Základy programování v Euler pro předmět PPAŘ

Introduction to programming in Euler for subject denoted as Computer Aided Automation Control

Ondřej Hadač

Bakalářská práce
2008

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky
Ústav aplikované informatiky
akademický rok: 2007/2008

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení:	**Ondřej HADAČ**
Studijní program:	**B 3902 Inženýrská informatika**
Studijní obor:	**Informační technologie**
Téma práce:	**Základy programování v Euler pro předmět PPAŘ**

Zásady pro vypracování:

1. **Seznamte se s prací v programu Euler a českým manuálem tohoto software.**
2. **Vytvořte programy ve Euler pokrývající obsahově celou výuku programování v MATLABu na FAI.**
3. **Seznamte se dále s prací v programu Octave a s jeho českým manuálem. Porovnejte odlišnosti syntaxe vašich programů v Euler vůči Octave.**
4. **Dále uveďte rozdíly v syntaxi vámi vytvořených programů v Euler vůči syntaxi MATLABu.**
5. **Vámi vytvořené programy v Euler umístěte na samostatné CD-ROM jako přílohu bakalářské práce.**

Rozsah práce:

Rozsah příloh:

Forma zpracování bakalářské práce: **tištěná/elektronická**

Seznam odborné literatury:

1. **Euler Dokumentation – http://mathsrv.ku-eichstaett.de/MGF/homes/grothmann/euler/german/index.html**
2. **Kreizlová, Z. (2007): Euler – průvodce v češtině. Bakalářská práce, UTB ve Zlíně, Fakulta aplikované informatiky.**
3. **Český průvodce programem Euler – http://euler.euweb.cz/**
4. **Perůtka (2005): MATLAB – Základy pro studenty automatizace a informačních technologií. Skriptum, 1. vydání, UTB ve Zlíně, Zlín.**
5. **Just, M. (2006): Český průvodce programem Octave. Bakalářská práce, UTB ve Zlíně, Fakulta aplikované informatiky.**

Vedoucí bakalářské práce: **Ing. Karel Perůtka, Ph.D.**
Ústav řízení procesů

Datum zadání bakalářské práce: **20. února 2008**

Termín odevzdání bakalářské práce: **5. května 2008**

Ve Zlíně dne 20. února 2008

UNIVERZITA TOMÁŠE BATI VE ZLÍNĚ
Fakulta aplikované informatiky

L.S.

prof. Ing. Vladimír Vašek, CSc.
děkan

doc. Ing. Ivan Zelinka, Ph.D.
ředitel ústavu

ABSTRAKT

Bakalářská práce se zabývá vytvářením programů ve výpočetním softwaru Euler Math Toolbox, který je volně šiřitelnou alternativou komerčního softwaru Matlab. Práce poskytuje studentům pomůcku při výuce programování v předmětu Programová podpora automatického řízení v podobě ilustračních programů v alternativním software. Dále uvádí rozdíly v syntaxích mezi Euler Math Toolbox a Matlab, ale i GNU Octave, což je další, volně šiřitelný software, což nezbytné pro převod kódu mezi těmito programy. Bakalářská práce by měla sloužit studentům jako nezbytný základ k programování v programu Euler Math Toolbox zejména při procvičování programování doma nebo na kolejích.

Klíčová slova: Euler Math Toolbox, Matlab, GNU Octave, programování

ABSTRACT

This bachelor thesis deals with programming in Euler Math Toolbox software. Euler Math Toolbox is free, open-source alternative to the commercial software Matlab. Creating the set of sample programmes in Euler Math Toolbox was the main goal of the bachelor thesis. This set should help students with programming in the subject denoted as Computer aided automation control. Furthermore, it describes the differences in the syntax language between Euler Math Toolbox and Matlab. The differences in the syntax language of GNU Octave, which is another open-source software, are provided as well. This bachelor thesis should make the programming in Euler Math Toolbox easier for all students who might be interested in it.

Keywords: Euler Math Toolbox, Matlab, GNU Octave, programming

Rád bych tímto poděkoval vedoucímu bakalářské práce, Ing. Karlu Perůtkovi, Ph.D., za jeho rady, připomínky ale i za odbornou výuku programování v Matlabu, díky které jsem si prohloubil vědomosti a stala se základem pro psaní této práce.

Nejvíce si však cením zázemí, které mi poskytla moje rodina a za to jsem ji neskonale vděčný. Bez nich by tato práce nikdy nevznikla.

Motto

“

Z ničeho se nemá dělat věda. Ani z vědy ne, natož pak ze života.

”

JAN WERICH (* 1905 – †1980)

Prohlašuji, že jsem na bakalářské práci pracoval samostatně a použitou literaturu jsem citoval. V případě publikace výsledků, je-li to uvolněno na základě licenční smlouvy, budu uveden jako spoluautor.

Ve Zlíně …….

Podpis diplomanta

OBSAH

ÚVOD

Úkolem této bakalářské práce je vytvořit sadu programů v softwaru Euler Math Toolbox, pokrývající obsahově výuku předmětu *Programová podpora automatického řízení*. Tento předmět je vyučován v oborech Chemické a procesní inženýrství a Inženýrská informatika na Fakultě aplikované informatiky Univerzity Tomáše Bati ve Zlíně.

Studenti se v předmětu seznamují zejména se softwarem Matlab a jeho blokově orientovanou nadstavbou Simulinkem. Ačkoliv se jedná o software, jež je celosvětově rozšířen a využíván v mnoha odvětvích (pro vědeckotechnické výpočty, měření a zpracování signálů, návrhu řídicích a komunikačních systémů, simulaci dynamických systémů,…), nejedná se o jediný programový systém. Mezi další patří GNU Octave, Scilab, ale rovněž i Euler Math Toolbox, či zkráceně EuMathT, který zde bude probrán.

Teoretická část práce obsahuje souhrn informací o prostředí EuMathT. V této části jsou popsána následující témata:

- Popis prostředí a seznámení se se softwarem
- Datové typy a proměnné
- Operátory
- Vektory, matice, řetězce
- Práce se soubory
- Vykreslování 2D a 3D grafů
- Tvorba vlastních skriptů a funkcí

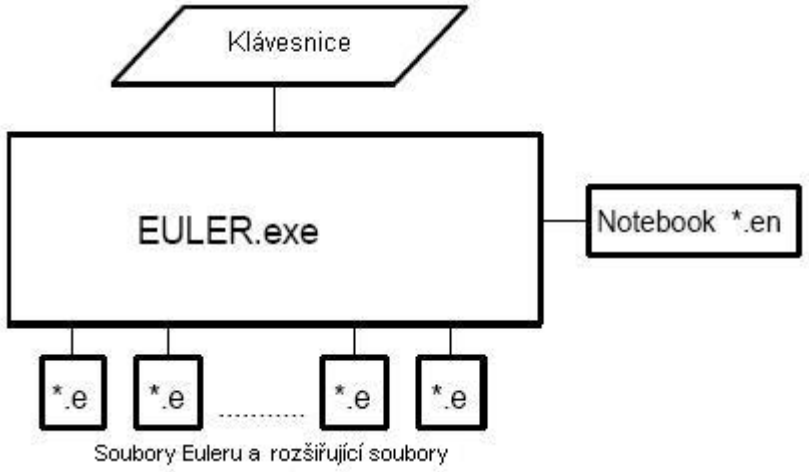
Veškerá práce v Euler Math Toolboxu je porovnávána s Matlabem a GNU Octave, zejména je kladen důraz na syntaxi. Pro každou teoretickou kapitolu byl vytvořen demonstrační program v Euler Math Toolboxu, jež je souhrnem popisované látky. Součástí těchto programů je i kód v Matlabu, reprezentující stejný výsledek.

Praktická část pak obsahuje samostatné programy, které ukazují schopnosti Euler Math Toolboxu. Vše za použití příkazů, zmíněných v teoretické části.

I. TEORETICKÁ ČÁST

1 POPIS PROSTŘEDÍ

Euler Math Toolbox, dříve označovaný jako Euler, je volně šiřitelný výpočetní software pod licencí GNU General Public Licence, vyvíjený R. Grothmannem, profesorem matematiky na Univerzitě v Eichstättu. Samotný software byl napsán v objektově orientovaném jazyce C++. Ačkoliv není tak obsáhlý, jako např. Matlab či GNU Octave, je velmi flexibilní. A to z toho důvodu, že jej lze rozšířit díky rozšiřujícím souborům (s příponou "`*.e`") a může tak být snadno nasazen pro řešení konkrétní úlohy. Strukturu programu znázorňuje obrázek níže:



Obrázek 1: Struktura programu Euler Math Toolbox

Software Euler Math Toolbox je svoji koncepcí podobný programům Notizblock či Notebook. Stejně jako v Notizblocku, i v Euler Math Toolboxu lze psát řádky s textem nebo matematickými vzorci. Po skončení práce je k dispozici pracovní list Euler Math Toolboxu, který může být uložen do souboru. Takový list se nazývá notebook. Soubory notebooků dostanou při uložení příponu "`*.en`". Notebook je přehledně dělen na vstupy (kam je možné zadávat sekvenci příkazů a komentáře) a výstupy. Vstupy je možné, narozdíl např. od Matlabu, kdykoliv editovat a znovu vyhodnotit.

Euler Math Toolbox je interaktivní výpočetní nástroj. Umožňuje vyhodnocení numerických výrazů s reálnými a komplexními čísly, vektory a maticemi. Pracuje se speciálním datovým typem pro exaktní výpočty. Využití nachází i při vykreslování 2D a 3D grafů stejně tak i pro čtení a zápis dat (a to i binárních). Jednu z jeho předností představuje práce s intervaly.

EuMathT má vlastní programovací jazyk podobný jazyku Basic – v tomto jazyce je možné psát rozšiřující soubory pro Euler Math Toolbox. Veškeré výpočty a vykreslenou grafiku je možné exportovat do HTML.

Výhodou používání Euler Math toolboxu je spolupráce s dalšími volně šiřitelnými počítačovými algebraickými systémy Maxima a Yacas, které jsou distribuovány pod GNU General Public Licence.

1.1 Základy práce s EuMathT

Všechny vstupní příkazy se v Euler Math Toolboxu zobrazují červenou barvou (za znakem ">"), tělo funkce modrou (za znakem "$") a výstupní hodnoty černou. Komentáře se pak v notebooku zobrazí červeně, v rozšiřujícím souboru zeleně. Značení má za úkol zpřehlednit práci s programem.

Jak již bylo zmíněno, software Euler Math Toolbox je schopný uložit dva druhy souborů. Notebooky, tedy sekvence příkazů, které spolu nemusí vůbec souviset, mají příponu "*.en". Oproti tomu u souborů s příponou "*.e" se očekává, že budou obsahovat funkce, které se po načtení stanou součástí softwaru a je možné je kdykoliv zavolat. Pokud je požadováno načítání těchto funkcí po každém spuštění programu, měl by být uživatelem vytvořen seznam těchto rozšiřujících souborů a ten poté vložen do souboru *euler.cfg*. Je zde patrná analogie s M-soubory Matlabu s tím, že Matlabu stačí funkci načíst jednou a poté se stará o její načítání sám bez jakéhokoliv zásahu uživatele.

Pokud uživatel má znalosti s prací v Matlabu či GNU Octave, pak by neměli začátky práce v Euler Math Toolboxu činit větší potíže. Pokud by přece jen narazil na problém, základy práce s tímto softwarem jsou výborně zmapované v češtině, viz literatura [4] případně je možné využít návodů zmíněných v kapitole 11.2, Nápověda k programu.

2 DATOVÉ TYPY, KONSTANTY A PROMĚNNÉ

2.1 Datové typy

Datové typy programu Euler Math Toolbox lze rozdělit na 3 oblasti – datové typy pracující s reálnými čísly, komplexními čísly a řetězci. Každý datový typ je symbolicky označen číslem:

- `0` – reálná čísla
- `1` – komplexní čísla
- `2` – vektory a matice s reálnými prvky
- `3` – vektory a matice s komplexními prvky
- `8` – řetězce
- `10` – intervaly

Čísla 4, 5, 6 a 7 nemají přiřazen žádný datový typ.

2.2 Konstanty

V Euler Math Toolboxu existují mimo jiné proměnné, které symbolizují konstanty, nicméně lze kdykoliv přepsat jejich hodnotu. Podobně si počíná i Matlab a GNU Octave. Všechna vyhrazená slova pro konstanty lze v Euler Math Toolboxu vypsat příkazem `listvar`.

