitole si ukážeme, jak lze řešit parciální diferenciální rovnice v systému TKSL. Opět se zaměříme na hyperbolickou parciální diferenciální rovnici. Z podkapitoly 3.1.2 víme, že v systému TKSL je zapotřebí zavádět substituce, protože systém TKSL dokáže řešit diferenciální rovnice pouze prvního řádu.

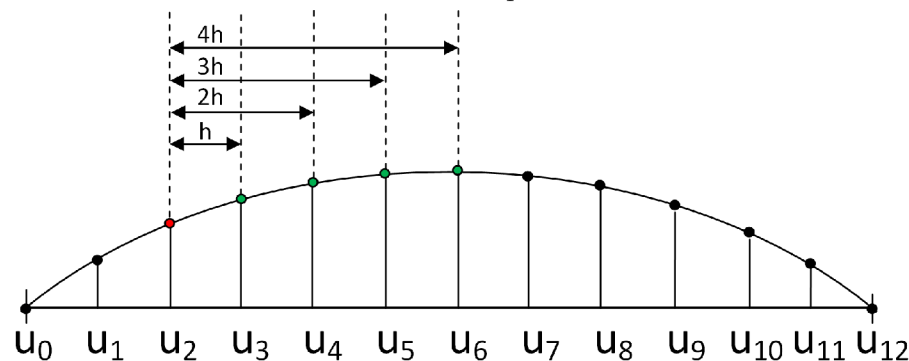
5.5.1 Dopředná metoda v systému TKSL

Dopředná metoda se vyznačuje tím, že pro svůj výpočet využívá pouze body, které se nacházejí směrem doprava od aktuálního bodu. Pro lepší pochopení problematiky uvedeme příklad 5.5.

Příklad 5.5. Máme strunu, která je rozdělena dvanácti řezy, uvažujeme pětibodovou aproximaci, integrační krok metody $dt = 0.01$, přesnost $eps = 1e - 20$. Odpovídající zdrojový kód je uveden níže.



Obrázek 5.16: Pětibodová aproximace – bod u_1



Obrázek 5.17: Pětibodová aproximace – bod u_2

```

var
  u0, u1, u2, u3, u4, u5, u6, u7, u8, u9, u10, u11, u12,
  v1, v2, v3, v4, v5, v6, v7, v8, v9, v10, v11;
Const
  PR = 12,
  A~ = PR*PR/12,
  tmax = 10, dt = 0.01,
  eps = 1e-20,
  PI = 3.141592653589793238462643383279502884197
  169399375105820974944592307816406286208998628034825342117068;
system
  u0 = 0;
  u12 = 0;
  v1' = A~* -4.333*(u2-u1) + 4.75*(u3-u1) - 2.333*(u4-u1) + 0.458*(u5-u1)
    &sin(PI*1/PR);
  v2' = A~* -4.333*(u3-u2) + 4.75*(u4-u2) - 2.333*(u5-u2) + 0.458*(u6-u2)
    &sin(PI*2/PR);
  v3' = A~* -4.333*(u4-u3) + 4.75*(u5-u3) - 2.333*(u6-u3) + 0.458*(u7-u3)
    &sin(PI*3/PR);
  v4' = A~* -4.333*(u5-u4) + 4.75*(u6-u4) - 2.333*(u7-u4) + 0.458*(u8-u4)
    &sin(PI*4/PR);
  v5' = A~* -4.333*(u6-u5) + 4.75*(u7-u5) - 2.333*(u8-u5) + 0.458*(u9-u5)
    &sin(PI*5/PR);
  v6' = A~* -4.333*(u7-u6) + 4.75*(u8-u6) - 2.333*(u9-u6) + 0.458*(u10-u6)
    &sin(PI*6/PR);
  v7' = A~* -4.333*(u8-u7) + 4.75*(u9-u7) - 2.333*(u10-u7) + 0.458*(u11-u7)
    &sin(PI*7/PR);
  v8' = A~* -4.333*(u9-u8) + 4.75*(u10-u8) - 2.333*(u11-u8) + 0.458*(u12-u8)
    &sin(PI*8/PR);
  v9' = A~* 0.458*(u9-u8) + 0.25*(u10-u9) + 0.166*(u11-u9) - 0.042*(u12-u9)
    &sin(PI*9/PR);
  v10' = A~* -0.042*(u10-u9) - 0.666*(u10-u8) + 0.666*(u11-u10) - 0.042*(u12-u10)
    &sin(PI*10/PR);
  v11' = A~* 0.458*(u11-u10) - 2.333*(u11-u9) + 4.75*(u11-u8) - 4.333*(u12-u11)
    &sin(PI*11/PR);
  u1' = v1 &0;
  u1' = v1 &0;
  u2' = v2 &0;
  u3' = v3 &0;
  u4' = v4 &0;
  u5' = v5 &0;
  u6' = v6 &0;
  u7' = v7 &0;
  u8' = v8 &0;
  u9' = v9 &0;
  u10' = v10 &0;
  u11' = v11 &0;
sysend.

```

□

Body u_0 a u_{12} jsou nastaveny na hodnotu 0, protože se jedná o koncové body, v nichž je struna ukotvena. Body u_1 až u_{11} poté popíšeme pomocí diferenčních vztahů. Situaci v bodě u_1 vystihuje obrázek 5.16. Výpočet bude probíhat na základě vzorků u_2, u_3, u_4, u_5 , šířka okna je tedy pět, protože aproximaci uvažujeme pětibodovou. Nyní bereme postupně vzorky od aktuálně počítaného bodu u_1 . Zdrojový kód pro systém TKSL byl vygenerován programem, který vznikl v rámci diplomové práce, podrobněji se mu budeme věnovat v kapitole 7.

Koeficienty před závorkami zde uvádíme zaokrouhleny na tři desetinná místa. Jedná se o koeficienty inverzní matice k matici A , tedy A^{-1} . Jak jsme již uvedli výše, pomocí skriptů jsme schopni počítat členy Taylorovy řady dle vztahu $DY = A^{-1} \cdot \vec{b}$. Vektor \vec{b} představuje rozdíly hodnot vzorků, které využíváme k výpočtu. V rovnici pro v_1' má vektor \vec{b} tvar uvedený níže. Všimněme si, že koeficienty jsou u rovnic v_1' až v_8' stejné. Je tomu tak z toho důvodu, že při výpočtech neuvažujeme žádný bod z levé strany a naopak čtyři body ze strany pravé. To znamená, že před závorky umísťujeme stále stejné koeficienty inverzní matice A_1^{-1} , viz 5.67.

Změna nastává až v rovnicích pro v_9' , v_{10}' a v_{11}' . Zde nemáme dostatek bodů na pravé straně. Tento problém vyřešíme tak, že rovnice pro tyto poslední tři body řešíme kombinovanou metodou (viz 5.3). Vezměme nejdříve situaci v rovnici v_9' . V tomto případě musíme uvažovat jeden bod z levé strany, čímž získáme výraz $(u_9 - u_8)$ a potom tři body ze strany pravé, čímž získáváme závorky $(u_{10} - u_9)$, $(u_{11} - u_9)$, $(u_{12} - u_9)$. Koeficienty získáme z druhého řádku inverzní matice 5.71. Nyní přistoupíme k rovnici pro v_{10}' . Zde musíme brát dva body z levé strany, získáme závorky $(u_{10} - u_9)$, $(u_{10} - u_8)$ a dva body z pravé strany, získáme rovnice $(u_{11} - u_{10})$, $(u_{12} - u_{10})$. V tomto případě jsme koeficienty získali z druhého řádku inverzní matice 5.73. Konečně v poslední rovnici pro v_{11}' uvažujeme tři body z levé strany, získáváme $(u_{11} - u_{10})$, $(u_{11} - u_9)$, $(u_{11} - u_8)$ a jeden bod ze strany pravé, pak získáváme $(u_{12} - u_{11})$. Koeficienty získáme z druhého řádku inverzní matice 5.75.

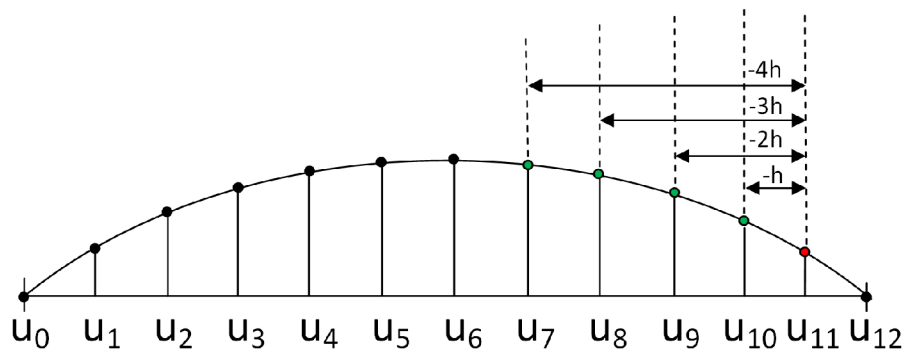
Můžeme si tedy všimnout, že poslední tři rovnice pro v_9' , v_{10}' a v_{11}' mají před závorkami umístěny odlišné koeficienty než rovnice předchozí, což je dáno tím, že využíváme jiný počet bodů z levé a pravé strany. Hodnoty inverzních matic jsou zaokrouhleny na tři desetinná místa. Pro výpočet vždy používáme druhé řádky inverzních matic, protože chceme vypočítat druhé derivace v bodech daných krokem h . Pokud bychom v daných bodech chtěli zjistit například čtvrté derivace, potom bychom používali čtvrté řádky těchto inverzních matic.

$$b = \begin{pmatrix} u_1 - u_0 \\ u_2 - u_0 \\ u_3 - u_0 \\ u_4 - u_0 \end{pmatrix} \quad (5.76)$$

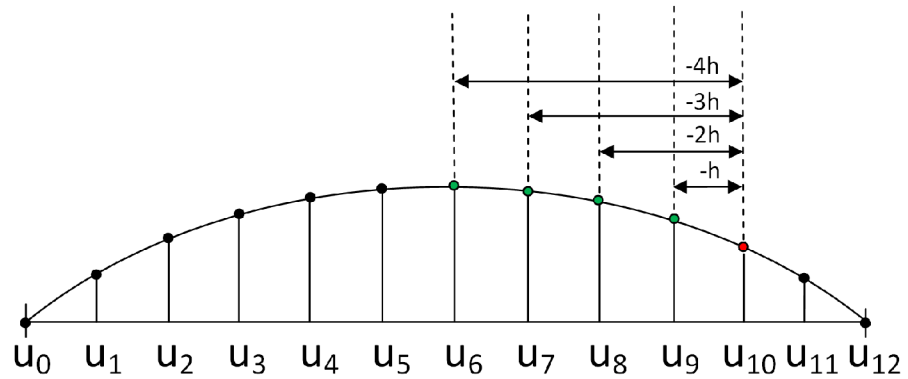
5.5.2 Zpětná metoda v systému TKSL

Bereme v úvahu opět hyperbolickou parciální diferenciální rovnici. Podobně jako u dopředné metody začneme příkladem 5.6.

Příklad 5.6. *Zadání příkladu je stejné jako u dopředné metody. Struna je rozdělena dvanácti řezy, aproximaci uvažujeme pětibodovou. Integrační krok metody je $dt = 0.01$ a přesnost $eps = 1e - 20$. Odpovídající zdrojový kód můžeme vidět níže.*



Obrázek 5.18: Pětibodová aproximace – bod u_{11}



Obrázek 5.19: Pětibodová aproximace – bod u_{10}

```

var
  u0, u1, u2, u3, u4, u5, u6, u7, u8, u9, u10, u11, u12,
  v1, v2, v3, v4, v5, v6, v7, v8, v9, v10, v11;
Const
  PR = 12,
  A~ = PR*PR/12,
  tmax = 10, dt = 0.01,
  eps = 1e-20,
  PI = 3.141592653589793238462643383279502884197
  169399375105820974944592307816406286208998628034825342117068;
system
  u0 = 0;
  u12 = 0;
  v11' = A~* 0.458*(u11-u10) - 2.333*(u11-u9) + 4.75*(u11-u8) + 0.458*(u11-u7)
    &sin(PI*11/PR);
  v10' = A~* 0.458*(u10-u9) - 2.333*(u10-u8) + 4.75*(u10-u7) + 0.458*(u10-u6)
    &sin(PI*10/PR);
  v9' = A~* 0.458*(u9-u8) - 2.333*(u9-u7) + 4.75*(u9-u6) + 0.458*(u9-u5)
    &sin(PI*9/PR);
  v8' = A~* 0.458*(u8-u7) - 2.333*(u8-u6) + 4.75*(u8-u5) + 0.458*(u8-u4)
    &sin(PI*8/PR);
  v7' = A~* 0.458*(u7-u6) - 2.333*(u7-u5) + 4.75*(u7-u4) + 0.458*(u7-u3)
    &sin(PI*7/PR);
  v6' = A~* 0.458*(u6-u5) - 2.333*(u6-u4) + 4.75*(u6-u3) + 0.458*(u6-u2)
    &sin(PI*6/PR);
  v5' = A~* 0.458*(u5-u4) - 2.333*(u5-u3) + 4.75*(u5-u2) + 0.458*(u5-u1)
    &sin(PI*5/PR);
  v4' = A~* 0.458*(u4-u3) - 2.333*(u4-u2) + 4.75*(u4-u1) + 0.458*(u4-u0)
    &sin(PI*4/PR);
  v3' = A~* 0.458*(u3-u2) - 2.333*(u3-u1) + 4.75*(u3-u0) - 4.333*(u4-u3)
    &sin(PI*3/PR);
  v2' = A~* -0.042*(u2-u1) + 0.666*(u2-u0) + 0.666*(u3-u2) - 0.042*(u4-u2)
    &sin(PI*2/PR);
  v1' = A~* 0.458*(u1-u0) + 0.25*(u2-u1) + 0.166*(u3-u1) - 0.042*(u4-u1)
    &sin(PI*1/PR);
  u1' = v1 &0;
  u1' = v1 &0;
  u2' = v2 &0;
  u3' = v3 &0;
  u4' = v4 &0;
  u5' = v5 &0;
  u6' = v6 &0;
  u7' = v7 &0;
  u8' = v8 &0;
  u9' = v9 &0;
  u10' = v10 &0;
  u11' = v11 &0;
sysend.