2.3 Proměnné

U proměnných platí identická pravidla jako v Matlabu či GNU Octave. Není třeba definovat ani deklarovat jejich datové typy. Proměnné musí začínat písmenem, po kterém mohou následovat čísla či písmena (nesmí obsahovat podtržítko, pomlčku ani další speciální znaky). Euler Math Toolbox obdobně jako Matlab či GNU Octave je "case sensitive", čily klade důraz na velikost písmen – proměnné „`retezecA`" a „`retezeca`" jsou pro program dvě zcela odlišné proměnné. Euler Math Toolbox zachovává s Matlabem i GNU Octave kompatibilitu pro globální a lokální proměnné. Tedy že s lokální proměnnou funkce není možné mimo ni pracovat. Naopak to neplatí.

2.3.1 Testování datového typu proměnné

Pro zjištění datového typu proměnné se používá funkce `typeof`, která vrací číslo, reprezentující datový typ. Dále k testování proměnných slouží další funkce, které vrací 1, pokud je výsledek pozitivní – v opačném případě 0.

- `isreal` – testuje, zda-li je proměnná číslo reálné (v Matlabu i GNU Octave je identický příkaz)
- `iscomplex` – testuje, zda-li je proměnná číslo komplexní (GNU Octave obsahuje identickou funkci, v Matlabu bychom mohli např. zjistit, zda-li je imaginární složka nulová pomocí funkce `imag`)
- `isinterval` – testuje, zda-li je proměnná interval (Matlab ani GNU Octave žádný takový příkaz nemají)
- `isstring` – testuje, zda-li je proměnná řetězec (v Matlabu i GNU Octave příkazem `ischar`)
- `isvar` – testuje, zda-li se jedná o proměnnou (v Matlabu příkazem `isvarname`, GNU Octave funkci nenabízí)
- `isfunction` – testuje, zda-li je proměnná funkcí

2.3.2 Přetypování proměnné

Euler Math Toolbox nenabízí mnoho příkazů pro přetypování, což je dáno rozsahem datových typů, které zná.

Převod mezi reálnými a komplexními čísly je popsán v Kapitole 6: Práce s komplexními čísly.

Pro převod čísla na řetězec a řetězce na číslo se používá funkcí:

- `printf` – převede číslo na řetězec (Matlab i GNU Octave disponují funkcí `num2str`, případně Matlab má i funkci `sprintf`)
- `evaluate`, `interpret` – (Matlab i GNU Octave používají funkci `str2num`)

3 OPERÁTORY

Euler Math Toolbox rozlišuje, stejně jako Matlab či GNU Octave, několik typů operátorů.

První typ, relační operátory, se v naprosté většině shodují s operátory v Matlabu – patří mezi ně <, >, ==, ≤, ≥. Poslední z nich, operátor pro nerovnost, se od Matlabu liší. Zatímco v Euler Math Toolboxu je syntaxe !=, Matlab používá ~= (GNU Octave zná obě varianty).

Druhý typ, logické operátory, se rovněž shodují v naprosté většině. Výjimku tvoří opět negace v Euler Math Toolboxu se syntaxí ! oproti syntaxi Matlabu ~. Ostatní operátory ||, && mají zcela stejné použití. Zachována je i priorita operátorů (nejvyšší prioritu má negace a nejnižší logický součet). Pokud si nejsme prioritou vyhodnocování složitějšího výrazu jistí, měli by se použít závorky.

Posledním typem jsou operátory matematické, které mají ve všech uvedených softwarech stejné užití. Jedná se o operace sčítání (+), odčítání (−), dělení (/), násobení (×), umocňování (^).

4 PODMÍNKY A CYKLY

Veškerá klíčová slova pro logické větvení a cykly mohou být používána pouze ve funkcích, nikoliv jako interaktivní příkazy v textovém okně.

4.1 Podmínky

Klíčová slova pro podmínky jsou `if`, `elseif`, `else` a `endif`. Matlab i GNU Octave užívá identické příkazy. Použití všech klíčových slov včetně obecné syntaxe je stejné a podmínky budou vracet pro Euler Math Toolbox, Matlab i GNU Octave stejné výsledky.

Jediný rozdíl oproti Matlabu a GNU Octave je ten, že Euler Math Toolbox nezná mnohonásobné větvení pomocí funkce `switch`.

4.2 Cykly

Klíčová slova pro cykly jsou `repeat`, `loop`, `for` a `break`. Euler Math Toolbox navíc užívá pro cykly interního generovaného čítače, který má symbol `#` nebo `index()` – díky němu je tak možné např. větvit program pro jednotlivé iterační kroky. Matlab i GNU Octave užívají z výše uvedených příkazů pouze `for` a `break`.

4.2.1 Cyklus loop

Tento cyklus je velmi podobný cyklu `for`. Po vyhodnocení konstant `m` a `n` je zjištěno, zda-li se má cyklus vůbec provádět (neprovede se, pokud bude `m` větší než `n`). Při prvním provedení cyklu je inicializováno počítadlo cyklů (symbol `#` či `index()`) a po každé iteraci je inkrementováno.

Obecná syntaxe:

```
loop m to n;
tělo_cyklu;
end;
```

Autor uvádí, viz literatura [5], že cyklus `loop` je rychlejší než `for` právě díky přístupu k internímu čítači. Při opakovaném testování pro 10 000 000 iterací byla tato skutečnost dokázána – při výpočtu stejného kódu a za identických podmínek.

Pro porovnání rychlosti cyklů byl vytvořen testovací program, který je uveden níže. Zároveň je uložen v souboru *program-kapitola-4-2.en*.

```
>function smyckaFOR(pocet)
$a=0;$start=time();
$for i=1 to pocet;
$a=a+i*0.123456789;
$end;
$konec=time();
$return (konec-start);
$endfunction;
>smyckaFOR(10000000)
                  29.078
>function smyckaLOOP(pocet)
$a=0;$start=time();
$loop 1 to pocet;
$a=a+#*0.123456789;
$end;
$konec=time();
$return (konec-start);
$endfunction;
>smyckaLOOP(10000000)
                   25.64
>function smyckaREPEAT(pocet)
$a=0;i=1;
$start=time();
$repeat
$if(i==pocet) break;endif;
$a=a+i*0.123456789;
$i=i+1;
$end;
$konec=time();
$return (konec-start)
$endfunction;
>smyckaREPEAT(10000000)
                   45.312
```

4.2.2 Cyklus repeat

Tento cyklus probíhá nekonečněkrát – vyskočit z něj lze pouze příkazem `break`. Ukončení cyklu probíhá často logickým větvením (podmínka `if`).

Obecná syntaxe:

```
repeat;
tělo_cyklu
end;
```

4.2.3 Cyklus for

Cyklus `for` se v Euler Math Toolboxu od stejného cyklu v Matlabu či GNU Octave liší pouze syntaxí. Stejné mají vše ostatní, včetně faktu, že inkrementace o krok `o` (`step`), může být vynechána (krok poté bude nastaven na hodnotu plus jedna).

Obecná syntaxe:

```
for i=m to n step o;
tělo_cyklu;
end;
```

Syntaxe Matlabu či GNU Octave je uvedena níže. Je patrné, že oproti EuMathT není nutné vše ukončovat středníkem.

Obecná syntaxe:

```
for i=m:o:n
tělo_cyklu
end
```

Popis proměnných v kódu:

`i` – jméno proměnné čítače

`m` – počáteční hodnota čítače

`n` – koncová hodnota čítače

`o` – krok, o který se bude iterovat

5 VEKTORY A MATICE

Euler Math Toolbox je podobně jako Matlab, maticově založený software. Matice a vektory jsou tudíž pro něj základním „stavebním kamenem“.

5.1 Základní informace

Matice v Euler Math Toolboxu je uspořádání čísel v řádcích (rows) a sloupcích (columns). Čísla jsou, stejně jako u vektorů, uzavřena v hranatých závorkách. Jednotlivé prvky matic mohou být reálné nebo komplexní. Většinu příkazů pro práci s maticemi je možné použít také pro práci s vektory. Matice [m,n] má m řádků a n sloupců. Řádky a sloupce matice se počítají od indexu 1, nikoliv od nuly.

Vektor je speciální typ matice, jejíž jedna dimenze má index 1. Pokud je rozměr [1,n], mluvíme o řádkovém vektoru, při rozměru [n,1] jde o sloupcový vektor.

5.2 Rozdíly při práci s vektory a maticemi

Rozdíly v syntaxi mezi výpočetními programy Euler Math Toolbox, Matlab a GNU Octave byly rozděleny do třech kategorií. V první jsou uvedeny ty funkce, které mají identické jméno a v drtivé většině generují i identický výstup (pokud tomu tak není, je jejich popis o tuto skutečnost doplněn a rozdíly jsou vysvětleny na demonstračních příkladech přiložených na CD-ROM). Analogicky je strukturována i druhá kategorie, s tím rozdílem, že funkce se jmenují jinak. Třetí část pak obsahuje funkce, pro které Matlab či GNU Octave nemá adekvátní alternativy.

5.2.1 Vytváření a výpis

Zde žádné syntaktické rozdíly nejsou – při ručním vytváření matice se jednotlivé prvky sloupců od sebe oddělují čárkami (nebo mezerami), jednotlivé řádky pak středníkem. Vše je nutné uzavřít do hranatých závorek. Vektor/matici však můžeme generovat i automaticky – operátorem dvojtečky (syntaxe je `prvniprvek:krok:posledniprvek`) nebo pomocí níže uvedených funkcí.

Při vytváření matic je však možné na jednu odlišnost narazit. Při definování prvků matice v Matlabu je možné vhodným formátováním (zalamováním řádků) určovat její dimenze – toto je vysvětleno na přiloženém CD-ROM.

Výpis matic lze provést podobně jako v Matlabu, avšak s malými rozdíly:

1) EuMathT nezná ukončovací prvek – klíčové slovo – `end`.
2) EuMathT nedokáže přistoupit v matici (2D poli) k prvku přes jeden index. Pro takovouto matici je nutné užít vždy dvou indexů.
3) v EuMathT není nutné při vytváření matice zadávat všem řádkům stejný počet prvků. Software sám doplní nulami řádky tak, aby měli stejnou délku, jako řádek s největším počtem prvků – Matlab by ohlásil chybu.

5.2.2 Operátory

Euler Math Toolbox používá jednoduchých operátorů, pro vykonání specifických operací:

- `'` (**horní apostrof**) – transponování vektorů/matic (Matlab i GNU Octave užívají stejného operátoru)
- `_` (**podtržítko**) – spojování vektorů/matic vertikálně, tzv. "pod sebe" (v Matlabu je možné použít např. funkci `vertcat`, v GNU Octave je např. spojíme do jedné matice pomocí středníku `;`)
- `|` (**roura**) – spojování vektorů/matic horizontálně, tzv. "vedle sebe" (v Matlabu je možné použít např. funkci `horzcat`, v GNU Octave je např. spojíme do jedné matice pomocí čárky `,`)

5.2.3 Operace a manipulace

Některé operace nám vrací stejný výstup, jako software Matlab. Patří sem:

- Přičítání, odečítání konstanty od všech prvků matice
- Násobení, dělení všech prvků matice konstantou
- Sčítání, odčítání dvou matic
- Násobení dvou matic po prvcích
- Dělení dvou matic po prvcích
- Dělení matice maticí zleva

Některé operace, na které může být uživatel Matlabu zvyklý, se provádí odlišně:

- Maticové násobení – místo operátoru "`*`", se kterým pracuje Matlab, je nutné použít operátoru tečky "`.`"
- Umocňování prvků matice – buď za pomoci "`^n`" nebo "`**n`", kde `n` je mocnina (Matlab užívá např. funkci `power`)
- Umocňování celé matice – místo operátoru "`^`", se kterým pracuje Matlab, musíme použít operátoru tečky "`.`"

Jiné základní operace, známé z Matlabu, pak nejsou implementovány vůbec:

- Dělení matice maticí zprava – Euler Math Toolbox takto není schopen za pomocí operátoru dělit

Při operaci s maticemi a vektory se lze, oproti Matlabu či GNU Octave, setkat pouze s drobnými odlišnostmi. Veškeré manipulace provádí to stejné, co Matlab a GNU Octave, pouze mají jiné jméno.