```

□

Body u_0 a u_{12} jsou opět nastaveny na hodnotu 0, protože se jedná o body ukotvení struny. Body u_{11} až u_1 popíšeme pomocí diferenčních vztahů. Situaci v bodě u_{11} vystihuje obrázek 5.18. Výpočet bude probíhat na základě vzorků u_{10}, u_9, u_8, u_7 , šířka okna je tedy pět, protože aproximaci uvažujeme pětibodovou. Situace v dalším bodě u_{10} , je znázorněna na obrázku 5.19. Koeficienty jsou zaokrouhleny na tři desetinná místa a byly opět získány pomocí programu, který vznikl v rámci diplomové práce. Pro rovnice v_{11}' až v_1' platí, že využíváme inverzní matici A_2^{-1} , viz 5.69, protože pro výpočet využíváme vždy čtyři body z levé strany od aktuálního bodu výpočtu.

Pro rovnice v_3' , v_2' a v_1' musíme využít kombinovanou metodu. Postup je obdobný jako u dopředné metody. Pro rovnici v_3' platí, že z levé strany vezmeme pouze tři body, získáváme výrazy (u_3-u_2) , (u_3-u_1) a (u_3-u_0) . Poté vezmeme jeden bod z pravé strany a dostáváme výraz (u_4-u_3) . Koeficienty jsou získány z druhého řádku inverzní matice 5.75. Pro rovnici v_2' je situace taková, že vezmeme dva body z levé strany a dva body z pravé strany, využíváme inverzní matici 5.73. Nakonec již uvažujeme pouze jeden bod z levé strany a tři body ze strany pravé, koeficienty získáme z druhého řádku inverzní matice 5.71.

5.5.3 Kombinovaná metoda v systému TKSL

Nakonec uvádíme, jak se řeší kombinovaná metoda v systému TKSL. Uvedeme ihned příklad 5.7. V tomto případě využijeme všechny možné tvary inverzní matice (viz tabulka 5.1).

Příklad 5.7. *Struna je rozdělena dvanácti řezy, využíváme pětibodovou aproximaci. Integrační krok metody je nastaven na hodnotu $dt = 0.01$, přesnost $eps = 1e - 20$. Zdrojový kód pro systém TKSL je uveden níže.*

```
var
  u0, u1, u2, u3, u4, u5, u6, u7, u8, u9, u10, u11, u12,
  v1, v2, v3, v4, v5, v6, v7, v8, v9, v10, v11;
Const
  PR = 12,
  A~ = PR*PR/12,
  tmax = 10,
  dt = 0.01,
  eps = 1e-20,
  PI = 3.141592653589793238462643383279502884197
  169399375105820974944592307816406286208998628034825342117068;
system
  u0 = 0;
  u12 = 0;
  v1' = A~* 0.458*(u1-u0) + 0.25*(u2-u1) + 0.166*(u3-u1) - 0.042*(u4-u1)
    &sin(PI*1/PR);
  v2' = A~* 0.458*(u2-u1) + 0.25*(u2-u0) + 0.166*(u3-u2) - 0.042*(u4-u2)
    &sin(PI*2/PR);
  v3' = A~* -0.042*(u3-u2) + 0.666*(u3-u1) + 0.666*(u4-u3) - 0.042*(u5-u3)
    &sin(PI*3/PR);
  v4' = A~* -0.042*(u4-u3) + 0.666*(u4-u2) + 0.666*(u5-u4) - 0.042*(u6-u4)
    &sin(PI*4/PR);
```

```

v5' = A~* -0.042*(u5-u4) + 0.666*(u5-u3) + 0.666*(u6-u5) - 0.042*(u7-u5)
      &sin(PI*5/PR);
v6' = A~* -0.042*(u6-u5) + 0.666*(u6-u4) + 0.666*(u7-u6) - 0.042*(u8-u6)
      &sin(PI*6/PR);
v7' = A~* -0.042*(u7-u6) + 0.666*(u7-u5) + 0.666*(u8-u7) - 0.042*(u9-u7)
      &sin(PI*7/PR);
v8' = A~* -0.042*(u8-u7) + 0.666*(u8-u6) + 0.666*(u9-u8) - 0.042*(u10-u8)
      &sin(PI*8/PR);
v9' = A~* -0.042*(u9-u8) + 0.666*(u9-u7) + 0.666*(u10-u9) - 0.042*(u11-u9)
      &sin(PI*9/PR);
v10' = A~* -0.042*(u10-u9) + 0.166*(u10-u8) + 0.666*(u11-u10) - 0.042*(u12-u10)
      &sin(PI*10/PR);
v11' = A~* 0.458*(u11-u10) - 2.333*(u11-u9) + 4.75*(u11-u8) - 4.333*(u12-u11)
      &sin(PI*11/PR);
u1' = v1 &0;
u1' = v1 &0;
u2' = v2 &0;
u3' = v3 &0;
u4' = v4 &0;
u5' = v5 &0;
u6' = v6 &0;
u7' = v7 &0;
u8' = v8 &0;
u9' = v9 &0;
u10' = v10 &0;
u11' = v11 &0;
sysend.

```

□

Kombinovaná metoda se vyznačuje tím, že v sobě sdružuje jak dopřednou, tak i zpětnou metodu. Opět uvažujeme celkem dvanáct řezů na struně a pětibodovou aproximaci. Výpočet začíná v bodě u_1 , čemuž odpovídá rovnice v_1' . Zde bereme jeden bod z levé strany, dostáváme výraz (u_1-u_0) , poté vezmeme tři body z pravé strany, získáváme (u_2-u_1) , (u_3-u_1) a (u_4-u_1) . V bodě u_2 je situace jiná, máme k dispozici již dva body z levé strany, dostáváme výrazy (u_2-u_1) , (u_2-u_0) a dva body ze strany levé, čímž dostáváme (u_3-u_2) , (u_4-u_2) . Pro rovnice v_3' až v_{10}' je postup stejný, vždy vezmeme dva body z levé strany a dva body z pravé strany. Změna nastává až v poslední rovnici pro v_{11}' , zde musíme uvažovat tři body z levé strany, získáváme $(u_{11}-u_{10})$, $(u_{11}-u_9)$ a $(u_{11}-u_8)$ a jeden bod ze strany pravé, čímž získáme $(u_{12}-u_{11})$. Pro lepší názornost jsou k dispozici obrázky 5.12, 5.13 a 5.14.

5.6 Přesnost výpočtu

Nyní se zaměříme na přesnost výpočtu pomocí dopředné metody. Budeme porovnávat hodnoty druhých derivací vypočtené dopřednou metodou s řešením analytickým. Struna je modelována funkcí $f(x) = \sin(x)$. Pro přehlednost uvádíme první až čtvrtou derivaci funkce $f(x)$.

$$f'(x) = \cos(x) \quad (5.77)$$

$$f''(x) = -\sin(x) \quad (5.78)$$

$$f'''(x) = -\cos(x) \quad (5.79)$$

$$f^{(4)}(x) = \sin(x) \quad (5.80)$$

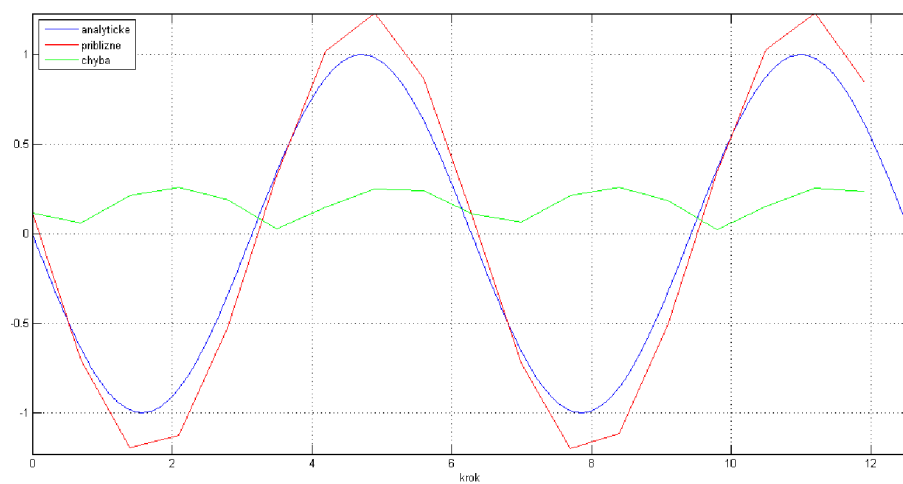
Nyní si povíme, jaké parametry ovlivňují přesnost výpočtu. Prvním parametrem je krok výpočtu. Čím menší bude krok výpočtu, tím přesnější výsledky obdržíme. Ilustraci této skutečnosti znázorňují obrázky 5.20 až 5.26. Grafy jasně ukazují, že čím menší krok zvolíme, tím více se přibližné řešení získané dopřednou metodou (červená křivka) blíží analytickému řešení (modrá křivka) a klesá tedy chyba výpočtu (v grafu znázorněna zelenou křivkou). Pro větší přehlednost byly grafy funkce sinus vykresleny v intervalu $\langle 0; 4\pi \rangle$. Skutečně jako výstup dopředné metody dostáváme grafy, které aproximují druhou derivaci funkce sinus, tedy $f''(x) = -\sin(x)$.

Dalším parametrem ovlivňujícím přesnost výpočtu je volba n -bodové aproximace. Platí, že čím je n vyšší – tedy čím více bodů k aproximaci používáme, tím je výpočet přesnější. Volba n -bodové aproximace souvisí s volbou kroku výpočtu. Pokud zvýšíme n , potom si můžeme dovolit zvětšit krok výpočtu, aniž by se snížila přesnost řešení. Volba n -bodové aproximace ovlivňuje řády derivací, které získáme. Pokud zvolíme nám již známou pětibodovou aproximaci, potom jsme schopni v každém bodě vypočítat derivace až čtvrtého řádu. Pokud bychom například použili padesátibodovou aproximaci, lze získat v každém bodě až derivaci čtyřicátého devátého řádu.

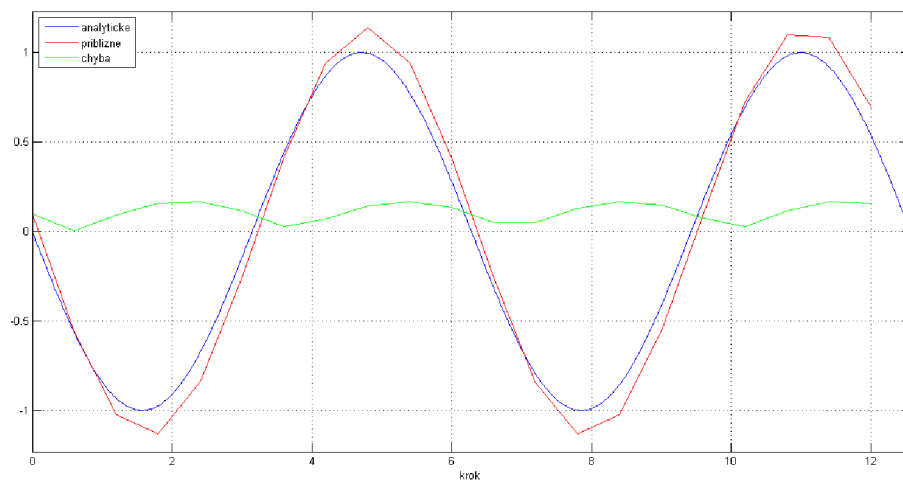
Nyní se podíváme, jak jsou přesné jednotlivé metody. Zamysleme se tedy, jakým způsobem provádíme výpočet u jednotlivých metod. V případě dopředné a zpětné metody vždy bereme $n-1$ bodů z levé či pravé strany od aktuálně počítaného bodu (s výjimkou počátečních a koncových bodů, ve kterých musíme využít jiné asymetrické vzorce). Platí výše zmíněné principy – chybu lze zmenšit buď zmenšením kroku metody nebo zvýšením počtu vzorků pro výpočet (zvýšení n pro n -bodovou aproximaci). Na obrázcích 5.27 až 5.29 můžeme vidět jak se mění chyba výpočtu pomocí pětibodové aproximace, pokud snižujeme krok metody.

Pokud zvolíme pětibodovou aproximaci a krok $h = 0.1$ vidíme, že chyba výpočtu se pohybuje v řádech 10^{-3} . Při volbě kroku $h = 0.01$ je řád chyby 10^{-7} a při volbě $h = 0.001$ již pouze 10^{-10} . Zde ukazujeme grafy pro dopřednou metodu, pro zpětnou metodu jsou totiž grafy totožné.

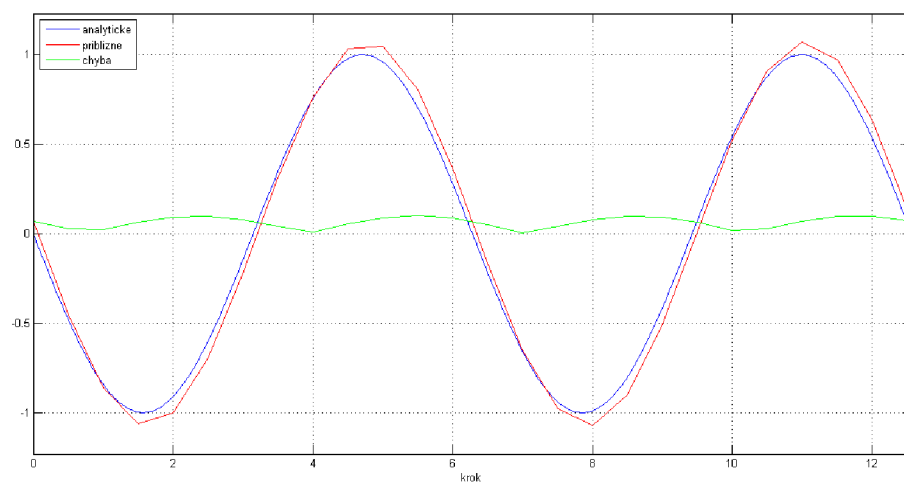
Obrázky 5.30 až 5.32 potom ukazují, že ke snížení chyby také dochází, pokud pro výpočet používáme vícebodové aproximace. Zvolíme krok $h = 0.01$ a budeme měnit pouze n -bodové aproximace. Pro tříbodovou aproximaci se chyba výpočtu pohybuje v řádech 10^{-3} , pro pětibodovou v řádech 10^{-7} . Dále pro sedmibodovou aproximaci se chyba výpočtu pohybuje v řádech 10^{-11} a pro devítibodovou dokonce 10^{-13} . Pro přehlednost uvádíme parametry použité pro výpočet dopředné a zpětné metody systémem Matlab (viz tabulky 5.2, 5.3). Pro připomenutí poznamenáme, že parametr l označuje počet bodů z levé strany od aktuálního bodu výpočtu a parametr k značí, kolik bodů pro výpočet používáme ze strany pravé. Pro dopřednou metodu se odkážeme na příklad 5.5, pro zpětnou na příklad 5.6.



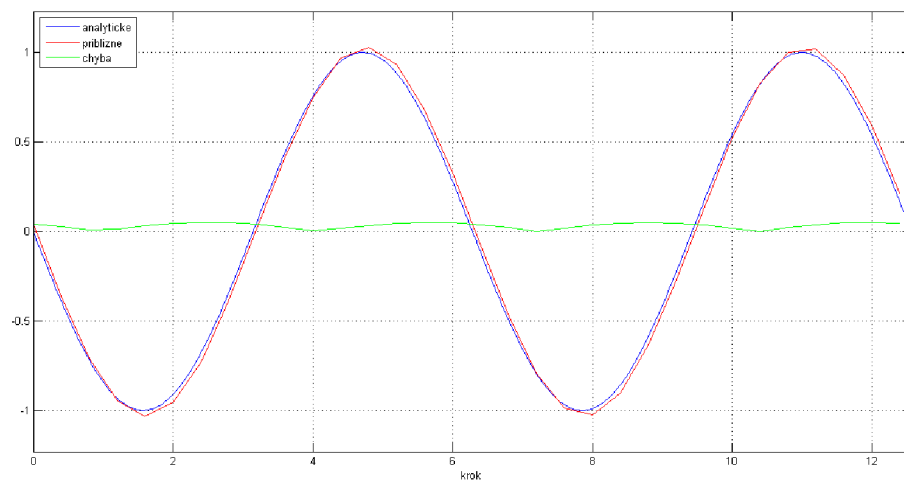
Obrázek 5.20: Pětibodová aproximace, krok $h = 0,7$



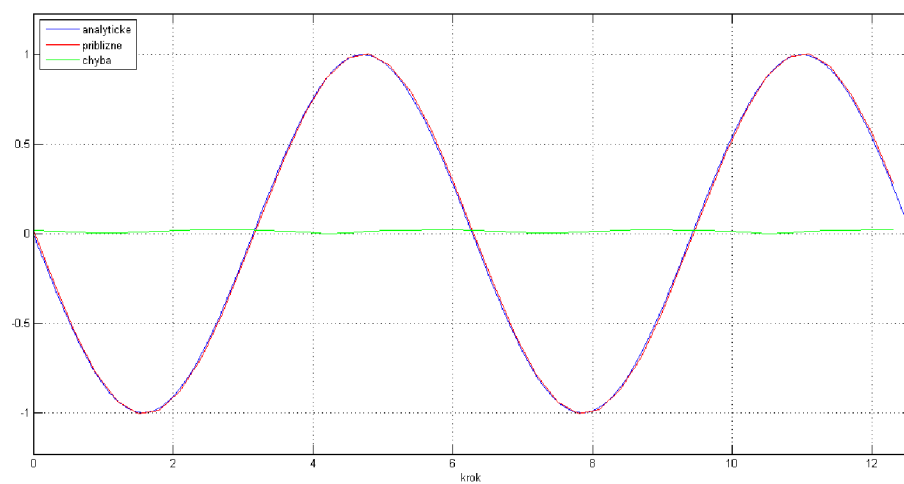
Obrázek 5.21: Pětibodová aproximace, krok $h = 0,6$



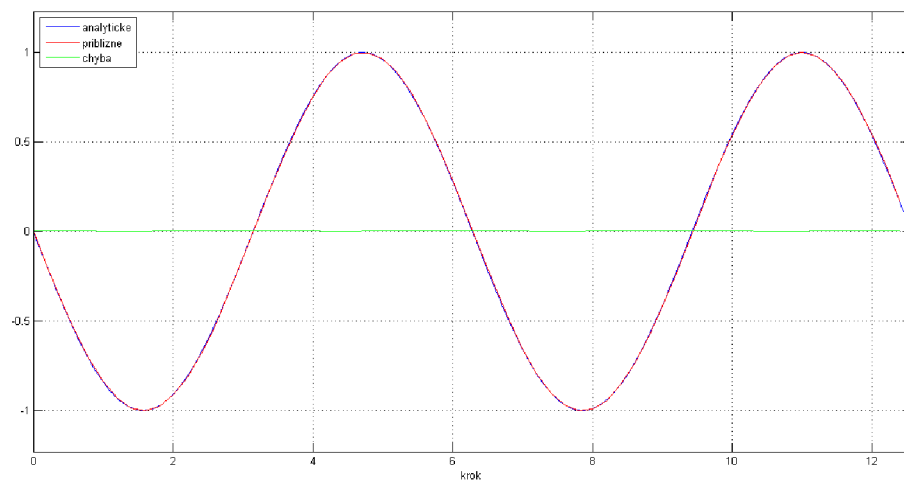
Obrázek 5.22: Pětibodová aproximace, krok $h = 0,5$



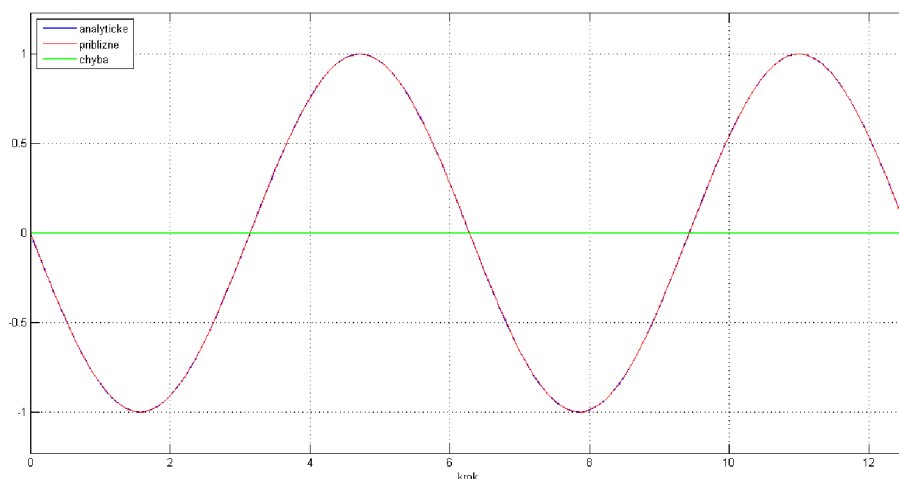
Obrázek 5.23: Pětibodová aproximace, krok $h = 0,4$



Obrázek 5.24: Pětibodová aproximace, krok $h = 0,3$



Obrázek 5.25: Pětibodová aproximace, krok $h = 0,2$



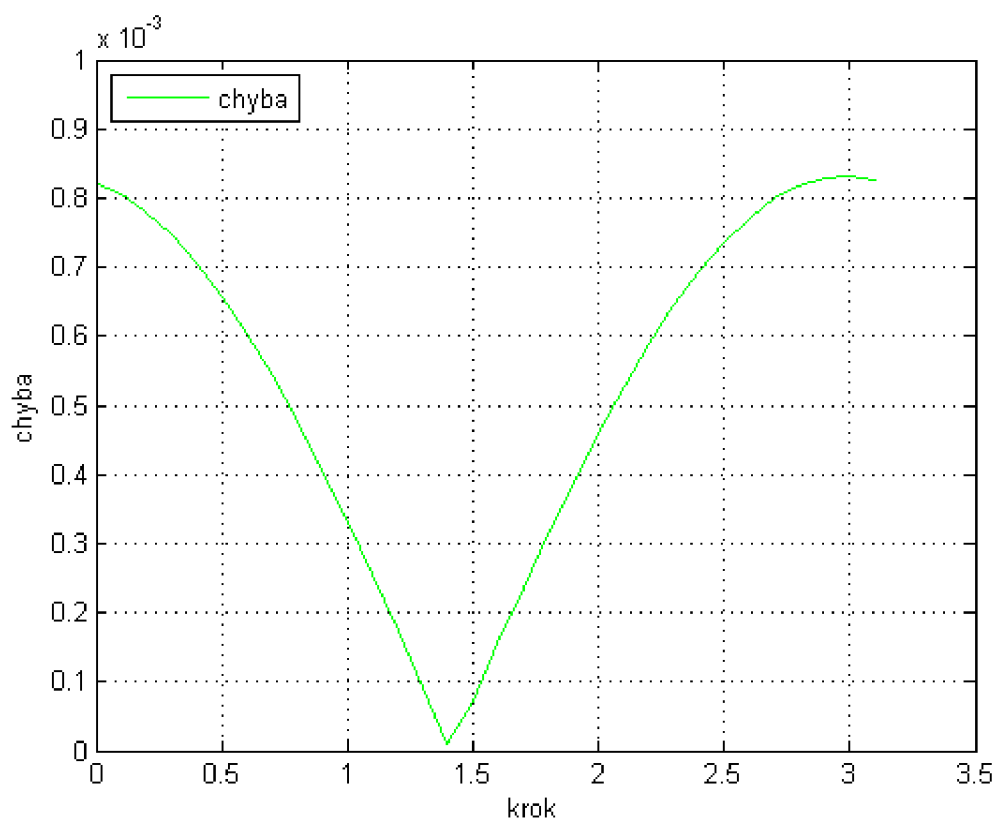
Obrázek 5.26: Pětibodová aproximace, krok $h = 0, 1$

Tabulka 5.2: Kombinace parametrů l a k pro dopřednou metodu

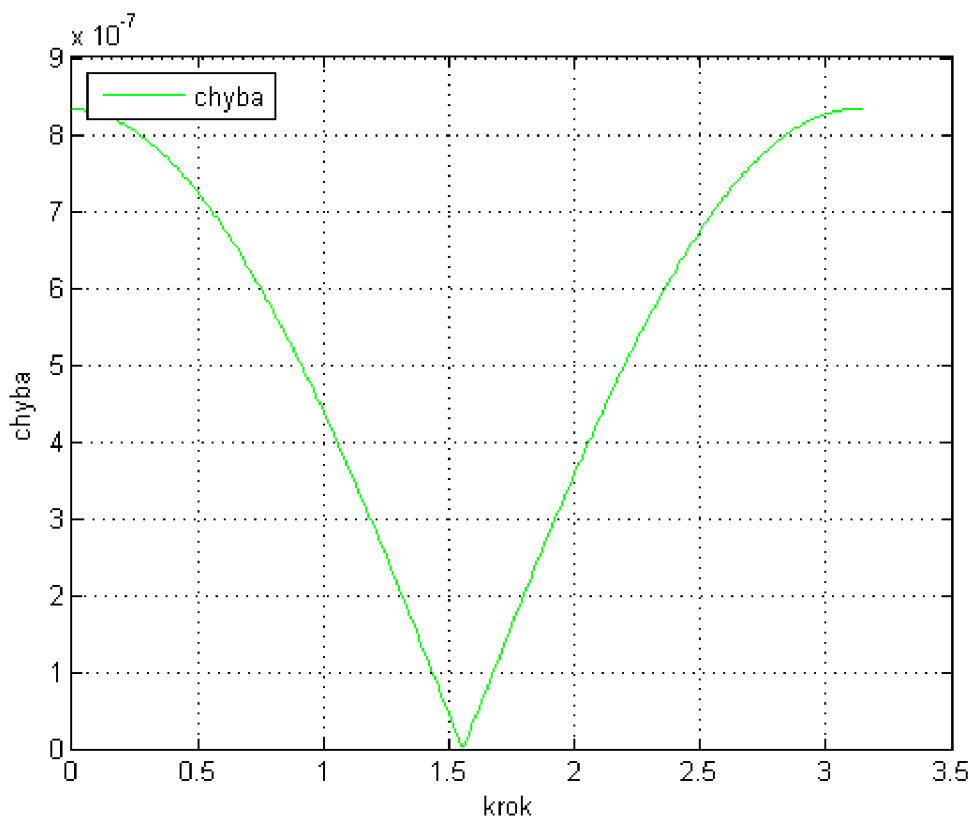
rovnice	parametr l	parametr k
v1'	0	4
v2'	0	4
v3'	0	4
v4'	0	4
v5'	0	4
v6'	0	4
v7'	0	4
v8'	0	4
v9'	1	3
v10'	2	2
v11'	3	1

U kombinované metody také samozřejmě platí výše uvedené principy. Rozdíl oproti dopředné a zpětné metodě je ve způsobu výpočtu. Jak jsme si řekli dříve, kombinovaná metoda využívá pro výpočet $\frac{n-1}{2}$ bodů z levé i pravé strany od aktuálně počítaného bodu (kromě počátečních a koncových bodů, kde je nutné opět využít asymetrické aproximace), n označuje n -bodovou aproximaci.