- `rotleft` – rotace řádků matice vlevo (první sloupec se posouvá na pozici posledního [Matlab ani GNU Octave funkci nenabízí])
- `rotright` – rotace řádků matice vpravo (poslední sloupec se posouvá na pozici prvního [Matlab ani GNU Octave funkci nenabízí])
- `shiftleft` – podobné jako `rotleft`, navíc se však první sloupec, po posunutí na pozici posledního, vynuluje
- `shiftright` – podobné jako `rotright`, navíc se však poslední sloupec, po přesunutí na pozici prvního, vynuluje
- `flipx` – rotace prvků kolem svislé osy (ekvivalent v Matlabu i GNU Octave – fce `flipud`)
- `flipy` – rotace prvků kolem vodorovné osy (ekvivalent v Matlabu i GNU Octave – fce `fliplr`)

5.2.3.1 Analýza matic

Pro analýzu matic a polí slouží v Euler Math Toolboxu sada příkazů, které ve většině případů mají v Matlabu svoji alternativu. Některé funkce se jmenují zcela identicky:

- `det` – determinant matice
- `inv` – inverzní matice
- `size` – číselná hodnota rozměru matice/pole
- `length` – číselná hodnota délky vektoru (u matice vrací počet sloupců)
- `rank` – hodnost matice
- `min, max` – nejmenší, největší prvek/prvky v poli
- `prod` – součin prvků vektoru, u matic pak sloupcový vektor součinů řádků (Matlab násobí jednotlivé sloupce)
- `sort` – setřídění prvků vektoru vzestupně
- `find` – indexy prvků (i nulových) seřazeného vektoru funkcí `sort`
- `sum` – součet prvků vektoru, u matic pak sloupcový vektor sum řádků (Matlab sčítá jednotlivé sloupce)

Další funkce mají odlišné jméno, ale vrací identický výstup:

- `eigen` – vlastní čísla matice (ekvivalent v Matlabu je `eig`)

Jiné funkce pak zná pouze EuMathT a GNU Octave:

- `cols, rows` – číselná hodnota počtu sloupců, řádků (`columns` a `rows` jsou příkazy GNU Octave)

5.2.4 Elementární a speciální matice a vektory

Pro generování specifických typů matic slouží příkazy, které jsou identické pro Matlab i GNU Octave:

- `zeros` – matice nebo vektor samých nul
- `ones` – jedničková matice nebo vektor

- `hilbert` – Hilbertova matice (ekvivalent v Matlabu je `hilbert`, GNU Octave užívá příkazu `hilb`)
- `logspace` – generování vektoru v logaritmickém prostoru
- `linspace` – generování vektoru v logaritmickém prostoru
- `diag` – diagonála matice, diagonální matice (požadovaný výstup je nezbytné upřesnit parametry funkce)

Další příkazy mají oproti Matlabu či GNU Octave různé názvy, ale stejný výstup:

- `id` – jednotková matice (ekvivalent v Matlabu i GNU Octave je `eye`)
- `random` – náhodně rovnoměrně distribuované pole (GNU Octave užívá identický příkaz, ekvivalent v Matlabu je `rand`)
- `band` – anuluje prvky matice, které leží mimo hranice, označené diagonálami

  Obecný zápis:

  `band(maticeA,celeCisloN,celeCisloM)`

  Anuluje prvky `maticeA`, které leží mimo diagonály `celeCisloN` a `celeCisloM`
- `extrema` – nejmenší a největší prvek ve vektoru včetně indexu výskytu (ekvivalent v Matlabu jsou funkce `min`, `max`)

Jiné příkazy pak v Matlabu a GNU Octave nemají adekvátní příkaz:

- `matrix` – matice o definovaných rozměrech, jejíž veškeré prvky jsou nastaveny na stejnou hodnotu
- `dup`, `redim` – vytvoří matici z vektoru (obě mají rozdílný výstup – ukázku je možné nalézt na CD-ROM)
- `shuffle` – náhodně zamíchá vektor

5.2.5 Submatice

Submatice se, stejně jako vektory, často používají k dosažení efektů komplexní manipulace s daty. Klíčem k tomu je tzv. "dvojtečkový zápis" (který se používá jak pro generování vektorů tak i pro odkazování na submatice) a indexování pomocí vektorů. Jejich využívání dovoluje omezit užití cyklů, které více zatěžují procesor. Výpis submatic je identický jako v Matlabu a GNU Octave, proto zde není vysvětlen – několik málo ukázek obsahuje přiložené CD-ROM.

5.2.6 Testování matic

Tyto funkce nejsou v samotném programu Euler Math Toolbox implementovány. Matice je možné testovat pouze za pomoci softwaru Yacas.

Nicméně v praktické části byli tyto funkce naprogramovány jako rozšiřující soubor – jejich obsah je možné nalézt na přiloženém CD-ROM.

6 PRÁCE S KOMPLEXNÍMI ČÍSLY

Komplexní číslo, tedy číslo, které vzniklo rozšířením reálného tak, aby v něm každá algebraická rovnice měla příslušný počet řešení podle základní věty algebry, je v Euler Math Toolboxu chápáno zcela stejně jako v Matlabu, GNU Octave a jím podobných software. Při práci v Euler Math Toolboxu proto, kromě jediného rozdílu, na žádné syntaktické odlišnosti oproti předchozím dvěma softwarům nelze narazit.

Touto rozdílností je komplexní jednotka. V předchozích verzích byla komplexní jednotka reprezentována symbolem `i` ale i `j`. Nyní je to pouze `i`.

Při počítání v oboru komplexní čísel se používají následující příkazy:

- `complex` – explicitní přetypování proměnné do oboru komplexních čísel

  Obecný zápis:

  ```
  complex(a)
  ```

- `isreal` – testování, zda-li je číslo reálné
- `iscomplex` – testování, zda-li je číslo komplexní
- `re`, `im` – vrací reálnou, imaginární složku čísla (Matlab používá funkce `imag` a `real`)

Funkce `complex` se liší vůči syntaxi Matlabu. V Matlabu vyžaduje dva vstupní argumenty (první pro reálnou a druhý pro imaginární složku čísla) na místo jednoho v Euler Math Toolboxu.

Více funkcí pro práci s komplexními čísly samotný Euler Math Toolbox nezná, ale je možné si je naprogramovat jako rozšiřující soubory případně využít funkcí z programu Yacas.

7 ŘETĚZCE A PRÁCE S NIMI

Řetězec je chápán jako posloupnost znaků uzavřená do uvozovek. Narozdíl od Matlabu či GNU Octave, Euler Math Toolbox neinterpretuje řetězec jako jednorozměrné pole (vektor), který obsahuje znaky anglické abecedy.

Řetězcům je v Euler Math Toolboxu přidělen vlastní datový typ – ten je označen číslem 8.

Krátkých textů se zde využívá zejména v případech, kdy je nutné jako parametr předat funkci název souboru či výpis hlášky (např. chybové).

7.1 Vytváření řetězců

Řetězec musí být na obou stranách uzavřen v uvozovkách či dvojicí jednoduchých uvozovek. Zde tedy je oproti Matlabu či GNU Octave syntaktická odchylka, neboť v těchto softwarech je možné text uzavřít jedinou jednoduchou uvozovkou.

Dále, narozdíl od Matlabu či GNU Octave, není možné vytvářet matici řetězců (a to ani za předpokladu, že veškeré řetězce mají stejnou délku), tedy např. následující zápis vyvolá chybové hlášení:

```
maticeJmen=["jirina","jaroslav","jitka","jiri"]
```

Jak již bylo zmíněno výše, řetězec není chápán jako jednorozměrné pole, tedy i následující výpis vrátí chybu:

```
testovaciRetezec="Vysledek operace je OK"
testovaciRetezec[1]
```

7.2 Práce s řetězci

Kromě funkcí uvedených pod textem, je možné použít i některé operátory. Většina z nich slouží k porovnávání dvou řetězců (==,< ,> ,< =,> =,<>), kdy se srovnávají vždy znaky na stejných pozicích. Pro sloučení dvou řetězců je možné užít operátoru "|" (**roura**).

V software EuMathT jsou dostupné tyto funkce:

- `stringcompare` – porovnává dva řetězce; vrací 0 při rovnosti, -1 pokud je 2. výraz "větší" a 1 pokud je větší 1. výraz; porovnávají se postupně znaky na stejných indexech – větší je ten, který je v abecedě blíže k začátku (Matlab i GNU Octave má funkci `strcmp`)
- `substring` – vrací část řetězce, ohraničeného dvěma indexy, ze zdrojového řetězce (Matlab vypisuje jednotlivé znaky pomocí indexů, např. `retezec(1:5)`)

  Obecný zápis:
  `substring(zdroj,poziceM,poziceN)`

- `strlen` – délka řetězce (Matlabu i GNU Octave má funkci `length`)
- `strfind` – vrací pozici výskytu části řetězce ve zdrojovém řetězci; třetím argumentem je pozice, od které se má vyhledávat (Matlab i GNU Octave užívá funkci `findstr`, Matlab pak navíc i `strfind`, která vrací také pozici výskytu)

  Obecný zápis:
  `strfind(zdroj,hledanyRetezec,pozice)`

- `ascii` – ASCII kód prvního znaku řetězce (Matlab má např. syntaxi `'znak'-0`, GNU Octave užívá `toascii`)
- `char` – znak pro daný ASCII kód (Matlab i GNU Octave mají stejný příkaz)
- `tolower`, `toupper` – převede řetězec na malá, resp. velká písmena (GNU Octave má stejné příkazy, Matlab pak `lower` a `upper`)

8 PRÁCE SE SOUBORY

Přesný postup pro práci se soubory včetně všech příkazů pro čtení, zápis, otevírání a zavírání souborů je popsáno v literatuře [4] a [5]. Tato kapitola je zaměřena pouze na syntaktické odlišnosti a rozdílnou práci oproti Matlabu a GNU Octave.

První z nich je ta, že Euler Math Toolbox není schopen načítat soubor přes dialogové rozhranní jako Matlab.

Další odlišností je škála operací se soubory a adresáři. Euler Math Toolbox, narozdíl od GNU Octave, neumí např. mazat a vytvářet adresáře. Oproti Matlabu pak nejsou v Euler Math Toolboxu vytvořeny funkce pro čtení a import dat specifických souborů (CSV, XML, XLS,…).

Nicméně základní operace se soubory a adresáři Euler Math Toolbox podporuje.

8.1 Otevření, čtení souboru

Některé funkce jsou velmi podobné těm z Matlabu a GNU Octave:

- `open` – otevře datový soubor v adresáři (Matlab i GNU Octave používají `fopen`). Funkce má dva parametry (jméno souboru a mód operace). Počet módů operací se soubory je omezen na čtyři – čtení (`r`), zápis (`w`), binární čtení (`rb`), binární zápis (`wb`).

  Obecný zápis:
  ```
  open(jmeno_souboru.pripona,mod_operace)
  ```

Jiné pak Matlab ani GNU Octave nezná, avšak s oběma softwary lze dosáhnout stejného výsledku, pouze je nutné jiného přístupu pro čtení:

- `get*` – pod touto zkratkou se skrývá celá řada funkcí se specifickým názvem a specifickým způsobem čtení. EuMathT tak obsahuje funkce jako `getchar` (čte znak o velikosti 1 B), `getword` (čte znak o velikosti 2 B), `getlongword` (čte znak o velikosti 4 B), `getline` (čte celý řádek), `getstring` (čte počet znaků řetězce, definovaný jako parametr funkce), `getvector` (čte číslo nebo vektor), `getmatrix` (čte matici nebo vektor). Funkce, které čtou znak o stanoveném počtu bytů, ho čtou buď se znaménkem (viz funkce výše) nebo bez znaménka – funkce má pak tvar jako `getuword`, `getuchar`,…).

Matlab i GNU Octave jsou schopny stejného způsobu čtení např. pomocí funkce `fscanf`. Avšak konkrétní načítání vektorů a matic přes specifickou funkci v Matlabu ani v GNU Octave definováno není.

8.2 Zavření, zápis do souboru

Stejně jako v předchozí kapitole, i zde jsou funkce velmi podobné těm z Matlabu a GNU Octave.

- `close` – zavření jednoho nebo více otevřených souborů (Matlab i GNU Octave používají `fclose`)
- `put*` – pod touto zkratkou se skrývá celá řada funkcí se specifickým názvem a specifickým způsobem zápisu. Euler Math Toolbox tak obsahuje funkce jako `putchar` (zapisuje znak o velikosti 1 B), `putword` (zapisuje znak o velikosti 2 B), `putlongword` (zapisuje znak o velikosti 4 B). Funkce, které zapisují znak o stanoveném počtu bytů, ho zapisují buď se znaménkem (viz funkce výše) nebo bez znaménka – funkce má pak tvar jako `putuword`, `putuchar`,…).

  Matlab i GNU Octave jsou schopny stejného způsobu zápisu např. pomocí funkce `fprintf`.
- `write` – zápis řetězce (Matlab i GNU Octave používají `fprintf`)
- `printformat` – zápis čísla (Matlab i GNU Octave používají `fprintf`)

8.3 Další příkazy pro práci se soubory

Z dalších příkazů se EuMathT s Matlabem a GNU Octave shodují pouze v několika málo funkcích.

- `eof` – testování konce souboru (Matlab i GNU Octave používají `feof`)
- `remove` – smaže soubor (Matlab užívá `delete`, GNU Octave `unlink`)

Další z funkcí jsou specifické pouze pro program Euler Math Toolbox a GNU Octave ani Matlab k nim nenabízejí žádnou alternativu. Obecný zápis i význam jednotlivých parametrů funkcí je dobře vysvětlen v literatuře [5].