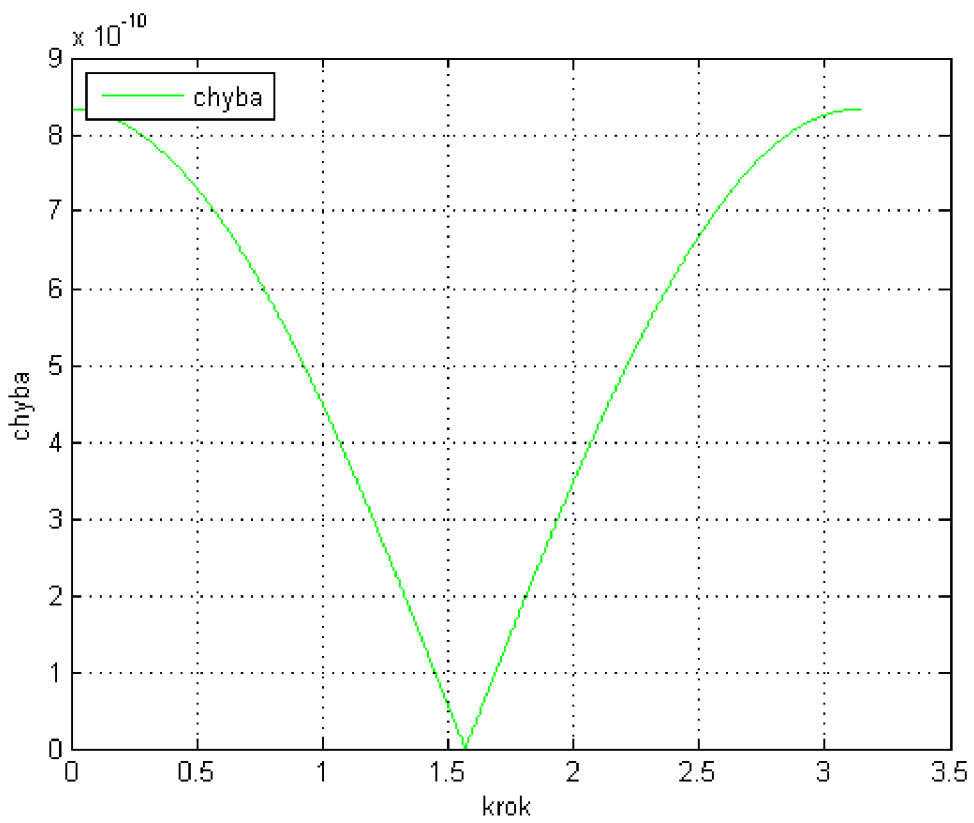
Díky tomu, že k výpočtu využíváme převážně symetrické vzorce, dochází ke výraznému zpřesnění výpočtu. Situaci názorně ukazují obrázky 5.33 až 5.38. Tyto výstupy můžeme porovnat s výstupy dopředné a zpětné metody. Řády chyby výpočtu se nezměnily, ale narozdíl od dopředné a zpětné metody má křivka znázorňující chybu kombinované metody vanovitý tvar. Vezměme opět pětibodovou aproximaci, z tabulky 5.4 jasně vidíme, že skutečně po většinu výpočtu využíváme dva body z levé strany a dva body z pravé strany. Díky symetrickým aproximacím nedochází k tak výrazné akumulaci chyby v každém kroku výpočtu a výpočet je mnohem přesnější. Pro ilustraci kombinované metody využijeme příklad 5.7.



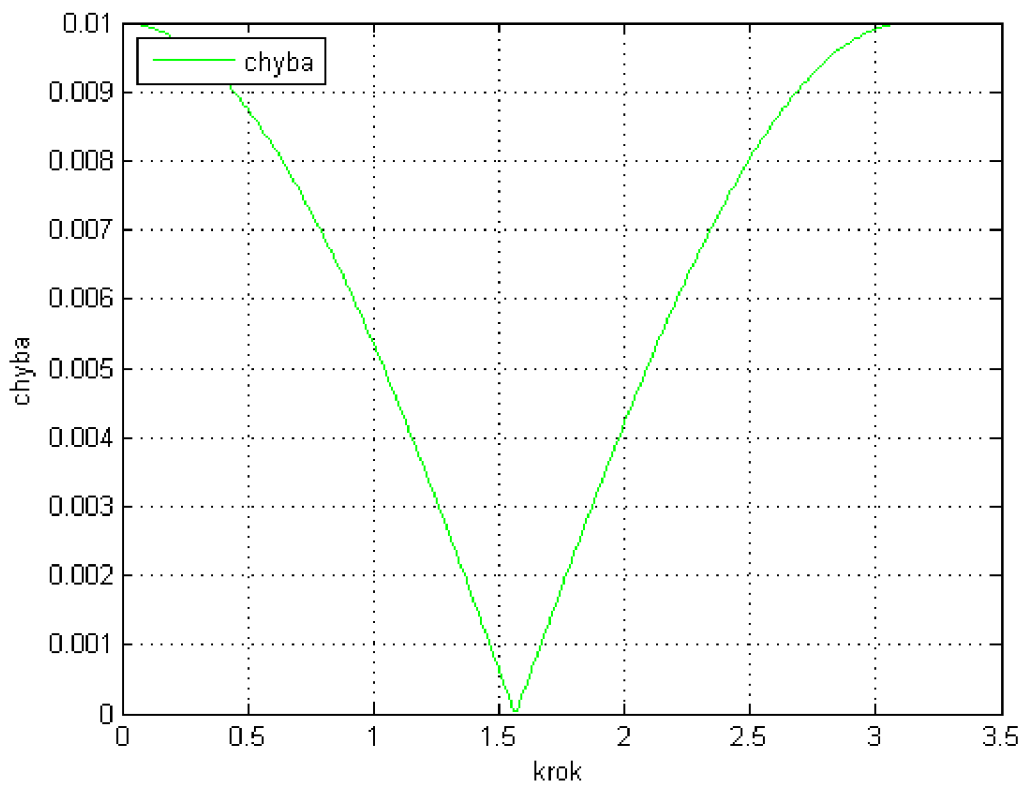
Obrázek 5.27: Chyba výpočtu Dopředné metody – $n = 5$, $h = 0.1$



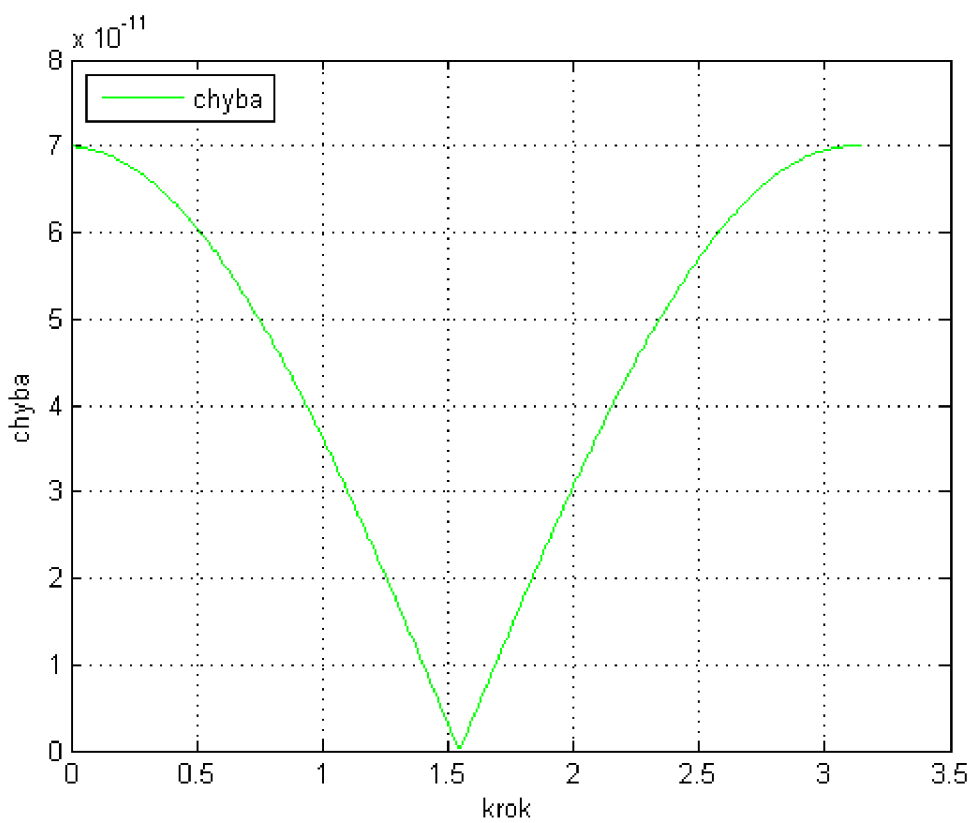
Obrázek 5.28: Chyba výpočtu Dopředné metody – $n = 5$, $h = 0.01$



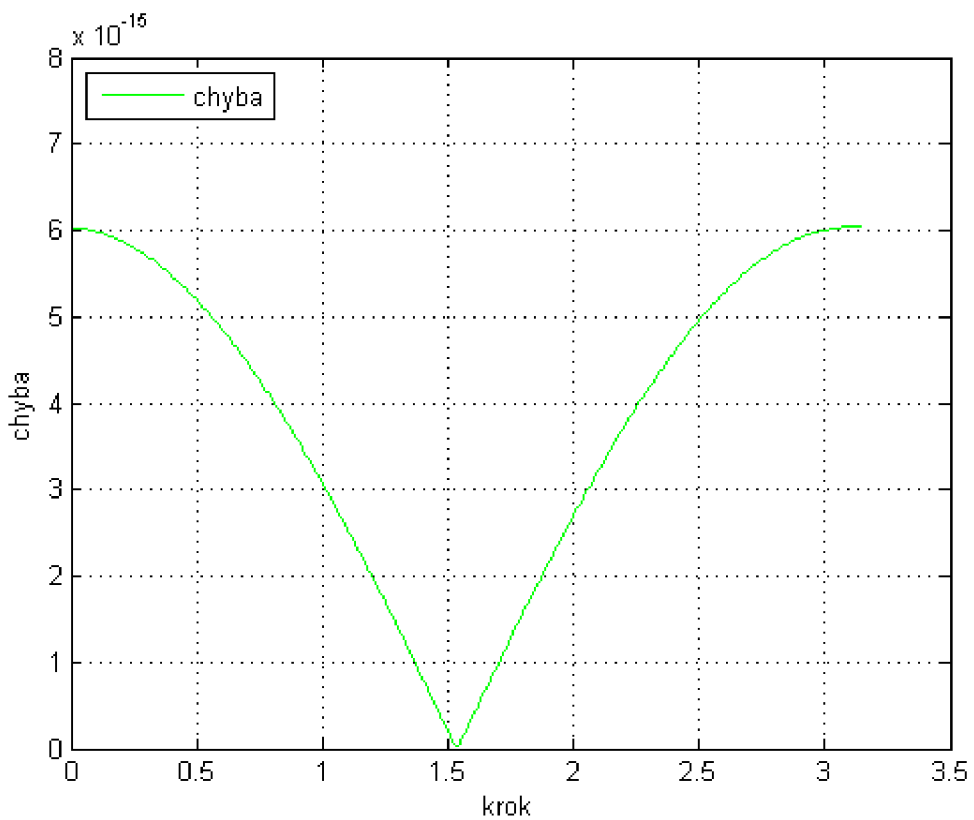
Obrázek 5.29: Chyba výpočtu Dopředné metody – $n = 5, h = 0.001$



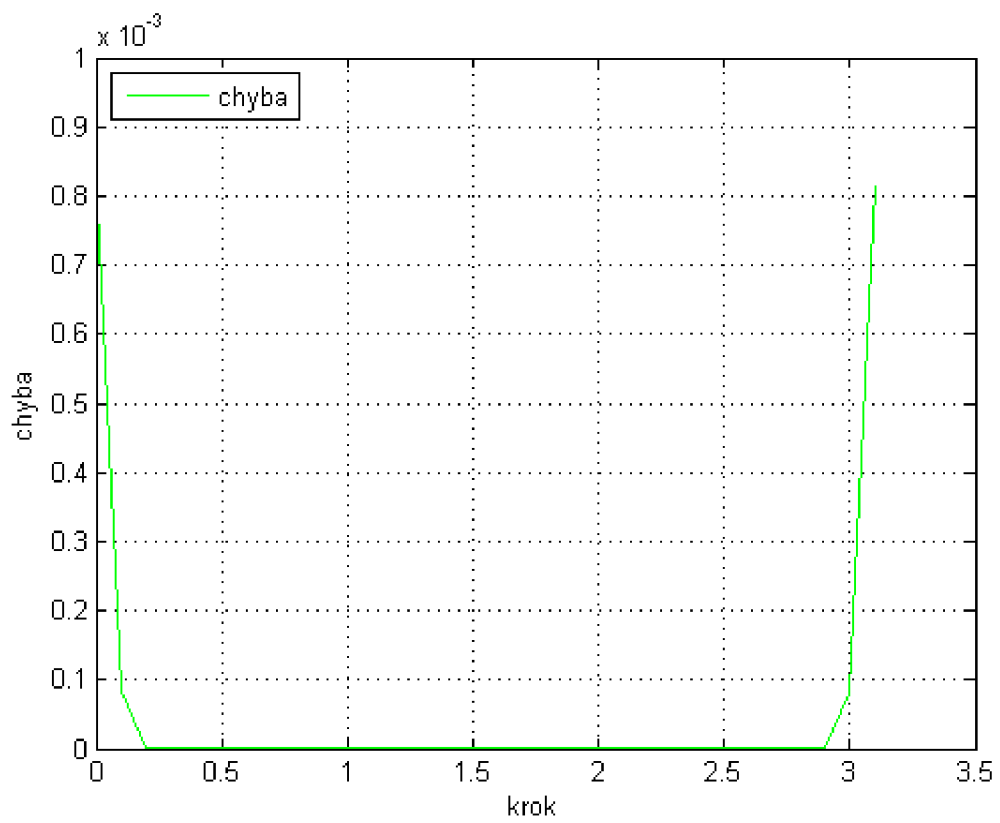
Obrázek 5.30: Chyba výpočtu Dopředné metody – $n = 3, h = 0.01$



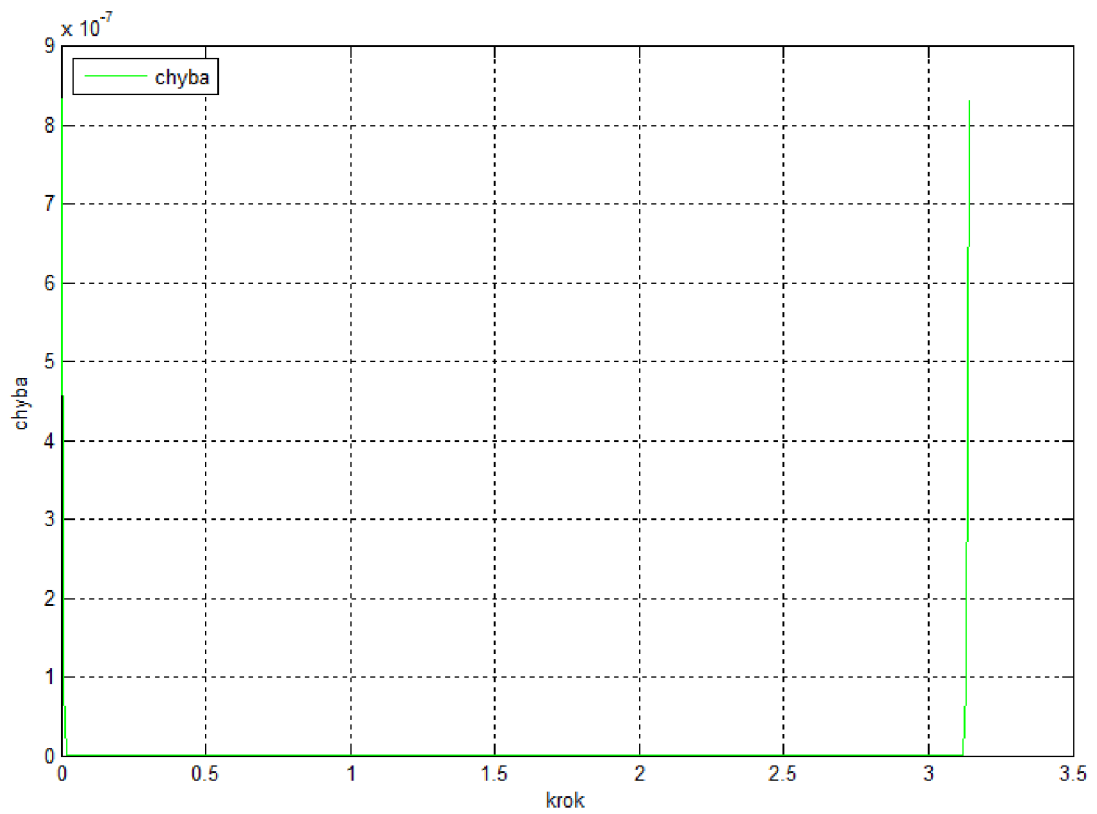
Obrázek 5.31: Chyba výpočtu Dopředné metody – $n = 7$, $h = 0.01$



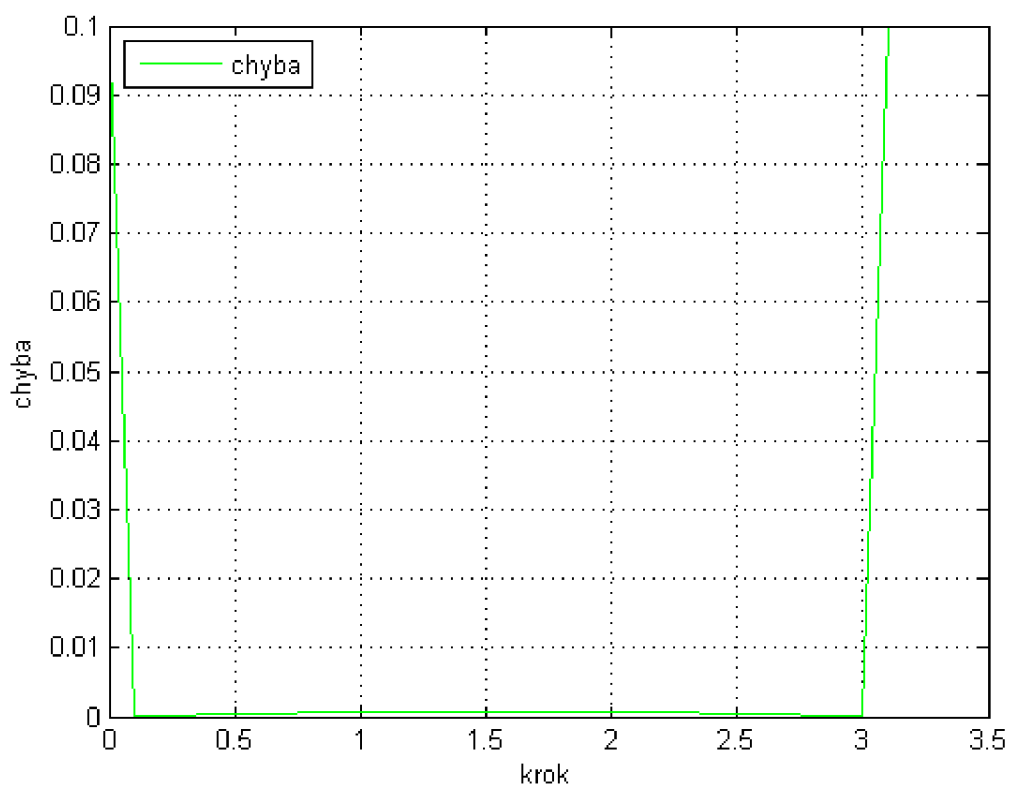
Obrázek 5.32: Chyba výpočtu Dopředné metody – $n = 9$, $h = 0.01$



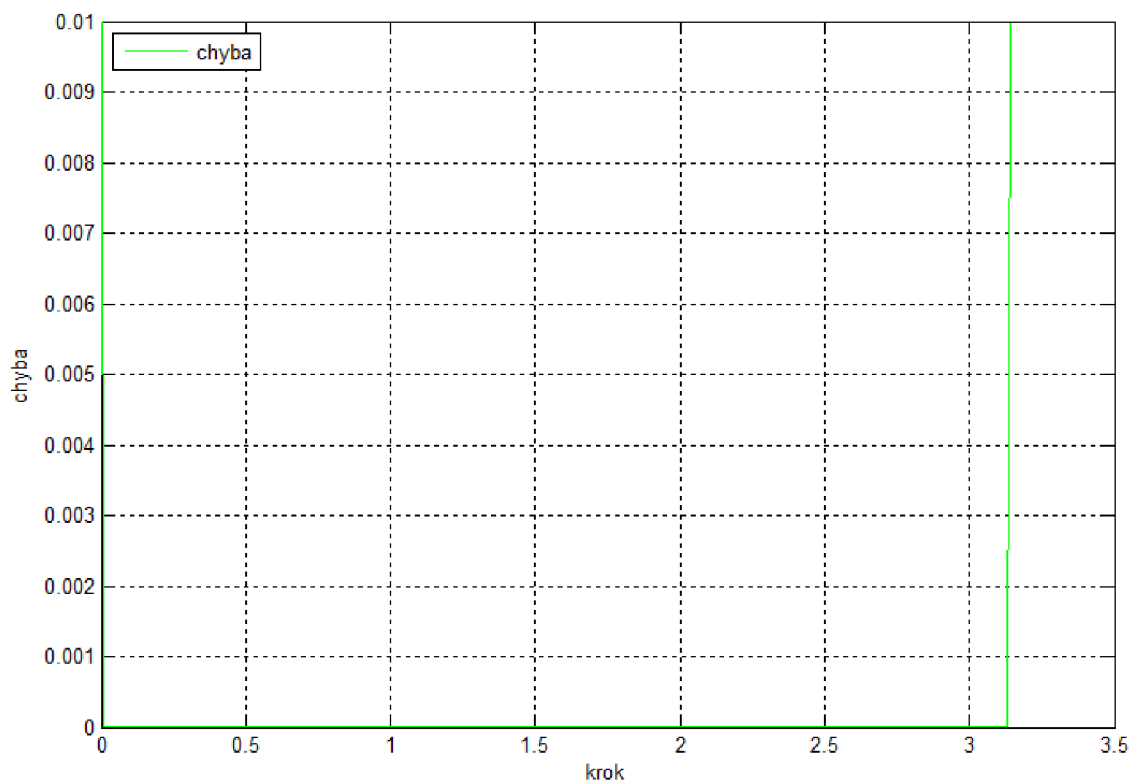
Obrázek 5.33: Chyba výpočtu Kombinované metody – $n = 5$, $h = 0.1$



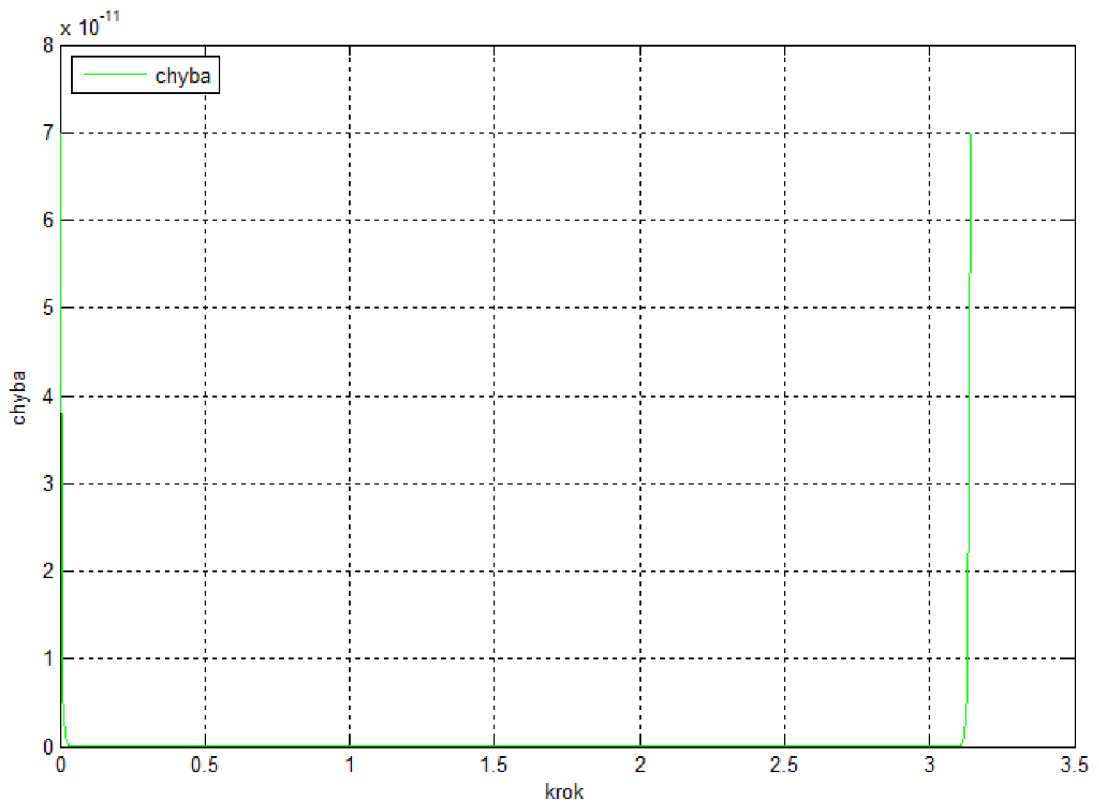
Obrázek 5.34: Chyba výpočtu Kombinované metody – $n = 5$, $h = 0.01$



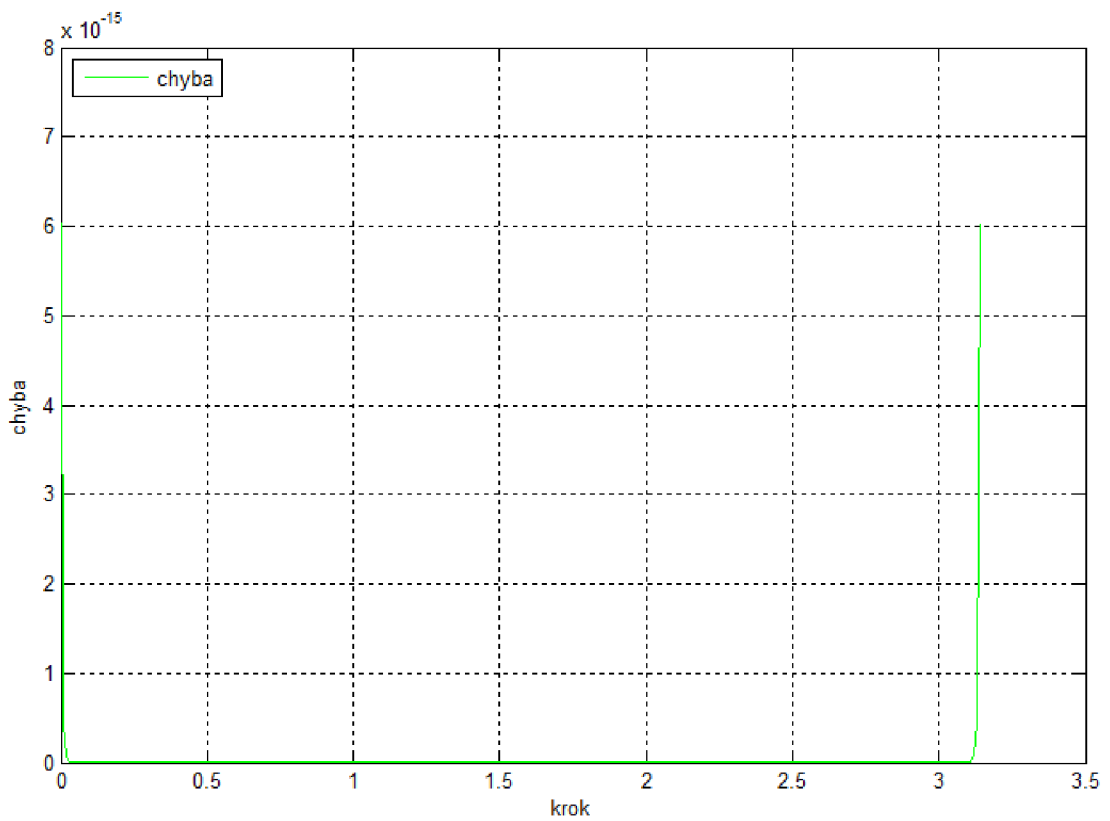
Obrázek 5.35: Chyba výpočtu Kombinované metody – $n = 3$, $h = 0.1$