- `writematrix` – zápis matic a vektorů
- `readmatrix` – načtení matice, není nutné znát její rozměr
- `getmatrix` – podobně jako `readmatrix` načte matici, je však nutné znát rozměr

9 2D A 3D GRAFY

9.1 2D grafy

Pro vykreslování 2D grafů EuMathT nabízí celou řadu funkcí (`xplot, fplot`, `lplot` a další – více v citované literatuře [5]), stejně jako Matlab či GNU Octave.

Nejčastěji využívané funkce jsou zřejmě `plot` a `plot2d`. Obě umožňují vykreslení grafů s tím, že druhá zmiňovaná má širší možnosti užití. Právě funkce `plot2d` je rovnocenná alternativa k funkci `plot`, užívané Matlabem a GNU Octave. Díky ní je možné s grafem manipulovat (otáčet, přibližovat/oddalovat, posouvat se), omezit oblast vykreslení, měnit styl čáry (tloušťka, barva, typ), měnit typ souřadných os – všechny tyto vlastnosti je nezbytné, narozdíl od Matlabu, před výpočtem zadat do parametrů grafu, jinak nebudou zpřístupněny. Z toho je patrný další rozdíl a to ten, že oproti Matlabu je vše nutné zadávat textově, nelze tedy některé vlastnosti grafu dodatečně měnit jako v Matlabu v tzv. „Property editoru“.

Stejně jako v Matlabu či GNU Octave je možné vykreslovat více grafů do jednoho okna a také export grafu – dostupný je formát EPS (přípona souborů `*.ps`) a PNG (přípona souborů `*.png`)

9.1.1 Speciální 2D grafy

EuMathT neobsahuje funkce pro vykreslování speciálních typů grafů jako např. koláčový či plošný graf. Některé další typy grafů však zná. Patří mezi ně histogram, svislý sloupcový graf a parametrické grafy. Pro vykreslení prvních dvou zmiňovaných postačí do funkce `plot2d` doplnit další parametry, které zajistí požadovaný výstup. Parametrické grafy, tedy grafy s proměnnými `x` a `y`, jež obě mají stejný parametr, lze vykreslit pomocí funkce `plot`.

9.2 3D grafy

3D grafy v Euler Math Toolboxu, tedy grafy prostorově znázorňující funkce dvou proměnných, mají obecný způsob vytváření stejný, jako Matlab či GNU Octave. Stejně tak i příkazy jsou velmi podobné a vytváření 3D grafů ve všech třech softwarech by nemělo činit potíže. EuMathT nabízí sadu příkazů, z nichž je zmíněno pouze několik.

Základní funkce `solid` definovaná hodnotami `x`, `y` a `z` má adekvátní alternativu v Matlabu a GNU Octave `meshgrid`. Další funkcí je `plot3d` (Matlab a GNU Octave mají `plot3`) a `mesh`. Použití všech funkcí je demonstrováno na příkladech na přiloženém CD-ROM. Podrobnější informace o funkcích je možné nalézt v literatuře [4].

9.3 Animace

Animace je v Euler Math Toolboxu chápána jako sekvence snímků, které se postupně promítají s definovanou prodlevou. Je možné je pouze prohlížet (tedy nikoliv exportovat). Matlab i GNU Octave (za pomoci vykreslovacího nástroje Gnuplot) užívají pro vytváření videa či animace stejný postup, nabízí však daleko víc.

Dostupné funkce pro vytvoření animace v Euler Math Toolbox jsou `addpage`, `deletepages`, `showpage`, `pages` a zejména pak `animate`.

10 E-SOUBORY

E-soubory jsou soubory s příponou `*.e`, které umožňují uživatelům psát vlastní funkce. Tyto E-soubory poté mohou být zařazeny jako tzv. rozšiřující soubory Euler Math Toolboxu, čím zvýší jeho uplatnění. Vlastní kód se píše v interním programovacím jazyce, který je podobný např. jazyku Basic.

Při psaní vlastních funkcí jsou uplatněny znalosti ze všech předchozích kapitol. Rovněž právě při programování vlastních funkcí je možné využívat cykly a podmínky.

Psaní E-souborů je analogické psaní M-souborů v Matlabu či souborů pro funkce GNU Octave. Pokud má uživatel osvojené techniky z jednoho z těchto dvou software, nemělo by činit potíže psát rozšiřující soubory pro EuMathT. Každý uživatel by měl mít vždy na mysli, stejně jako při programování v jakémkoliv jiném jazyce, dodržování obecných zásad psaní (ať již se to týká vhodného formátování kódu, jeho komentování, názvů proměnných, …) kódu.

Obecná syntaxe je identická jak pro Matlab, tak pro GNU Octave, včetně klíčových slov. Uvození a ukončení funkce je provedeno dvojicí `function` a `endfunction`. Rovněž návratová hodnota funkce je uvozena příkazem `return`.

Doporučení jako u Matlabu, že název M-souboru a funkce uvedené na prvním řádku, by měl být shodný, v Euler Math Toolboxu vůbec neplatí.

Pro funkce jsou vyhrazena klíčová slova, jejíž užití se očekává právě v těle funkce.

- `args`, `arg1`, `arg2`, ..., `argn` – skupina příkazů, umožňující přístup k parametrům funkce (Matlab i GNU Octave mají příkaz `nargin`)
- `exec` – slouží k zavolání externího programu
- `forget`, `load` – zruší link na externí funkci („zapomene ji“), načte funkci do seznamu standardních a již načtených funkcí
- `global`, `useglobal` – pokud funkci píšeme v notebooku, těmito příkazy můžeme definovat přístup k proměnným v textovém okně
- `help`, `type` – vypíše nápovědu (pokud existuje) funkce, vypíše zdrojový kód funkce

II. PRAKTICKÁ ČÁST

11 PRÁCE S PROGRAMEM

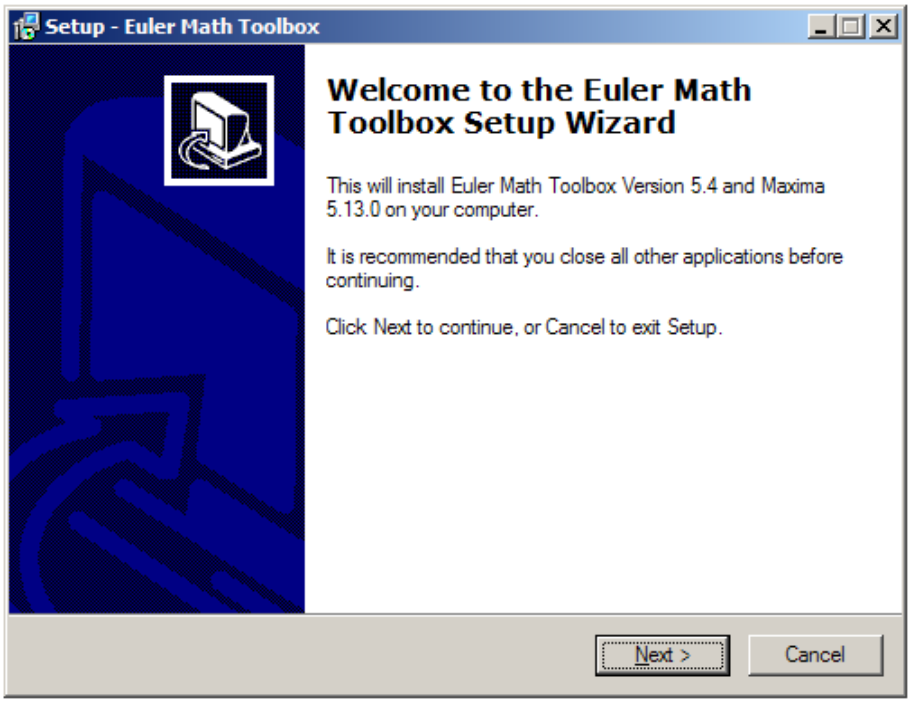
11.1 Stažení a instalace

Program EuMathT je možné stahovat v několika různých verzích pro různé operační systémy. Aktuální verze programu pro operační systém Microsoft Windows je 5.4. Tato bakalářská práce se zabývá právě touto verzí. Je možné ji stáhnout z oficiálních webový stránek autora Dr. Rene Grothmanna na adrese http://mathsrv.ku-eichstaett.de/MGF/homes/grothmann/euler/euler_download.html.

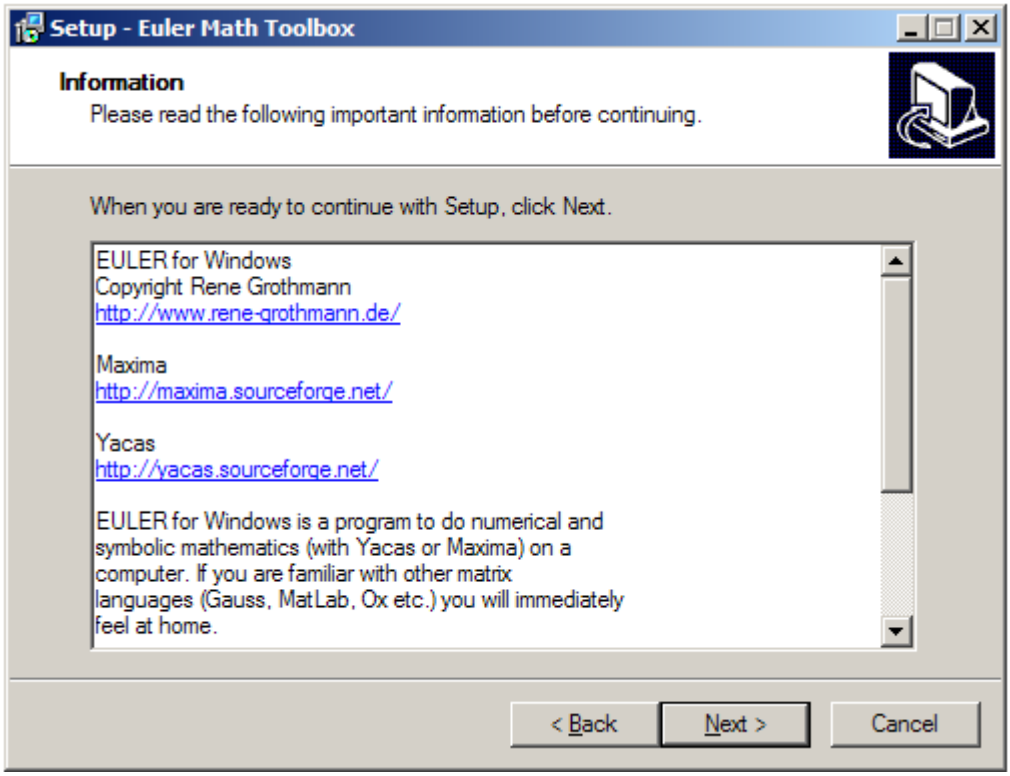
K dispozici je rovněž verze pro Linux/BSD/Unix ve verzi 1.61, kterou je možné nalézt na adrese http://euler.sourceforge.net/. Ta je vyvíjena Ericem Boucharém ve spolupráci s Puji Prasetiyem. Poslední z dostupných verzích je pro systém OS/2, nicméně již dále není vyvíjena.

Pokud by došlo k aktualizaci verze softwaru, není nutné celý program odinstalovat a opětovně instalovat. Stačí pouze stáhnout a instalovat aktualizační balíček v podobě souboru *EulerUpdate.exe* o velikosti přibližně 3 MB.

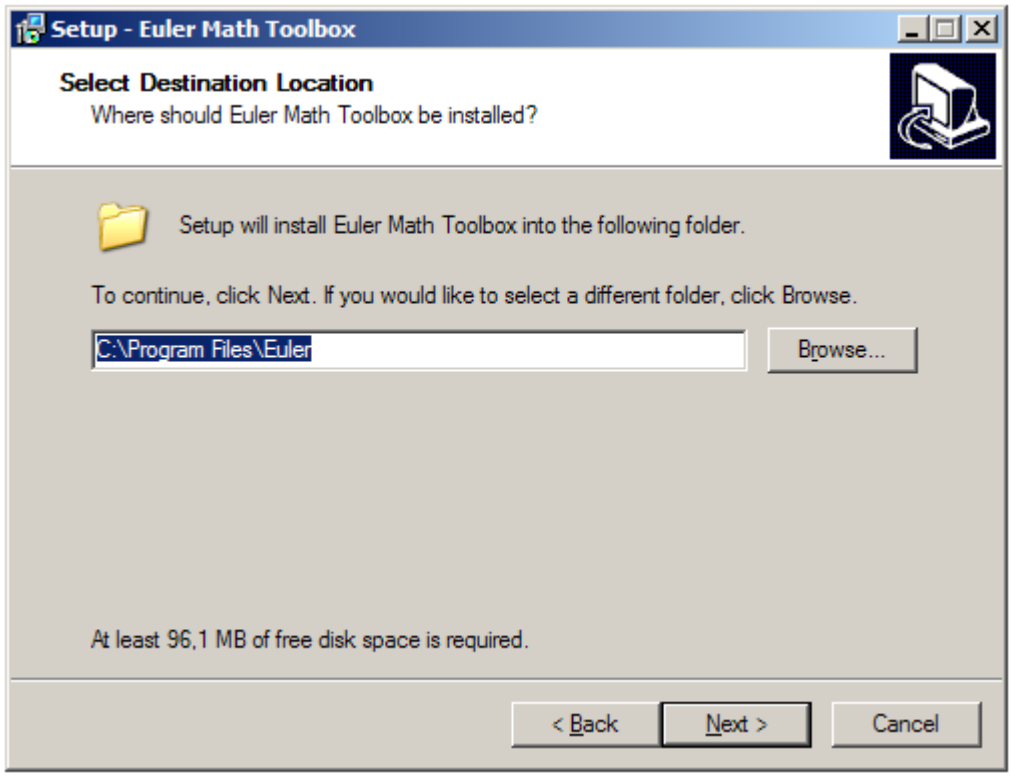
Stažený soubor *EulerSetup.exe* je samorozbalovací spustitelný instalační soubor o velikosti cca 35 MB. Po spuštění probíhá instalace.