Obrázek 5.36: Chyba výpočtu Kombinované metody – $n = 3$, $h = 0.01$



Obrázek 5.37: Chyba výpočtu Kombinované metody – $n = 7, h = 0.01$



Obrázek 5.38: Chyba výpočtu Kombinované metody – $n = 9, h = 0.01$

Tabulka 5.3: Kombinace parametrů 1 a k pro zpětnou metodu

rovnice	parametr 1	parametr k
v11'	4	0
v10'	4	0
v9'	4	0
v8'	4	0
v7'	4	0
v6'	4	0
v5'	4	0
v4'	4	0
v3'	3	1
v2'	2	2
v1'	1	3

Tabulka 5.4: Kombinace parametrů 1 a k pro kombinovanou metodu

rovnice	parametr 1	parametr k
v1'	1	3
v2'	2	2
v3'	2	2
v4'	2	2
v5'	2	2
v6'	2	2
v7'	2	2
v8'	2	2
v9'	2	2
v10'	2	2
v11'	3	1

5.7 Časová náročnost výpočtů

V této podkapitole se podíváme na srovnání časové náročnosti řešení PDR v systémech TKSL/C a Matlab. Ověření výpočtů PDR v systému Matlab probíhalo za pomoci aplikační knihovny *Partial Differential Equation Toolbox*¹, která obsahuje nástroje pro řešení parciálních diferenciálních rovnic ve dvou prostorových rozměrech a čase. Pro řešení hyperbolické PDR lze využít funkci `hyperbolic`². Tato funkce pro řešení využívá metodu konečných prvků (FEM – Finite Element Method), řešení vede na převod na rozsáhlou soustavu lineárních rovnic, poté využití LU dekompozic a velice rychlý výpočet matic v Matlabu. Výpočty prováděné systémem Matlab jsou tedy vysoce optimalizované a rychlé. Naproti tomu výpočty hyperbolických PDR pomocí dopředné, zpětné a kombinované metody v systému TKSL/C takových rychlostí nedosahují a bylo by zapotřebí provést optimalizace těchto výpočtů. Je nutné ovšem podotknout, že řešení PDR v systému TKSL/C se provádělo pomocí jejich převodu na ODR (obyčejné diferenciální rovnice) prvního řádu, jde tedy o naprosto odlišný způsob výpočtu, než jaký je v systému Matlab. Metoda je ale vhodná pro paralelizaci výpočtů (viz kapitola 6), dále lze také uvedenou metodu snadno modifikovat pro

¹<http://www.mathworks.com/products/pde/>.

²<http://www.mathworks.com/help/pde/ug/hyperbolic.html>.

výpočet parabolických a eliptických PDR. Dalším důležitým aspektem je také to, že pomocí této metody lze vypočítat i derivace vyšších řádů, než druhého. Pokud bychom potřebovali vyčíslit derivaci n -tého řádu, využijeme pro tento účel n -tý řádek inverzní matice. Pokud pro výpočet využijeme n -bodovou aproximaci, potom lze získat až $(n - 1)$ derivaci v každém bodě výpočtu. Pokud bychom například výpočet prováděli pomocí desetibodové aproximace, potom bychom byli schopni v každém bodě získat derivaci až devátého řádu. Na závěr dodáme, že námi zvolená metoda je vhodná pro testovací sadu úloh z předmětu VNV (Vysoce náročné výpočty).

Kapitola 6

Paralelní řešení PDR

V této kapitole se seznámíme se způsobem paralelního řešení parciálních diferenciálních rovnic. Ukážeme si, jak vytvořit programové schéma pro hyperbolickou parciální diferenciální rovnici v závislosti na počtu řezů na struně, n -bodové aproximaci a také použité metodě výpočtu. Podrobnější informace jsou uvedeny v [5]. Vše si vysvětlíme na následujícím příkladu 6.1. Pro řešení parciálních diferenciálních rovnic budeme využívat síť integrátorů. Integrátory budou nezávisle na sobě provádět výpočty nad vstupními daty. Díky tomuto principu lze PDR řešit paralelně.

Příklad 6.1. *Mějme tříbodovou aproximaci a celkem deset řezů na struně. Na základě těchto informací máme sestavit rovnice pro kombinovanou metodu řešení PDR pro systém TKSL.*

```
var
  u0, u1, u2, u3, u4, u5, u6, u7, u8, u9, u10,
  v1, v2, v3, v4, v5, v6, v7, v8, v9;
Const
  PR = 10,
  A~ = PR*PR/10,
  tmax = 10,
  dt = 0.01,
  eps = 1e-20,
  PI = 3.141592653589793238462643383279502884197
  169399375105820974944592307816406286208998628034825342117068;
system
  u0 = 0;
  u10 = 0;
  v1' = A~* 0.5*(u1-u0) + 0.5*(u2-u1) &sin(PI*1/PR);
  v2' = A~* 0.5*(u2-u1) + 0.5*(u3-u2) &sin(PI*2/PR);
  v3' = A~* 0.5*(u3-u2) + 0.5*(u4-u3) &sin(PI*3/PR);
  v4' = A~* 0.5*(u4-u3) + 0.5*(u5-u4) &sin(PI*4/PR);
  v5' = A~* 0.5*(u5-u4) + 0.5*(u6-u5) &sin(PI*5/PR);
  v6' = A~* 0.5*(u6-u5) + 0.5*(u7-u6) &sin(PI*6/PR);
  v7' = A~* 0.5*(u7-u6) + 0.5*(u8-u7) &sin(PI*7/PR);
  v8' = A~* 0.5*(u8-u7) + 0.5*(u9-u8) &sin(PI*8/PR);
  v9' = A~* 0.5*(u9-u8) + 0.5*(u10-u9) &sin(PI*9/PR);
  u1' = v1 &0;
```

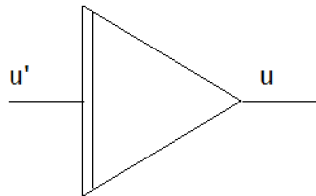
```

u1' = v1 &0;
u2' = v2 &0;
u3' = v3 &0;
u4' = v4 &0;
u5' = v5 &0;
u6' = v6 &0;
u7' = v7 &0;
u8' = v8 &0;
u9' = v9 &0;
sysend.

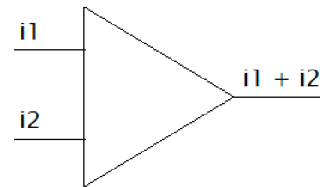
```

□

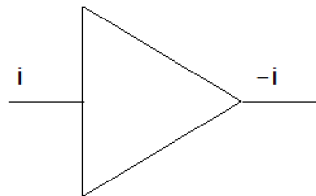
Na základě výše uvedeného zdrojového kódu pro systém TKSL můžeme zjistit, jaké komponenty budeme potřebovat pro sestavení programového schématu. Nejprve si ujasníme, jaké komponenty budou potřeba pro každou samostatnou rovnici. Nejdříve uvádíme obrázky jednotlivých prvků, které budeme ve schématu využívat. Jedná se o prvky integrátor (obrázek 6.1), sumátor (obrázek 6.2), invertor (obrázek 6.3) a konstanta (obrázek 6.4). Na jednotlivých obrázcích zároveň vidíme, jaké operace daný prvek provádí.



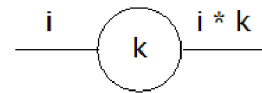
Obrázek 6.1: Integrátor



Obrázek 6.2: Sumátor



Obrázek 6.3: Invertor

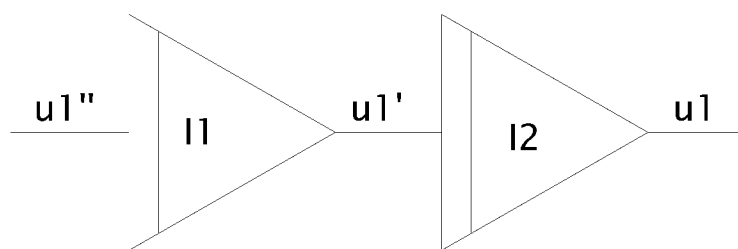


Obrázek 6.4: Konstanta

Protože řešíme PDR druhého řádu, určitě budeme muset pro výpočet využít dva numerické integrátory. Při sestavování programového schématu platí, že levá strana rovnice tvoří výstup a pravá strana vstup. Vezměme rovnici pro $v1'$ z výše uvedeného příkladu.

$$v1' = A \sim * 0.5*(u1-u0) + 0.5*(u2-u1) \&sin(PI*1/PR);$$

Na levé straně rovnice vidíme derivovanou proměnnou $v1'$. Protože řešíme hyperbolicke PDR, vyskytují se zde substituce, tedy $u1'=v1$. Ve výsledku dostáváme, že $v1'=u1''$. Skutečně jsme tedy obdrželi diferenciální rovnici druhého řádu. Sériové zapojení dvou integrátorů je znázorněno na obrázku 6.5. Vstupem integrátoru I1 je tedy $u1''$, výstupem je potom integrovaná hodnota vstupu, tedy $u1'$, která zároveň tvoří vstup druhého integrátoru I2. Výstup druhého integrátoru je potom samotné $u1$. Nyní se podíváme na pravou stranu rovnice, která bude tvořit vstup do série integrátorů. Můžeme zde vidět operace sčítání, odčítání a násobení. Zapojení sestavíme takovým způsobem, že jednotlivé výrazy uvedené v závorkách přivedeme na vstup sumátorů.



Obrázek 6.5: Sériové zapojení integrátorů

V našem případě budou v programovém schématu celkem dva sumátory ($S1$, $S2$), každý z nich bude opatřen dvěma vstupy, protože potřebuje odčítat dvě hodnoty. Toho docílíme tak, že na vstup daného sumátoru přivedeme zápornou hodnotu, kterou získáme pomocí invertoru (označeného jako INV). Prvek invertor se ve schématu nachází za posledním integrátorem a obrací polaritu. To znamená, že pokud například na jeho vstup přivedeme hodnotu $u1$, na výstupu bude hodnota opačná, tedy $-u1$.

Dále vidíme, že každý výraz v závorce je násoben jistou konstantou (konstanty $k1$, $k2$) a následně jsou tyto vynásobené výrazy přivedeny na vstup sumátoru $S3$. Na závěr dodáme, že výraz $\&sin(PI*1/PR)$ označuje počáteční podmínky výpočtu, které budou nastaveny na integrátorech. Výsledné zapojení pro rovnici $v1'$ můžeme vidět na obrázku 6.6. Pro jednoduchost byla uvedena zapojení pro rovnice $v1'$, $v2'$, $v3'$ a $v9'$. Nyní víme, jakým způsobem vypadá zapojení pro jednu rovnici. Pro ostatní rovnice $v2'$ až $v9'$ postupujeme obdobně. Výsledné schéma znázorňuje obrázek 6.7. Všimněme si, že pro hodnoty $u0$ a $u10$ nesestavujeme zapojení integrátorů, protože se jedná o body ukotvení struny, jejichž hodnota je nulová. Podíváme se, jak bude zapojení vypadat, pokud budeme mít opět deset řezů, ale pětibodovou aproximaci. Navíc budeme uvažovat dopřednou metodu řešení PDR. Pro ilustraci uvádíme příklad 6.2.

Příklad 6.2. *Mějme pětibodovou aproximaci a celkem deset řezů na struně. Naším úkolem je sestavit rovnice pro dopřednou metodu a poté odpovídající programové schéma.*

```

var
  u0, u1, u2, u3, u4, u5, u6, u7, u8, u9, u10,
  v1, v2, v3, v4, v5, v6, v7, v8, v9;
Const
  PR = 10,
  A~ = PR*PR/10,
  tmax = 10,
  dt = 0.01,
  eps = 1e-20,
  PI = 3.141592653589793238462643383279502884197
  169399375105820974944592307816406286208998628034825342117068;
system
  u0 = 0;
  u10 = 0;
  v1' = A~* -4.333*(u2-u1) + 4.75*(u3-u1) - 2.333*(u4-u1) + 0.458*(u5-u1)
  &sin(PI*1/PR);

```

```

v2' = A~* -4.333*(u3-u2) + 4.75*(u4-u2) - 2.333*(u5-u2) + 0.458*(u6-u2)
      &sin(PI*2/PR);
v3' = A~* -4.333*(u4-u3) + 4.75*(u5-u3) - 2.333*(u6-u3) + 0.458*(u7-u3)
      &sin(PI*3/PR);
v4' = A~* -4.333*(u5-u4) + 4.75*(u6-u4) - 2.333*(u7-u4) + 0.458*(u8-u4)
      &sin(PI*4/PR);
v5' = A~* -4.333*(u6-u5) + 4.75*(u7-u5) - 2.333*(u8-u5) + 0.458*(u9-u5)
      &sin(PI*5/PR);
v6' = A~* -4.333*(u7-u6) + 4.75*(u8-u6) - 2.333*(u9-u6) + 0.458*(u10-u6)
      &sin(PI*6/PR);
v7' = A~* 0.458*(u9-u8) + 0.25*(u10-u9) + 0.166*(u11-u9) - 0.042*(u12-u9)
      &sin(PI*9/PR);
v8' = A~* -0.042*(u10-u9) - 0.666*(u10-u8) + 0.666*(u11-u10) - 0.042*(u12-u10)
      &sin(PI*10/PR);
v9' = A~* 0.458*(u11-u10) - 2.333*(u11-u9) + 4.75*(u11-u8) - 4.333*(u12-u11)
      &sin(PI*11/PR);
u1' = v1 &0;
u1' = v1 &0;
u2' = v2 &0;
u3' = v3 &0;
u4' = v4 &0;
u5' = v5 &0;
u6' = v6 &0;
u7' = v7 &0;
u8' = v8 &0;
u9' = v9 &0;
sysend.

```

□

Při vytváření programového schématu postupujeme obdobným způsobem jako v předchozím příkladu. Opět nejdříve uvedeme zapojení pro první rovnici $v1'$ (viz obrázek 6.8). Levá strana rovnice bude i v tomto případě reprezentována dvěma integrátory zapojenými v sérii. Rozdíl nastává při zapojování vstupu, který je dán pravou stranou rovnice. Oproti předchozímu příkladu zde můžeme vidět celkem čtyři výrazy v závorkách. To pro nás znamená, že budeme potřebovat čtyři sumátory $S1$, $S2$, $S3$, $S4$, které provedou sečtení výrazů v závorkách. Jednotlivé výrazy jsou násobeny konstantami $k1$, $k2$, $k3$ a $k4$. Takto vynásobené výrazy jsou přivedeny do sumátoru $S5$, jehož výstup pak tvoří vstup do série integrátorů. Výsledné schéma je ukázáno na obrázku 6.9, můžeme zde vidět zapojení pro rovnice $v1'$, $v2'$ a $v9'$. Nyní můžeme odvodit, jak bude vypadat programové schéma pro n -bodovou aproximaci a libovolný počet řezů na struně. Pro tříbodovou aproximaci a deset řezů na struně jsme pro jednu rovnici využili tento počet členů:

- 2 integrátory,
- 1 invertor,
- 3 sumátory, z toho 2 pro součet prvků v závorkách a 1 pro součet samotných výrazů v závorkách,
- 2 konstanty pro vynásobení výrazů v závorkách.

Pro pětibodovou aproximaci a deset řezů na struně je situace následující:

- 2 integrátory,
- 1 invertor,
- 5 sumátorů, z toho 4 pro součet prvků v závorkách a 1 pro součet samotných výrazů v závorkách,
- 4 konstanty pro vynásobení výrazů v závorkách.

Označme nyní n -bodovou aproximaci písmenem n a počet řezů na struně písmenem r . Pro počty prvků v programovém schématu pak platí vztahy uvedené v tabulce 6.1. Položka tabulky *Počet dílčích sumátorů* vyjadřuje počet sumátorů, které provádí součet prvků v jednotlivých závorkách, položka *Počet sumátorů* značí počet souhrnných sumátorů, které sčítají již jednotlivé závorky. Jak již bylo naznačeno na začátku kapitoly, klíčovým parametrem pro paralelní zpracování je právě počet integrátorů. Každou z rovnic jsme schopni popsat pomocí dvou integrátorů a určitého počtu dalších prvků. Tento celek pak tvoří samostatnou operační jednotku. Zrychlení tedy bude dáno počtem integrátorů, které jsou ve schématu využity. Pro zrychlení výpočtu platí tedy následující vztah.

$$zrychlení = 2 \cdot (pocetRezu - 1) \quad (6.1)$$

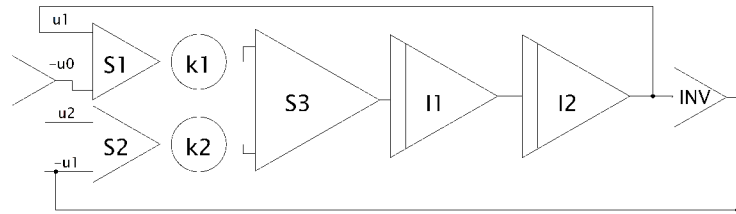
Pro přehlednost uvádíme tabulku 6.2, ve které můžeme vidět vypočtená zrychlení pro různý počet řezů na struně.

Tabulka 6.1: Počet jednotlivých prvků v programovém schématu v závislosti na n-bodové aproximaci a počtu řezů na struně

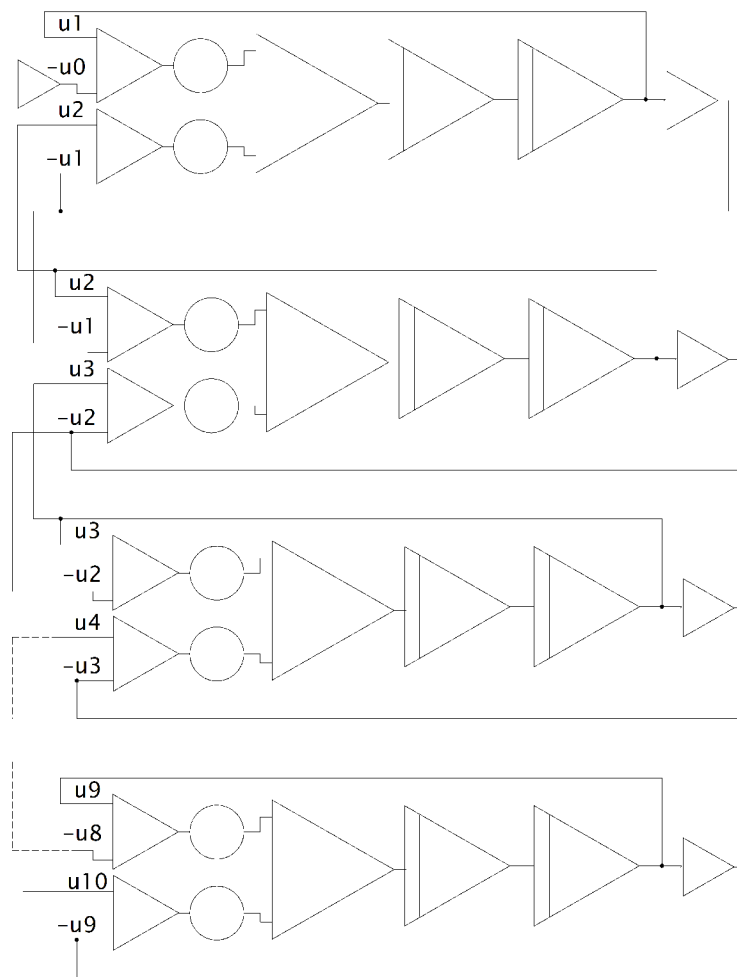
Prvek	Výsledný počet
Počet integrátorů	$2 \cdot (r - 1)$
Počet dílčích sumátorů	$(n - 1) \cdot (r - 1)$
Počet sumátorů	$r - 1$
Počet invertorů	$r - 1$
Počet konstant	$(n - 1) \cdot (r - 1)$

Tabulka 6.2: Počet jednotlivých prvků v programovém schématu v závislosti na n-bodové aproximaci a počtu řezů na struně

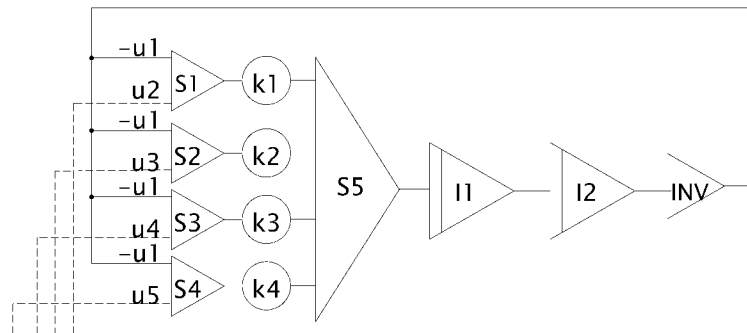
Počet řezů	Výsledné zrychlení
10	18
12	22
14	26
16	30
20	38
30	58
40	78
50	98
100	198
1000	1998
10000	19998
100000	199998



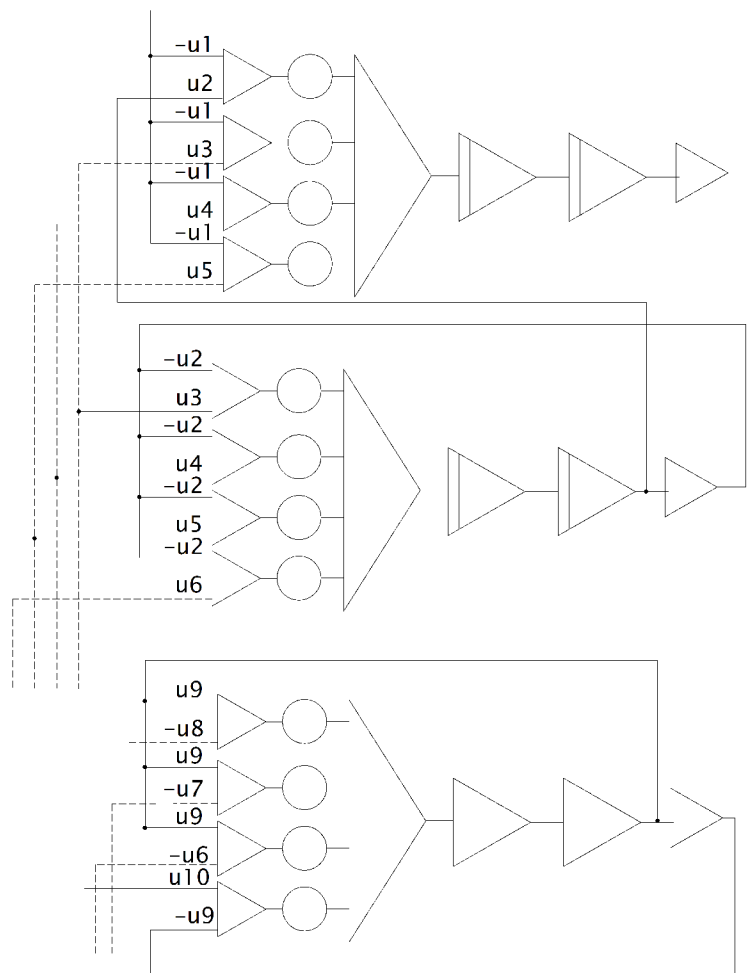
Obrázek 6.6: Programové schéma pro příklad 6.1 — rovnice $v1'$



Obrázek 6.7: Celkové programové schéma pro příklad 6.1



Obrázek 6.8: Programové schéma pro příklad 6.2 — rovnice v_1'



Obrázek 6.9: Celkové programové schéma pro příklad 6.2

Kapitola 7

Implementace

Tato kapitola se zabývá implementací aplikace *PDE Generator*, která slouží pro generování soustav ODR pro systém TKSL. Tyto soustavy ODR reprezentují hyperbolickou PDR. Aplikace byla naprogramována v jazyce Java. Pro implementaci uživatelského rozhraní bylo využito architektury MVC (Model View Controller). Pro správnou funkčnost aplikace je zapotřebí mít nainstalované vývojové prostředí Eclipse¹ a také JDK (Java Development Kit)².

Model je v aplikaci reprezentován programy `nthderivative`, `getSin` a `TKSLGen.py`. Programy `nthderivative` a `getSin` byly vytvořeny v jazyce C. Pro překlad těchto programů je nutné mít ve vývojovém prostředí Eclipse nainstalovaný nástroj CDT (C/C++ Development Tooling). Nástroj CDT může pro samostatný překlad využívat různé kompilátory, v tomto případě byl zvolen kompilátor MinGW. Je nutné, aby kompilátor zahrnoval knihovny GMP i MPFR (viz níže). Nyní se dostáváme k jednotlivým programům. Program `nthderivative` slouží k vygenerování koeficientů inverzní matice. Dále program `getSin` se využívá pro vygenerování počátečních podmínek, jejich výpočet probíhá dle vztahu $PP(i) = \sin(\pi \cdot \frac{i}{cuts})$, kde *cuts* znamená počet řezů na struně. Je důležité uvést, že oba programy využívají pro svoje výpočty funkce z knihoven GMP³ a MPFR⁴. Knihovna GMP (GNU Multi Precision Arithmetic Library) umožňuje provádět výpočty s libovolnou přesností pro celá čísla, racionální čísla, reálná čísla s pohyblivou řádovou čárkou. Knihovna je původně napsána pro jazyk C, ale existují rozhraní i pro další jazyky (C/C++, C#, Perl, PHP, Python, Java). Další využívanou knihovnou je knihovna MPFR (GNU Multiple Precision Floating-Point Reliably), která je založena na již zmíněné knihovně GMP. Na rozdíl od knihovny GMP je v knihovně MPFR k dispozici mnohem více funkcí jako například transcendentní funkce (exponenciální funkce, logaritmy, sinus, cosinus, tangens, cotangens, atd.). Právě funkce sinus byla využita v programu `getSin` pro vygenerování počátečních podmínek. Poslední součástí modelu je program `TKSLGen.py`, který byl vytvořen v jazyce Python a slouží pro vygenerování závorkových výrazů pro jednotlivé metody řešení hyperbolické PDR. Abychom mohli volat z Java aplikace funkce z tohoto skriptu, bylo nezbytné si opatřit knihovnu Jython⁵.

Vrstvu pohledu (view) zaštiťuje třída `TKSLGenView.java`, v této třídě jsou umístěny jednotlivé prvky uživatelského rozhraní. Vrstva kontroléru je reprezentována třídou

¹<http://www.eclipse.org>.