Obrázek 2: Instalace Euler Math Toolboxu – úvod

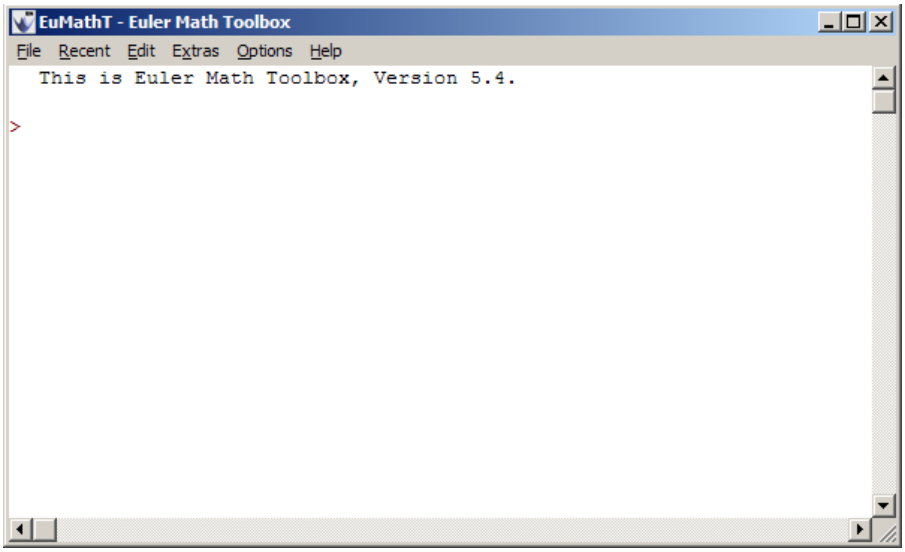


Obrázek 3: Instalace Euler Math Toolboxu – odsouhlasení licenční smlouvy



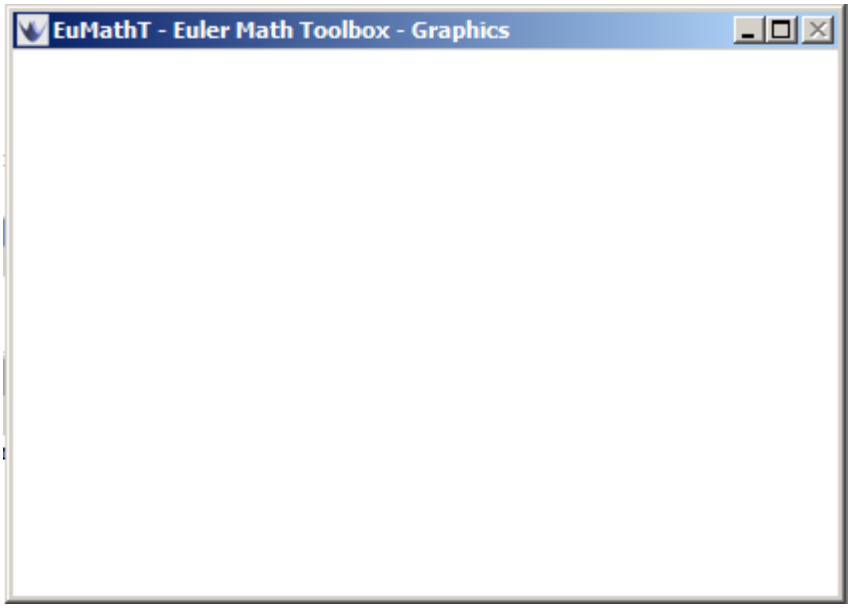
Obrázek 4: Instalace Euler Math Toolboxu – výběr umístění programu

Poté dialogovém okně znázorněném na Obrázku 6 následuje výběr jména složky ve Start menu. Po úspěšné instalaci je program možné ihned používat. Po spuštění máme k dispozici dvě okna – jedno *textové*:



Obrázek 5: Výchozí *textové* okno po spuštění Euler Math Toolbox

Druhé pak *grafické* pro vykreslení veškeré grafiky.



Obrázek 6: Výchozí *grafické* okno po spuštění Euler Math Toolbox

11.2 Nápověda k programu

Euler Math Toolbox nabízí širokou paletu nápovědy. První z nich je ve formě HTML stránek v umístění instalace v adresáři *docs*. Některé ze souborů v adresáři *docs* nabízí další odkazy, které vedou na HTML stránky obsahující demonstrační soubory dané kapitoly. Demo soubory je možné vyvolat syntaxí:

```
>demos functions
```

Místo slova functions se dosadí daná kapitola – jejich kompletní seznam je umístěn na http://mathsrv.ku-eichstaett.de/MGF/homes/grothmann/euler/demos/index.html.

Dále je možné zobrazit nápovědu jednotlivých příkazů:

```
>help function
```

Tento příkaz nabízí nápovědu pro funkce, které jsou v programu nadefinovány v souboru *euler.cfg*.

Další možností je studium již vytvořených funkcí a notebooků, které mají za úkol pomoci začínajícím uživatelům – ty jsou uloženy v adresářích *progs/applications* a *progs/user*. Zajímavostí je, že existuje i diskusní skupina na serveru Google, na adrese *http://groups.google.com/group/EulerHelp*. Účastníkem a velmi častým přispěvatelem diskuse je samotný autor softwaru EuMathT. Škálu nápověd uzavírá wiki stránka s adresou *http://compute.ku-eichstaett.de/MGF/wikis/euler/doku.php*, kde autor prezentuje novinky a nové funkce programu.

12 UKÁZKOVÉ PROGRAMY

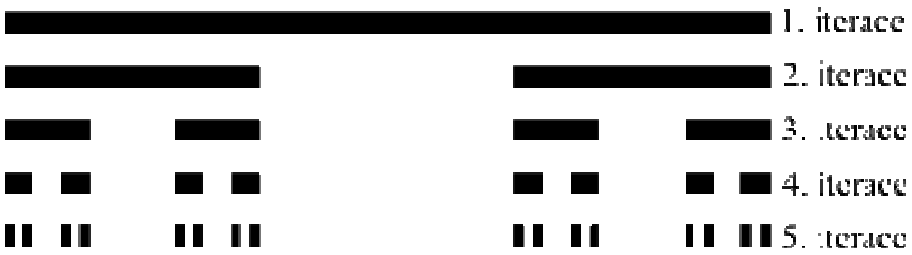
Stěžejní funkci této bakalářské práce tvoří sada několika samostatných programů, které pokrývají svým rozsahem probírané učivo předmětu *Programová podpora automatického řízení*. Všechny jsou uvedeny a okomentovány níže.

Dále byl vytvořen ke každé teoretické kapitole notebook, který nastiňuje probíranou problematiku. Každý z nich je proložen komentáři a vždy je součástí i kód, reprezentující identický výstup v Matlabu.

Všechny vypracované programy jsou uloženy na CD-ROM, které bylo součástí této práce.

12.1 Cantorovo diskontinuum

Cantorovo diskontinuum je matematický pojem označující jistou množinu bodů na přímce. Tato množina má některé velmi zvláštní vlastnosti. Cantorovo diskontinuum bývá také často považováno za fraktál.



Obrázek 7: Cantorovo diskontinuum pro pátou iteraci

Definice Cantorova diskontinua říká, že máme-li uzavřený interval [0,1] a odebereme-li z něj jeho prostřední třetinu (bez krajních bodů), získáme tím dva nové uzavřené intervaly třetinové délky. Pokud u obou těchto intervalů opět odebereme jejich prostřední třetiny, získáme celkem čtyři nové intervaly devítinové délky. Budeme-li takto pokračovat dál, tj. budeme-li odebírat v každém kroku vždy prostřední třetiny všech vzniklých intervalů, a provedeme-li těchto kroků nekonečně mnoho, získáme množinu bodů, které zůstanou neodebrány.

Program demonstruje některé funkce Euler Math Toolboxu popsané v teoretické části. Zejména pak *podmínky* (pro omezení zadání vstupních parametrů), *matice* (ukládání

jednotlivých souřadnic bodů úseček), *čtení a zápis těchto matic z a do souboru* a v neposlední řadě i *rekurzivní volání funkce*.

Pro demonstraci výpočtu Cantorova diskontinua stačí do textového souboru, který obsahuje funkci pro výpočet, zapsat příkaz:

```
cantor(0, 0, 100, 0, 3);
```

Vstupem je pět parametrů. Úsečka, definovaná dvěma body, každý o souřadnicích `x` a `y` (pro kód uvedený výše $x_1 = 0$, $y_1 = 0$, $x_2 = 100$, $y_2 = 0$), nad kterou se provádí veškeré operace a hloubka zanoření (počet cyklů dělení) – pro kód výše je zanoření rovno třem.

Výstupem není, jak by se dalo očekávat, grafický výstup, ale množina bodů. Tyto body, po spojení a vynesení na čtverečkový papír, vytvoří soustavu úseček, které by vizuálně připomínaly Cantorovo diskontinuum.

Soubor se zdrojovým kódem, dalšími informacemi a odkazy nese označení *cantorovo-diskontinuum.en*.

12.2 IsVector, IsMatrix, IsQuadratic

Tyto tři funkce slouží k testování, zda-li je vstupní proměnná vektor, matice či čtvercová matice. Každá z nich má právě jeden vstupní argument. Každou z funkcí je nezbytné volat alespoň s jedním parametrem, a pokud je zadáno parametrů více, funkce je ignoruje.

Funkce je možno nalézt v souboru `is_matrix_vector_quadratic.e`. Jejich obsah je uveden i zde včetně komentářů.

```
comment
Demonstracni program "Is_Matrix & Is_Vector & Is_Quadratic"
IsMatrix testuje, zda-li je zadany parametr funkce matice
IsVector testuje, zda-li je zadany parametr funkce vektor
IsQuadratic testuje, zda-li je zadany parametr funkce
ctvercovou matici
endcomment
```

Výše uvedená část kódu stručně popisuje, co vše program obsahuje a vykonává. Text uvedený mezi příkazy `comment` a `endcomment` se vypíše, pokud funkci načteme příkazem `load`.

Každá funkce musí být uvozena klíčovým slovem `function`, poté následuje jméno funkce a v závorkách seznam parametrů. Parametr není nutné uvozovat datovým typem – vstupem tak může být číslo, řetězec, interval, apod.

12.2.1 IsMatrix

Kód této funkce vypadá takto:

```
function ismatrix(matice)
##Funkce vraci 1, pokud je vstup matice, jinak 0
     nejmensi=min(size(matice));
     nejvetsi=max(size(matice));
     if((nejmensi>=1) && (nejvetsi>1)) return 1;
     else return 0;
     endif;
endfunction;
```

Test, zda-li je vstupem matice, probíhá tak, že za matici je považováno vše, co má rozměr `m×n`. Dále pak musí být splněna podmínka, že jeden z rozměrů je větší než jedna (druhý pak může nabývat hodnoty jedna). Lokální proměnná `nejmensi` obsahuje menší z rozměrů vstupní proměnné, `nejvetsi` pak větší z rozměrů proměnné `matice`. Pokud je `nejmensi` větší nebo rovna jedné a zároveň `nejvetsi` je větší než jedna, vstupní parametr funkce je skutečně maticí.

Pokud bychom chtěli funkci vytvořit v Matlabu či GNU Octave, postupovali bychom stejným způsobem.

12.2.2 IsVector

Funkce testuje, zda-li je vstupním parametrem vektor. Za vektor je považováno vše, co má datový typ označen číslem 2 nebo 3 (což odpovídá vektorům a maticím s reálnými či komplexními čísly) a zároveň má jeden z rozměrů roven jedné. Vstupní parametr funkce tedy může být sloupcový či řádkový vektor.

Oproti Matlabu byla zjištěna jedna odlišnost. Pokud zadáme např. příkaz `cislo=[1]`, Matlab proměnnou `cislo` bude chápat jako pole. Pro EuMathT jde však o číslo, ačkoliv by se na první pohled mohlo zdát, že jde o jednoprvkový vektor. Nicméně v obou případech funkce `isvector` vrátí pro tento případ výsledek nula (nepravda či false).

```
function isvector(v)
##Funkce vraci 1, pokud je vstup vektor, jinak 0
     if((typeof(v)!=2)&&(typeof(v)!=3)) return 0;endif;
     rozmer=min(size(v));
     if(rozmer==1) return 1;
     else return 0;
     endif;
endfunction;
```

12.2.3 IsQuadratic

Poslední funkce je `isquadratic`. Parametr funkce je testován, zda-li se jedná o čtvercovou matici. Funkce nejprve otestuje, zda-li vůbec je parametrem funkce matice, a poté do proměnných `nejmensi` a `nejvetsi` uloží rozměry matice. Pokud se obě čísla shodují, jde o čtvercovou matici a funkce vrací 1 (true či pravda). Pokud v jakémkoliv testu parametr nevyhovuje podmínkám, funkce vrátí 0 (false, tedy nepravda).

```
function isquadratic(vstup)
##Funkce vraci 1, pokud je vstup ctvercova matice, jinak 0
     if(!ismatrix(vstup)) return 0;endif;
     nejmensi=min(size(vstup));
```

```
    nejvetsi=max(size(vstup));
    if(nejmensi==nejvetsi) return 1;
    else return 0;
    endif;
endfunction;
```

12.3 IsEAN13

Zkratka EAN znamená European Article Number. Nejčastější EAN kód a pravděpodobně nejčastější čárový kód vůbec je EAN-13, který byl definován standardizační organizací GS1. Kódy EAN-13 jsou používány po celém světě k označování jednotlivých druhů zboží.

Tento program ukazuje užití cyklů `for` a `loop`, a též podmínek `if` ve funkci. Notebook obsahuje celkem dvě funkce – jejich kód je možné najít v souboru `is_ean.en`.

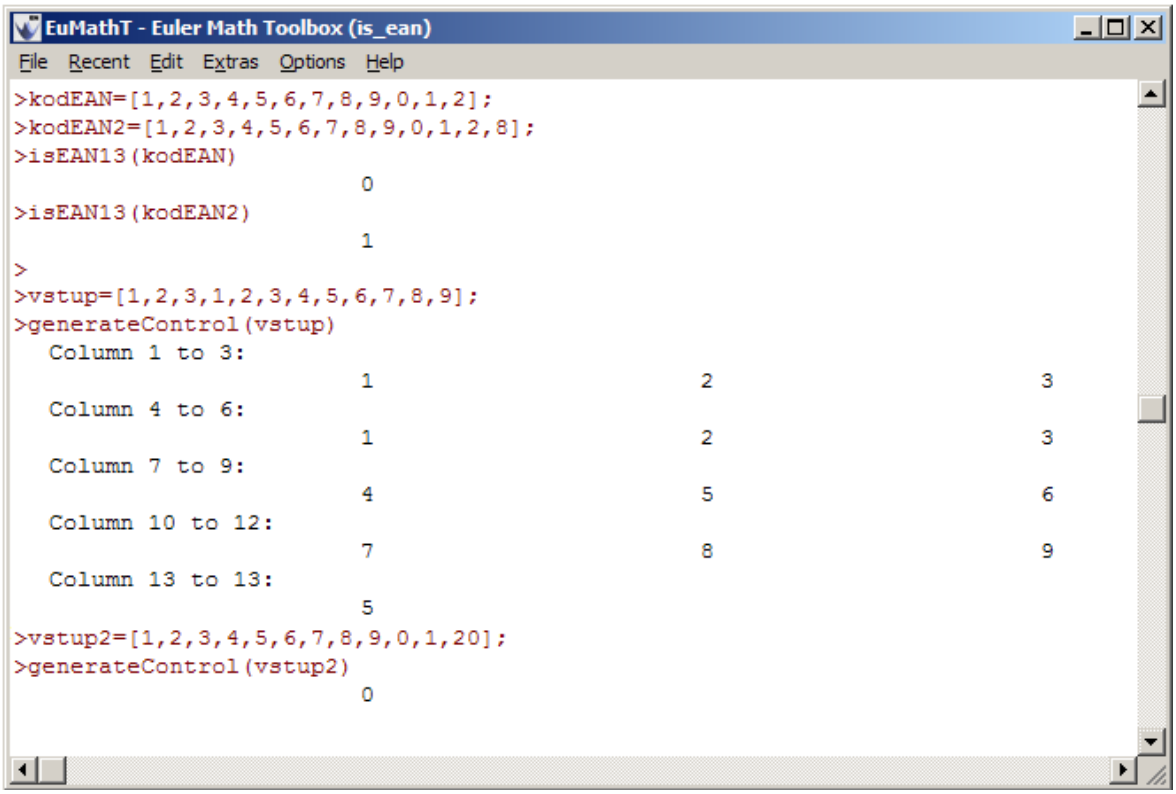
12.3.1 Funkce IsEAN13

Funkce `isEAN13` zjišťuje, zda-li je vstupní parametr platný kód EAN-13 (pokud ano, vrací 1, jinak 0). V úvodu těla funkce jsou uvedeny podmínky testující vstupní proměnnou. Vstupem tak může být pouze vektor o třinácti prvcích. Každý z prvků pak může nabývat hodnot z intervalu [0,9]. Test probíhá tak, že se sečtou všechny cifry na lichých pozicích, k nim se připočte trojnásobek součtu všech cifer na sudých pozicích. Pokud je součet dělitelný číslem deset, jde o kód EAN-13. Funkci by bylo možné dále rozšířit. Jednou z možností by byla analýza EAN-13 kódu – např. výpis země, ze které produkt označený EAN kódem pochází (k této identifikaci slouží první tři číslice).

12.3.2 Funkce generateControl

Funkce `generateControl` má za úkol vypočítat třináctou kontrolní číslici pro EAN-13 kód. Funkce na začátku opět testuje vstupní parametr – musí se jednat o vektor dvanácti prvků a prvky musí být čísla z intervalu [0,9]. Algoritmus je identický jako u funkce isEAN13, s tím rozdílem, že třináctá kontrolní cifra je hledána. Musí to být takové číslo, které po přičtení k celkovém součtu bude dělitelné číslicí deset beze zbytku.

Níže je vložen obrázek, který názorně ilustruje, co funkce vykonávají.



Obrázek 8: Ilustrační příklad testování EAN-13 kódu

12.4 Převod barevných prostorů (RGB na HSL)

Tento program demonstruje několikanásobné větvení příkazy `if`, `elseif` a `else`.

RGB barevný model je aditivní barevný model, ve kterém je smícháno společně červené, zelené a modré světlo různými cestami k reprodukci obsáhlého pole barev. Název modelu pochází z počátečních písmen tří aditivních primárních barev – červené, zelené a modré. RGB model sám o sobě nedefinuje co je míněno červenou, modrou, zelenou kolorometricky a tak výsledek smíchaní složek není přesný, ale relativní. Model RGB je možné zobrazit jako jednotkovou krychli, ve které každá z kolmých hran udává škálu mohutností barevných složek. Potom libovolný bod se souřadnicemi (r,g,b) v této krychli udává hodnotu výsledné barvy. Hodnoty `r`, `g`, `b` jsou reálná čísla mezi 0 a 1.

HSL model je matematicky definován jako transformace souřadnic z RGB prostoru. Popisuje barvy jako body ve válci jehož centrální osa sahá od černé až dolů k bílé a nahoru k neutrální barvě mezi nimi, kde úhel kolem osy odpovídá „odstínu“ (`Hue`), vzdálenost od osy odpovídá „saturaci“ (`Saturation`), a vzdálenost podél osy odpovídá „světlosti“,

„hodnotě“ nebo „jasu“ (`Lightness`). Hodnoty modelu HSL představují tři proměnné – `h` je úhel [0, 360 °] a `s`, `l` jsou reálná čísla mezi 0 a 1, symbolizující saturaci a světlost.

Funkce pro jednosměrný převod z RGB prostoru do HSL je uložena v souboru *rgb2hsl.en*.

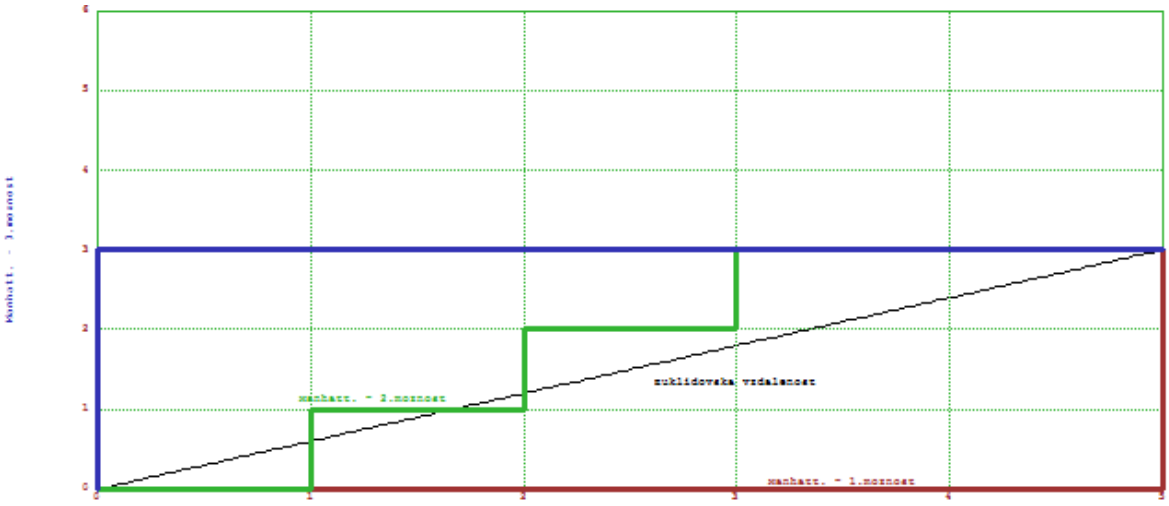
Vstupem funkce jsou tři čísla z intervalu [0, 255] reprezentující složky RGB modelu. Výstup pak tvoří tříprvkový vektor, obsahující složky HSL barevného modelu.

12.5 Manhattanská vzdálenost

Manhattanská vzdálenost je jedna z metrik, jimiž lze definovat vzdálenost. Tato konkrétní vzdálenost je definována jako vzdálenost, kterou je třeba ujít mezi dvěma křižovatkami na Manhattanu, mezi kterými se lze pohybovat jen po na sebe kolmých ulicích ve směru obou os).

Program si klade za cíl demonstrovat vykreslování několika 2D grafů (různých barev, popisků) v jednom. Výstupem je tak několik různých křivek, které mají stejný výchozí i cílový bod a rovněž stejnou délku. Do výsledného grafu je též přidána křivka reprezentující Euklidovskou metriku, která ze všech vykreslených má nejkratší vzdálenost.

Program také obsahuje funkci, která po zadání dvojice bodů o souřadnicích `x` a `y` spočte jejich vzájemnou Manhattanskou vzdálenost.



Obrázek 9: Grafický výstup programu Manhattanská vzdálenost

12.6 Euklidův algoritmus

Euklidův algoritmus je pojmenován podle starověkého filozofa Euklida, který jej uvedl ve svém díle Základy (cca 300 př.n.l.), přestože tento postup zřejmě sám nevynalezl. Jedná se pravděpodobně o nejstarší lidstvu známý netriviální algoritmus.

Euklidův algoritmus je postup, kterým lze určit největšího společného dělitele dvou přirozených čísel, tzn. nejvyšší číslo takové, že beze zbytku dělí obě čísla.

Princip je takový, že jsou dána dvě přiřozená čísla, uložená v proměnných `u` a `v`. Dokud proměnná `v` není nulová, opakuj:

1. Do pomocné proměnné `r` ulož zbytek po dělení čísla `u` číslem `v`
2. Do `u` ulož `v`
3. Do `v` ulož `r`

Algoritmus vrací největšího společného dělitele původních čísel v proměnné `u`. Praktický příklad pro čísla 120 a 23 by vypadal takto:

```
120  ÷   23   = 5 (zb. 5)
23   ÷   5    = 4 (zb. 3)
5    ÷   3    = 1 (zb. 2)
3    ÷   2    = 1 (zb. 1)
2    ÷   1    = 2 (zb. 0)
```

Proměnná `v` nyní obsahuje hodnotu 0, algoritmus končí. Zároveň v proměnné `u` je uložena hodnota 1, která je právě největším společným dělitelem čísel 120 a 23.