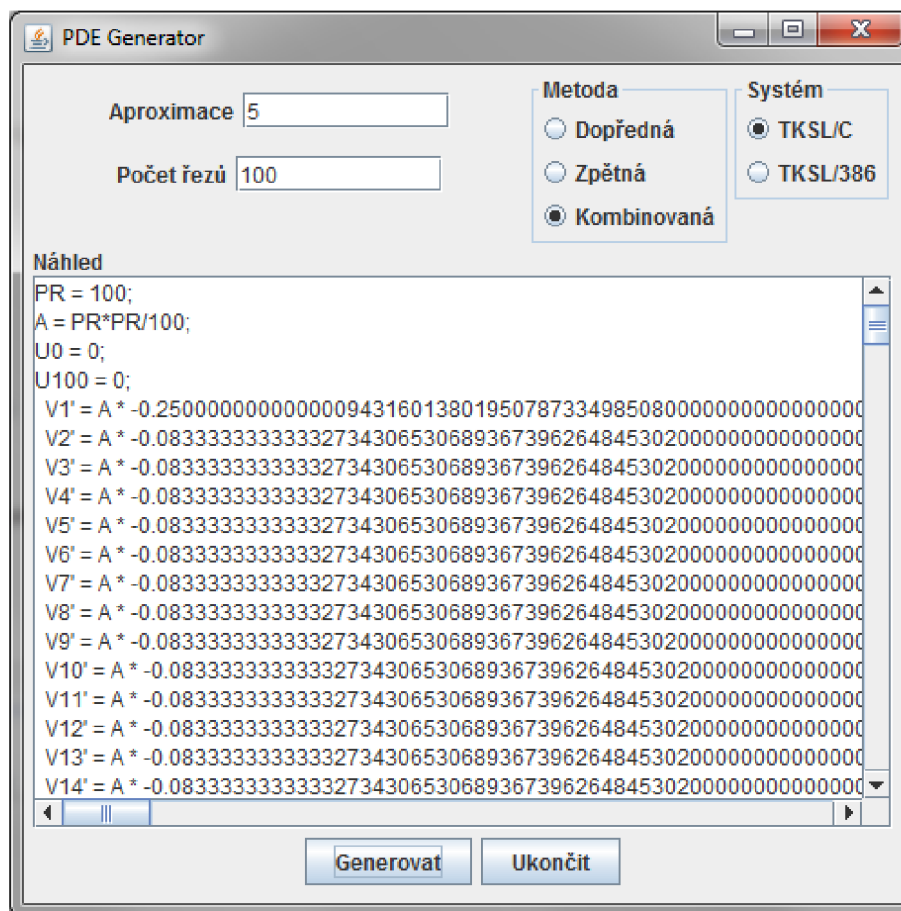
²<http://www.oracle.com/technetwork/java/javase/downloads/index.html>.

³<https://gmplib.org>

⁴<http://www.mpfr.org>

⁵<http://www.jython.org>.

TKSLGenController.java. Grafické uživatelské rozhraní je ukázáno na obrázku 7.1.

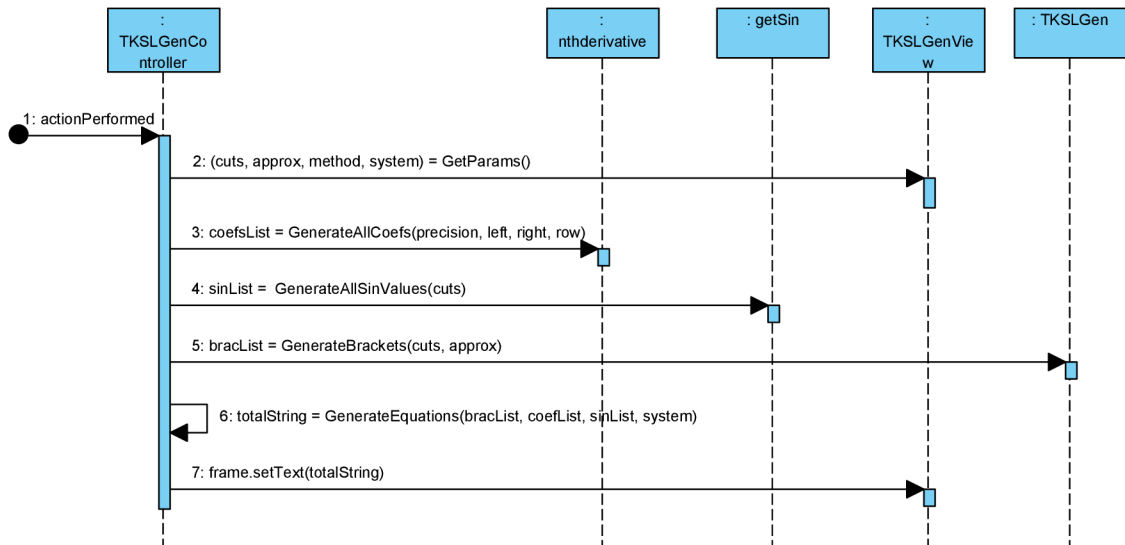


Obrázek 7.1: Grafické uživatelské rozhraní

Činnost aplikace znázorňuje sekvenční diagram 7.2. Při stisku tlačítka **Generovat** se nejdříve zjistí parametry výpočtu:

- jaký je počet řezů na struně,
- kolikabodová aproximace bude využita,
- jaká metoda výpočtu byla zvolena (dopředná, zpětná, kombinovaná),
- pro jaký systém se budou rovnice generovat (TKSL/C, TKSL/386).

Tyto informace získáme z třídy `TKSLGenView.java`, která reprezentuje pohled. Poté probíhá komunikace s modelem. Získají se koeficienty inverzní matice pomocí volání programu `nthderivative`, přičemž parametry jsou požadovaná přesnost koeficientů (`precision`), počet vzorků z levé strany (`left`), počet vzorků z pravé strany (`right`) a řádek inverzní matice, který si přejeme získat (`row`), v našem případě vždy získáváme druhý řádek inverzní matice, protože počítáme druhé derivace. Poté je volán program `getSin` pro získání hodnot počátečních podmínek, parametrem je počet řezů na struně (`cuts`). Výsledky vrácené programy `nthderivative` a `getSin` jsou ukládány do dočasných souborů. Pro komunikaci s programy napsanými v jazyce C byla využita třída `ProcessBuilder`.



Obrázek 7.2: Sekvenční diagram činnosti aplikace

Níže můžeme vidět příklad volání programu `nthderivative`. V následujících zdrojových kódech jsou pro větší přehlednost vynechány bloky `try-catch`.

```

temp = File.createTempFile("tempCoefs", ".txt", new File(pathTemp));
ProcessBuilder coefBuilder = new ProcessBuilder(path + File.separator +
        "nthderivative.exe", precision, left, right, row);
coefBuilder.redirectOutput(ProcessBuilder.Redirect.to(temp));
Process coefProcess = coefBuilder.start();
  
```

V další části programu probíhá volání funkcí ze skriptu napsaném v jazyce Python. Tento skript slouží, jak již bylo řečeno, pro vygenerování příslušných závorkových výrazů. Příklad volání funkce z tohoto skriptu můžeme vidět níže.

```

ScriptEngineManager factory = null;
ScriptEngine engine = null;

// získání enginu pro vyhodnocování skriptu
factory = new ScriptEngineManager();
engine = factory.getEngineByName("python");
// vyhodnocení skriptu
engine.eval(new java.io.FileReader(path));
Invocable inv;
inv = (Invocable) engine;
// volání funkce z Python skriptu
PyList bracketsBegin = new PyList();
bracketsBegin = (PyList) inv.invokeFunction("GenerateCombined",
        cutsInt, approxInt);
  
```

Pomocí funkcí pro generování rovnic je poté možné sestavit výsledné rovnice pro systémy TKSL. Výsledný zdrojový kód je zobrazen přímo v grafickém uživatelském rozhraní a je také ukládán do souboru. Tento soubor nese jméno `aproximace_pocetRezu.txt`. Tedy,

pokud zadame například dvanáctibodovou aproximaci a počet řezů pět, výsledný soubor se bude jmenovat 12.5.txt.

Kapitola 8

Závěr

Cílem práce bylo analyzovat problematiku paralelního numerického řešení parciálních diferenciálních rovnic (PDR). Nejdříve jsme se seznámili s obyčejnými diferenciálními rovnicemi (ODR) a ukázali, jak je možné tyto rovnice řešit pomocí Taylorovy řady. Tato problematika byla předvedena na RC článku. Následně byly rovnice popisující obvod RC článku převedeny do simulačního systému TKSL, který pro výpočet používá proměnný řád metody. Řád metody reprezentuje, kolik členů Taylorovy řady bylo v jednotlivých časových bodech využito. Bylo zjištěno, že snížení řádu metody lze dosáhnout snížením integračního kroku výpočtu. Další možností, jak snížit řád metody, je využít modifikovaného algoritmu výpočtu. Modifikace algoritmu spočívá v tom, že jakmile je aktuálně vypočítaný člen Taylorovy řady menší než daná přesnost, není tento člen již uvažován v dalších výpočtech a jeho hodnota je nastavena na nulu.

V další části práce jsme se seznámili s parabolickou, hyperbolickou a eliptickou PDR, vysvětlili jsme si jejich možné řešení pomocí převodu na soustavu obyčejných diferenciálních rovnic (ODR). Dále bylo předvedeno, jakým způsobem lze tyto typy PDR převést na soustavy ODR a následně je řešit v systému TKSL. Tento převod je nezbytný, protože systém TKSL umožňuje řešit pouze ODR.

Dále se práce zabývala metodami řešení PDR. Konkrétně jsme se zaměřili na hyperbolickou PDR, která popisuje kmitání struny. Pro řešení PDR lze využít dopřednou, zpětnou nebo kombinovanou metodu. Tyto metody využívají pro svůj výpočet n -bodové aproximační vzorce. Při výpočtu pomocí těchto metod vycházíme z Taylorovy řady. Pro ověření správnosti řešení byly vytvořeny programy v systému Matlab, pomocí kterých byla ověřena přesnost aproximace funkce $f(x) = -\sin(x)$, tedy druhé derivace funkce $f(x) = \sin(x)$. Bylo zjištěno, že dopředná i zpětná metoda vykazují stejnou chybu výpočtu, která je zapříčiněna použitím nesymetrických aproximačních vzorců. Naopak kombinovaná metoda dosahuje výrazně větší přesnosti aproximace, protože pro svůj výpočet využívá převážně symetrické aproximační vzorce. Ty způsobí, že akumulace chyby v průběhu výpočtu je nižší, než u dopředné a zpětné metody.

Práce se také zabývala paralelním řešením PDR. Vysvětlili jsme si, jak sestavit programové schéma pro libovolné parametry (tedy libovolný počet řezů na struně a libovolnou n -bodovou aproximaci). Díky tomu, že jsme schopni PDR převést na soustavu ODR, lze každou ODR reprezentovat jako samostatnou operační jednotku. Tento přístup nám zajišťuje významné zrychlení výpočtu, které je dáno počtem integrátorů v programovém schématu.

V rámci diplomové práce byl navržen program, který je schopen vygenerovat soustavu ODR pro systém TKSL. Tyto rovnice pak reprezentují hyperbolickou PDR s libovolnými

parametry.

Dalšími rozšířeními by mohla být optimalizace programu pro rychlejší generování výstupů a rozšíření jeho funkcionality o další typy PDR, dále aplikace navrženého algoritmu pro snížení řádu metody pro systém TKSL, analýza stability výpočtu dopředné, zpětné a kombinované metody a také podrobná analýza a možnosti dalších optimalizací paralelního výpočtu.

Literatura

- [1] *Úvod do teorie PDR*. [cit. 30-12-2013],
URL: http://mathonline.fme.vutbr.cz/download.aspx?id_file=787 [online].
- [2] Řezáč, D.: *Stiff Systems of Differential Equations and Modern Taylor Series Method*. Dizertační práce, FIT VUT v Brně, 2004.
- [3] Fajmon, B.; Růžičková, I.: *Matematika 3*. Brno: Vysoké učení technické v Brně – Fakulta elektrotechniky a komunikačních technologií.
- [4] Franců, J.: *Parciální diferenciální rovnice*. Brno: Vysoké učení technické v Brně – Fakulta strojního inženýrství, 2003.
- [5] Kraus, M.: *Parallel Computer Systems based on numerical integrations*. Dizertační práce, FIT VUT v Brně, 2013.
- [6] Kubíček, M.; Dubcová, M.; Janovská, D.: *Numerické metody a algoritmy*. VŠCHT Praha, 2005, ISBN 80-7080-558-7.
- [7] Kunovský, J.: *Modern Taylor series method*. Habilitation work, VUT v Brně.
- [8] Kunovský, J.: *Teorie obvodů*. Brno: Vysoké učení technické v Brně – Fakulta informačních technologií, 2008.
- [9] TKSL software: *High Performance Computing*. [cit. 4-1-2014],
URL: <http://www.fit.vutbr.cz/~kunovsky/TKSL/index.html.en> [online].
- [10] Rektorys, K.: *Přehled užití matematiky*. Praha: Prometheus, 2009, ISBN 978-80-7196-180-2.
- [11] Vitásek, E.: *Základy teorie numerických metod pro řešení diferenciálních rovnic*. Academia, Praha, 1994, ISBN 80-200-0281-2.

Příloha A

Obsah příloženého CD

Příložené CD obsahuje následující položky:

- soubor `DP_Necasova.pdf` – technická zpráva diplomové práce,
- složka `DP_Text` – zdrojové kódy technické zprávy (soubory `.tex`),
- složka `DP_UI` – zdrojové kódy grafického uživatelského rozhraní PDE Generator,
 - `TKSLGenerator_GUI` – zdrojové kódy v jazyce Java,
 - `TKSLGenerator` – zdrojové kódy v jazyce Python pro vygenerování závorkových výrazů,
 - `CGenCoefs` – zdrojové kódy v jazyce C zajišťující generování koeficientů inverzních matic,
 - `CGenSinValue` – zdrojové kódy v jazyce C zajišťující generování počátečních podmínek výpočtu,
- složka `DP_Matlab` – zdrojové kódy pro systém Matlab,
- složka `DP_Taylor` – zdrojové kódy v jazyce C++, které byly využity k experimentům s proměnným řádem integrační metody.