Euklidův algoritmus má praktické uplatnění např. v kryptografii u RSA kódů nebo u polynomiálních metod návrhu regulátorů. Aplikací zobecněného Euklidova algoritmu je možné nalézt partikulární řešení Diofantických rovnic.

Celý program je uložen v souboru `eukliduv_algoritmus.en`, který obsahuje dvě funkce. Obě z nich bylo možné naprogramovat několika způsoby – buď cyklem s omezující podmínkou nebo rekurzivním voláním funkce.

Funkce `euklides` je rekurzivní funkce, která vrací největšího společného dělitele dvou čísel. Kód vypadá následovně.

```
function euklides(m,n)
$if((typeof(m)!=0) || (typeof(n)!=0)) return "";endif;
$m=floor(m); n=floor(n);
$if(n==0) return m;endif;
$if((m<0) || (n<0)) return "";
$else return euklides(n,mod(m,n));
$endif;
$endfunction;
```

Funkci je nutné předat dva parametry (první řádek). Ty jsou pak testovány – musí jít o reálná čísla (druhý řádek). Následně jsou převedeny na celá čísla (třetí řádek) a testovány, zda jsou kladná (pátý řádek). Pokud ano, volá se funkce znovu s novými parametry (šestý řádek) tak dlouho, dokud není proměnná n nulová (čtvrtý řádek). Pokud ne, funkce vrací prázdnou množinu.

Kód v Matlabu i GNU Octave by vypadal velmi podobně. Rozdíl v kódu by byl takový, že:

- Místo testu datového typu (druhý řádek) by se použila např. kombinace funkcí `isreal` a `length`. První testuje, zda-li vstup obsahuje pouze čísla, druhý pak vrací délku vektoru. Pokud `isreal` vrátí hodnotu jedna a `length` je též jedna, můžeme počítat.

Funkce `rozsirenyEuklides` využívá smyčky `repeat` s omezující podmínkou. Kromě největšího společného dělitele dvou čísel vrací i koeficienty `x` a `y` takové, že platí `NSD(m,n) = ax + by`. Kód vypadá následovně:

```
function rozsirenyEuklides(m,n)
$if((typeof(m)!=0) || (typeof(n)!=0)) return "";endif;
$m=floor(m); n=floor(n);
$a1=1; a2=0; a3=m;
```

```
$b1=0; b2=1; b3=n;
$repeat
$if(b3==0) break;endif;
$podil=floor(a3/b3);
$nahrada1=a1-podil*b1; $nahrada2=a2-podil*b2;
$nahrada3=a3-podil*b3;
$a1=b1; a2=b2; a3=b3;
$b1=nahrada1; $b2=nahrada2; $b3=nahrada3;
$end;
$result=a1|a2|a3;
$return result;
$endfunction;
```

Stějně jako v předchozí funkci, i zde je nutné předat dva parametry (první řádek). Ty jsou následně testovány (druhý a třetí ř.) identicky jako ve funkci `euklides`.

Poté jsou definovány prvky `a1`, `a2`, `a3`, `b1`, `b2`, `b3`, které reprezentují prvky matice (čtvrtý a pátý ř.). Následuje tělo cyklu `repeat` (šestý ř.), který se bude provádět, dokud nebude prvek `b3` roven nule (sedmý ř.).

Poté probíhají elementární úpravy prvků (osmý až desátý ř.). První řádek nyní obsahuje hodnoty druhého řádku. Druhý řádek byl pak získán odečtením násobku druhého řádku od prvního řádku (vše symbolizuje jedenáctý a dvanáctý řádek kódu). Dalším krokem je ukončení cyklu, předání výstupní proměnné `result` a ukončení funkce.

12.7 Komplexní program „Samostatná práce“

Tento skript obsahově pokrývá většinu kapitol probraných v teoretické části. Zadání bylo vytvořeno vyučujícími předmětu *Programová podpora automatického řízení*. Vypracovaný notebook i zadání je možné najít v souborech *komplexni-reseni-euler.en* a *komplexni-reseni-zadani.pdf*. Spolu s programem je možné samostatně prohlížet exportované grafické výstupy (soubory komplexni-*reseni-euler-001.png*, *komplexni-reseni-euler-002.png* až *komplexni-reseni-euler-008.png*.)

V programu již není obsažen kód Matlabu, který by zajišťoval identický výsledek. K pochopení všech funkcí notebooku a syntaktických odlišností oproti Matlabu a GNU Octave by měly postačovat notebooky teoretických kapitol (nesoucí označení *program-kapitola-číslo_kapitoly.en*).

12.7.1 Vytváření vektorů a matic

První část notebooku se zabývá vytvářením vektorů a matic. Zmíněny jsou pouze funkce pro generování speciálních matic. Kód této části:

```
A=id(30,30);

B=ones(20,20);

C=zeros(15,7);

d=ones(1,15);

e=zeros(18,1);

F=random(7,8);

g=random(1,35);

posloupnost=[1:25];

H=dup(posloupnost,25);

I=H(10:20;10:15);
```

12.7.2 Práce s vektory a maticemi

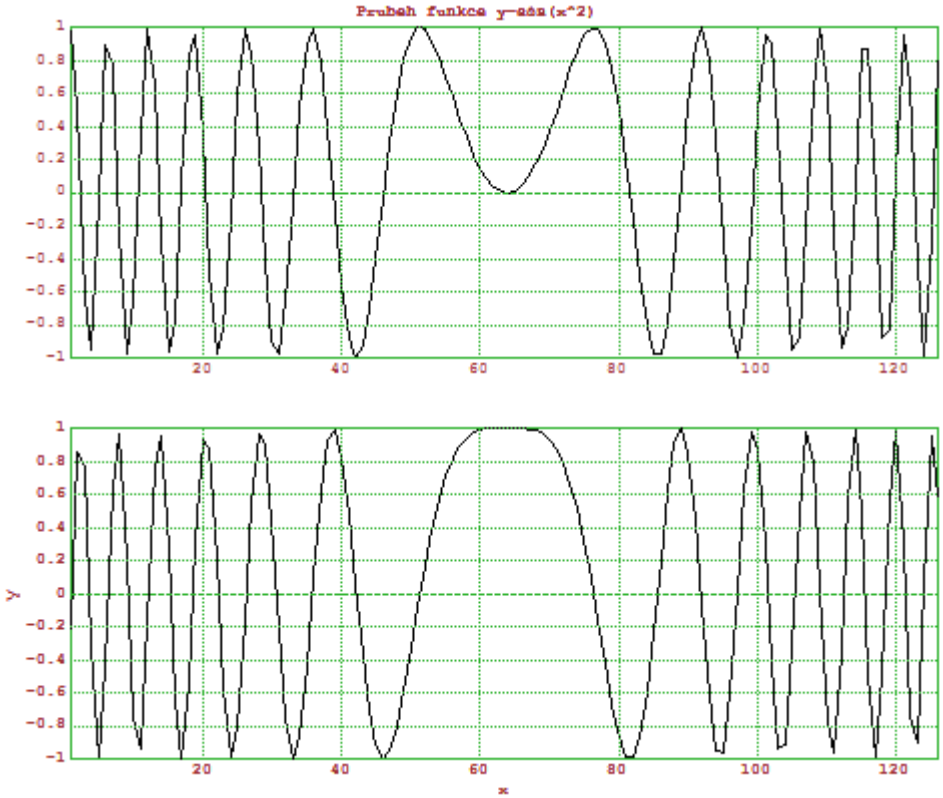
Tato kapitola navazuje na předchozí. K demonstraci práce s vektory a maticemi se užívá proměnných deklarovaných v kapitola 12.7.1, Vytváření vektorů a matic. Kód této části vypadá následovně:

```
C.C';

A+H;

g^(1/3);

soustava=[5,1,7;27,3,1;1,5,9]

reseni=[10;20;30]

soustava\reseni
```

```
G=[5,1,7;2,4,4;1,1,10];

inv(G);

det(G);

sum(diag(G,0));

eigen(G);

sum(1:1000);
```

12.7.3 Práce s 2D grafy

Vykreslení grafů polynomů a goniometrických funkcí, omezení x-ových hodnot, popisky os, nadpisy grafů, změna stylu grafů, to vše bylo demonstrováno v této kapitole. Sekce dále obsahuje praktickou ukázku, jak rozdělit grafické okno a vykreslit více grafů do jednoho.



Obrázek 10: Goniometrické funkce vykreslené v grafickém okně

12.7.4 Práce s 3D grafy

V této části je vysvětleno, jak vytvářet 3D grafy pomocí funkcí `framedsolid` a `mesh`. Z příkladů by mělo být patrné, co je před samotným vykreslením nezbytné zadefinovat. Například pro funkci mesh:

```
x=-10:0.5:10; y=x';
Z=((2*x^2-y^2)*(y^2-x^2));
clg; mesh(Z);
```

Prvním řádkem je definován vektor hodnot `x` a také hodnot `y`, který vznikl transponováním vektoru `x`. Na druhém řádku je definován vektor `z` pomocí proměnných `x` a `y`. Poslední řádek nejprve vyčistí grafické okno (příkaz `clg`) a poté vykreslí požadovaný 3D graf.

12.7.5 Polynomy

Téma polynomů nebylo popsáno v teoretické části. Kód této části vypadá následovně:

```
p=[2 3 0 2]; q=[1 4 0 7 1]
polysolve(p)
polymult(p,q)
g=[-1,-2,-3];
flipx(polycons(g))
polyval(p,-10:1:10)
```

Na prvním řádku jsou deklarovány polynomy jako vektory čísel. Vše totožné s Matlabem i GNU Octave.

Druhý řádek obsahuje funkci pro získání kořenů polynomu (Matlab má funkci `roots`). Třetí demonstruje násobení polynomů (Matlab užívá funkci `conv`). Pátý řádek ukazuje, jakým způsobem se vytváří polynom – operace `flipx` tu zajistí otočení vektoru tak, že výsledek je identický s Matlabem a GNU Octave. Tomu musí předcházet vytvoření vektoru

kořenů (čtvrtý řádek). Poslední řádek provádí tabelaci polynomu na intervalu [-10,10] s krokem 1.

12.8 Röslerův atraktor

Röslerův atraktor je umělý systém generující podivný atraktor, založený na nejjednodušších principech generace chaosu, a to transformaci "rozprostření a ohyb" (stretch and fold). Atraktor nese název po svém tvůrci, Ottu Rösslerovi, který jej v roce 1976 popsal. Vychází z Lorenzova atraktoru, který byl publikován o třináct let dříve. Rösslerovo jméno začalo být spojováno s jednoduchým atraktorem ve tvaru stuhy, svinuté do věnečku se záhybem. Rösler si ale také představoval atraktory z vyšších dimenzí jako ohýbání a stlačování stavového prostoru. které se skutečně stalo klíčem ke konstrukci podivných atraktorů.

Rösler zastával názor, že tyto tvary představují princip samoorganizace v přírodě (vznik disipativních struktur díky složitým a vzájemně souvisejícím procesům). Představoval si punčochu k měření větru, do které se chytil vítr. Vítr je v pasti a musí "proti své vůli" konat něco užitečného. Podle něj princip samoorganizace spočíval vtom, že příroda dělá něco proti své vůli a díky tomu se zaplétá do sebe a dává tak vzniknout kráse. Ať už však byly jeho filozofické úvahy jakékoliv, neodmyslitelně patří do historie fraktálů a chaosu.

Röslerův atraktor je popsán soustavou tří nelineárních diferenciálních rovnic

$$\frac{dx}{dt} = -y - z$$

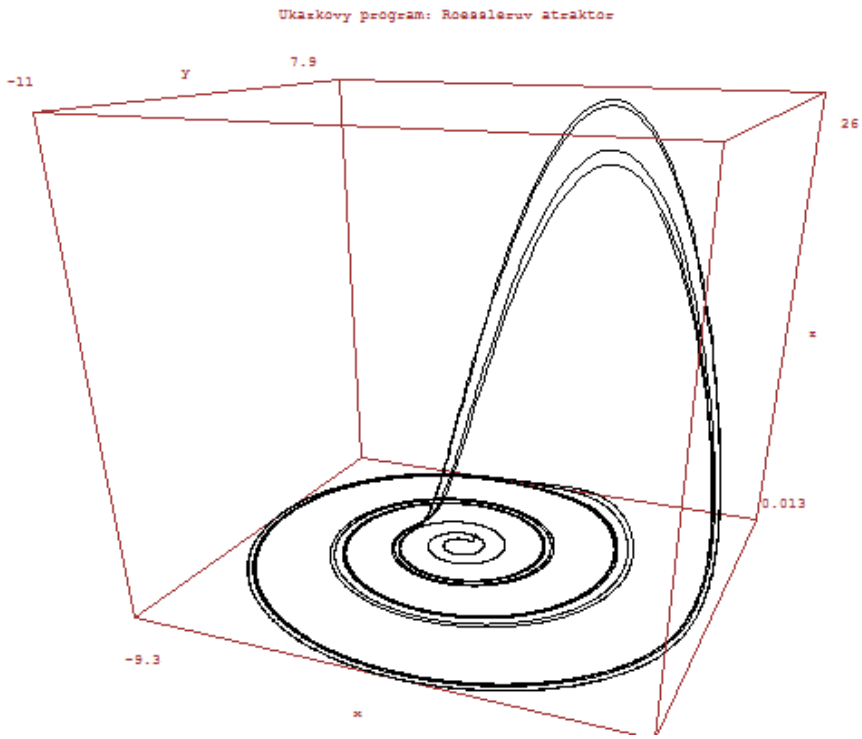
$$\frac{dy}{dt} = x + ay$$

$$\frac{dz}{dt} = b + z(x - c)$$

kde a = 0,2; b = 0,2; c = 5,7.

Zdrojový kód programu je uložen v souboru *rosleruv-atraktor.en*. Celý skript obsahuje funkci `atraktor`, která vyžaduje jeden vstupní parametr – počet iterací. Tělo funkce se skládá ze soustavy výše zmíněných diferenciálních rovnic a jejich vizualizace jako 3D

grafu. Po vykreslení v grafickém okně je možné tento atraktor přibližovat/oddalovat pomocí tlačítek +/− a posunovat pomocí šipek.



Obrázek č. 11: Grafický výstup programu Röslerův atraktor

12.9 Lineární regrese (metoda nejmenších čtverců)

Lineární regrese představuje aproximaci daných hodnot polynomem prvního řádu (přímkou) metodou nejmenších čtverců. Jinak řečeno, jedná se o proložení několika bodů v grafu takovou přímkou, aby součet druhých mocnin odchylek jednotlivých bodů od přímky byl minimální.

Zdrojový kód je umístěn v souboru *linearni-regrese.en*.

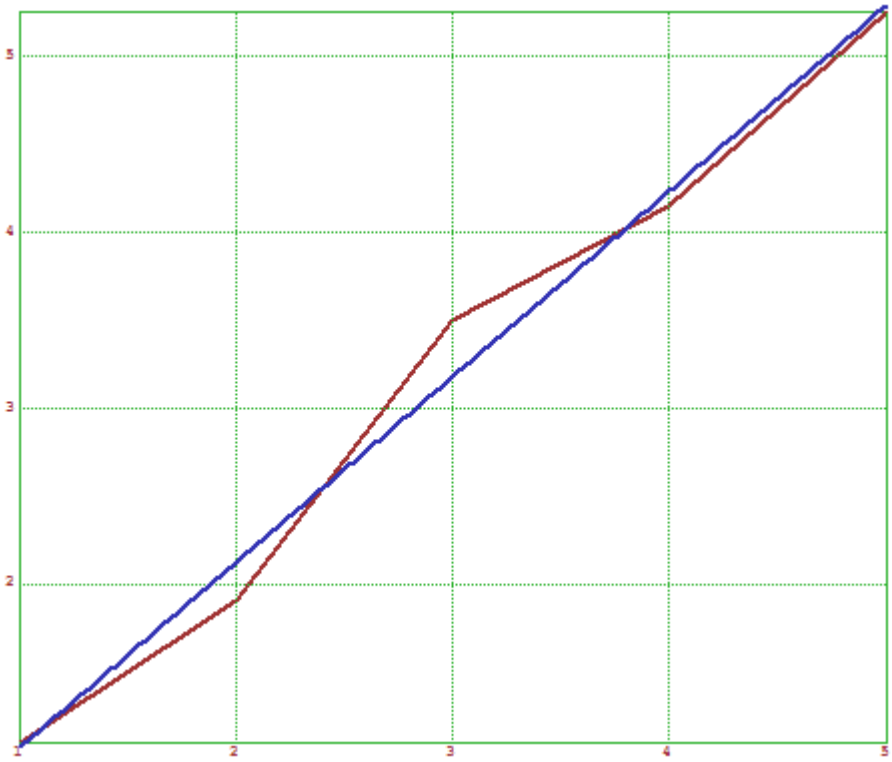
Obsah souboru tvoří funkce `polyfitONE`, se dvěma vstupními parametry. Parametry musí být vektory reálných čísel o stejném počtu prvků. Poté jsou metodou nejmenších čtverců pomocí dvou vzorců

$$a = \frac{n\sum x_i y_i - \sum x_i \sum y_i}{n\sum x_i^2 - \left(\sum x_i\right)^2}$$

$$b = \frac{\sum x_i^2 \sum y_i - \sum x_i \sum x_i y_i}{n \sum x_i^2 - \left(\sum x_i\right)^2}$$

vypočítány parametry, které určují aproximační přímku. Výsledná přímka je ve tvaru `y = a × x + b`. Tyto parametry `a`, `b` jsou zároveň výstupem funkce `polyfitONE`.

Parametry je pak možné využít ke konstrukci přímky lineární regrese pro data.



Obrázek 12: Lineární regrese

ZÁVĚR

Hlavním cílem této práce bylo ukázat možnosti programu Euler Math Toolbox, který se řadí do kategorie k softwarům GNU Octave, Matlab, Scilab, Freemat a dalších.

Snahou bylo popsat rozdíly při práci mezi Matlabem, GNU Octave a Euler Math Toolboxem. Zejména byl kladen důraz na syntaxi, rozdílné názvy a funkce jednotlivých příkazů. Popisované odlišnosti byly demonstrovány na praktických příkladech přiložených na CD-ROM.

Při porovnání tří dříve zmíněných software by bylo možné konstatovat, že nejkomplexnějším a nejrobustnějším z nich je Matlab a to zejména díky jeho velkému množství toolboxů. To, že je "nej" je však kompenzováno tím, že je běžným uživatelům, kteří se chtějí se softwarem seznámit a provádět základní výpočty, nepřístupný. Důvodem je fakt, že je vyvíjen jako komerční produkt a zakoupení licence je poměrně nákladné (v řádech desetitisíců). Díky tomu se tak pro velkou spoustu potenciálních zájemců stává nedostupným a je možné se s ním setkat pouze na středních odborných školách, technických vysokých školách či v zaměstnání.

Jestliže by byla v úvahu vzata cena softwaru, zvítězil by zřejmě GNU Octave, který se s Matlabem snaží zachovávat kompatibilitu a nabízí velký počet jeho funkcí, včetně dodatečných modulů (toolboxů). V minulosti měl GNU Octave jistý handikep – jelikož byl vyvíjen pro svobodný operační systém GNU/Linux, pod systémem Microsoft Windows program "padal" či rozdílně interpretoval příkazy, viz literatura [9]. Nicméně to dnes již neplatí a tudíž tento software je možné považovat za jeden z nejlepších, co je k dispozici.

Ačkoliv GNU Octave existuje již delší dobu (od r. 1988), byly programátory vytvořeny další výpočetní nástroje, mezi které řadíme i Euler Math Toolbox. Vše je otázka možností a volby – snahou této práce bylo přiblížit právě tento software a nalákat potenciální zájemce, zejména z řad studentů, k tomu, aby zvolili pro práci a výuku právě jej.

ZÁVĚR V ANGLIČTINĚ

The main goal of the bachelor thesis was to show potential of software Euler Math Toolbox.

The endeavour was to compare EuMathT with Matlab and GNU Octave. It was especially aimed at the differences in syntax, different names and functionality of commands. All discussed dissimilarities were demonstrated on programmes, that are included on CD-ROM.

One could claim that Matlab is the most complex and robust software from the three mentioned above. Reasons for that are large number of add-ons (toolboxes) it implements and lots of professional users in industry and academia (according to [17], it's over one million people). However Matlab remains inaccessible to common users due to it's high price – Matlab is commercial software. That limits its use only to secondary vocational schools, universities and employment.

If the price is taken into account, the winning software'll be GNU Octave. The reason is that it's free computer program, part of the GNU project. It also maintains compatibility with Matlab and includes a lot of Matlab functions and add-ons (toolboxes). GNU Octave used to have a drawback, which caused lesser spread out than Matlab – the software was developed under GNU/Linux and its version for Microsoft Windows sometimes suddenly shut the program down or misinterpreted commands, see Internet source [9]. Nevertheless the latest version of GNU Octave for Microsoft Windows works just fine and can be counted as the best computing software available.

Although GNU Octave was created in 1988, there were programmers who worked on another numerical computing environment. One of those is Euler Math Toolbox. Principal aim of the bachelor thesis was to grip potential users concerned, especially students to use this software.

SEZNAM POUŽITÉ LITERATURY

Monografie:

[1] PERŮTKA, K. (2005): MATLAB – Základy pro studenty automatizace a informačních technologií. Skriptum, 1. vydání. Zlín: Univerzita Tomáše Bati ve Zlíně, Zlín. ISBN 80-7318-355-2

[2] KREIZLOVÁ, Z. (2007): Euler – průvodce v češtině. Bakalářská práce, Univerzita Tomáše Bati ve Zlíně, Fakulta aplikované informatiky

[3] JUST, M. (2006): Český průvodce programem Octave. Bakalářská práce, Univerzita Tomáše Bati ve Zlíně, Fakulta aplikované informatiky

Internetové zdroje:

[4] Český průvodce programem Euler [online]. [cit. 2008-04-21]. Dostupný z WWW: <http://euler.euweb.cz/>

[5] Euler Dokumentation [online]. [cit. 2008-04-21]. Dostupný z WWW: <http://mathsrv.ku-eichstaett.de/MGF/homes/grothmann/euler/index.html>

[6] The GNU General Public Licence [online]. Free Software Foundation.

[cit. 2008-04-20]. Dostupný z WWW: http://www.gnu.org/licenses/gpl.html

[7] The Yacas computer algebra system [online]. PINKUS, A. [cit 2008-04-20].

Dostupný z WWW: <http://yacas.sourceforge.net/homepage.html>

[8] Maxima, a Computer Algebra System [online]. Maxima developement team. [cit. 2008-04-20]. Dostupný z WWW: <http://maxima.sourceforge.net/>

[9] OctaveForWindows [online]. Eaton, W. John. [cit. 2008-05-06]. Dostupný z WWW: http://wiki.octave.org/wiki.pl?OctaveForWindows

[10] Cantorovo diskontinuum [online]. Wikimedia Foundation. [cit. 2008-05-12]. Dostupný z WWW: http://cs.wikipedia.org/wiki/Cantorovo_diskontinuum

[11] Čárový kód [online]. Wikimedia Foundation. [cit. 2008-05-12]. Dostupný z WWW: http://cs.wikipedia.org/wiki/Čárový_kód#K.C3.B3dy_typu_EAN

[12] RGB color model [online]. Wikimedia Foundation. [cit. 2008-05-12]. Dostupný z WWW: http://en.wikipedia.org/wiki/RGB_color_model

[13] HSL and HSV [online]. Wikimedia Foundation. [cit. 2008-05-12]. Dostupný z WWW: http://en.wikipedia.org/wiki/HSL_and_HSV

[14] Euclidean algorithm [online]. Wikimedia Foundation. [cit. 2008-05-12]. Dostupný z WWW: http://en.wikipedia.org/wiki/Euclid_algorithm

[15] Deterministický chaos: Princip a aplikace [online]. Hladík, M. [cit. 2008-05-12]. Dostupný z WWW: http://hungry-lord.wz.cz/data/Rosler.php

[16] Metoda nejmenších čtverců [online]. Wikimedia Foundation. [cit. 2008-05-12]. Dostupný z WWW: http://cs.wikipedia.org/wiki/Metoda_nejmenších_čtverců

[17] MATLAB [online]. Wikimedia Foundation. [cit. 2008-05-13]. Dostupný z WWW: http://en.wikipedia.org/wiki/MATLAB

SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK

ASCII	American Standard Code for Information Interchange
AVI	Audio Video Interleave
CD-ROM	Compact Disc-Read Only Memory
CSV	Comma-Separated Values
EPS	Encapsulated PostScript
EAN	European Article Number
GIF	Graphics Interchange Format
GNU	GNU's Not Unix
GNU GPL	GNU General Public Licence
HSL	Hue Saturation Lightness
HTML	HyperText Markup Language
PNG	Portable Network Graphics nebo také PNG's Not Gif
RGB	Red Green Blue
XML	eXtensible Markup Language
XLS	eXceL Spreadssheet

SEZNAM OBRÁZKŮ

SEZNAM TABULEK

SEZNAM PŘÍLOH

PŘÍLOHA P I: CD-ROM (OBSAHUJE PROGRAMY V SOFTWARU EULER MATH TOOLBOX)