

Vysoká škola báňská – Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra aplikované matematiky

APLIKAČNÍ ÚLOHY PRO RADOST
P. Beremlijski, M. Sadowská

Matematika s radostí

Petr Beremlijski, Marie Sadowská
Aplikační úlohy pro radost

© Petr Beremlijski, Marie Sadowská, 2015

Úvod

Tato kniha je určena pro každého, koho zajímají aplikace matematiky a kdo se rád matematikou baví. Kniha čtenáře provede řadou matematických aplikací. Jejím obsahem je také řešení těchto aplikací. Text je určen zejména pro studenty posledních ročníků středních škol, ale může být zajímavou inspirací i pro jejich učitele a další zájemce o užitečné a zajímavé aplikace matematiky.

Tento text vznikl v letech 2013–2015 a je úzce spjatý s úlohami, které řeší studenti v průběhu semináře Škola matematického modelování. Tento seminář, pro který používáme zkratku ŠKOMAM, organizuje Katedra aplikované matematiky Fakulty elektrotechniky a informatiky Vysoké školy báňské – Technické univerzity Ostrava jednou ročně již od roku 2005. Podrobnosti o semináři naleznete na webové adrese <http://skomam.vsb.cz>. Text byl napsán díky finanční podpoře projektu Matematika s radostí. Díky realizaci tohoto projektu vznikla řada interaktivních testů, párovacích her, her Neriskuj, AZ kvízů a Poznej obrázek. Všechny tyto materiály jsou na webových stránkách projektu <http://msr.vsb.cz>.

Naše kniha je rozdělena do šesti kapitol. V první kapitole se věnujeme úlohám z matematické analýzy. Na úvod této kapitoly se seznámíme s některými základními pojmy a ty pak využijeme například pro přibližné určení čísla π , výpočet přibližného obsahu zadané plochy nebo nalezení přibližného řešení zadané rovnice.

Druhá kapitola je zaměřena na aplikace lineární algebry. Po úvodu do této oblasti si ukážeme, jak můžeme využít základní algebraické struktury jako jsou vektory a matice pro vyhledávání v databázích pomocí klíčových slov. Další aplikací, kterou se v této kapitole zabýváme, je realizace geometrických zobrazení (stejnolehlost, rotace, posunutí) pomocí nástrojů lineární algebry. Tímto způsobem můžeme geometrická zobrazení efektivně implementovat do počítačové grafiky.

V třetí kapitole se seznámíme se speciálním typem rovnic, tzv. diferenciálními rovnicemi. A pomocí nich si zavedeme tzv. počáteční úlohy. V úvodu této kapitoly si navíc ukážeme, jak tyto úlohy řešit. Počáteční úlohy mají široké uplatnění a popisuje se pomocí nich řada úloh z oblasti biologie, fyziky

či společenských věd. My si ukážeme, jak popsat například koncentraci léku v krvi pacienta v závislosti na čase, jak modelovat pohyb kyvadla, či jak odhadovat vývoj populace daného státu.

Ve čtvrté kapitole se opět potkáme s diferenciálními rovnicemi, ale budeme zde řešit tzv. okrajové úlohy. Pomocí takového typu úloh si můžeme modelovat průhyb struny pevně upevněné na obou jejích koncích, na kterou působí zadaná síla.

V páté kapitole se podíváme na optimalizační úlohy a ukážeme si, jakými metodami lze tyto úlohy řešit. Pomocí tohoto aparátu si pak popíšeme kuličku dané hmotnosti zavěšenou na pružině. Řešením optimalizační úlohy získáme polohu, ve které se kulička ustálí.

Nejrozsáhlejší šestá kapitola obsahuje podrobná a komentovaná řešení úloh popsaných v prvních pěti kapitolách. Součástí řešení jsou i kódy pro počítačové řešení těchto úloh. Pro počítačové řešení jsme zvolili komerční systém Matlab, proto jsou všechny kódy v této kapitole zapsány s použitím syntaxe tohoto systému. Matlab se skládá z několika součástí. Jde zejména o programovací jazyk zaměřený na numerické výpočty a vývoj numerických algoritmů. Matlab obsahuje také knihovnu funkcí pro řešení řady numerických úloh. A důležitou složkou Matlabu je grafické prostředí pro interaktivní zadávání příkazů. Podrobný popis tohoto jazyka je k dispozici v [7]. Pokud čtenář naší knihy nemá Matlab k dispozici, může použít systém Octave, který se značně podobá Matlabu a je volně k dispozici na webové adrese <https://www.gnu.org/software/octave>. Řešení jsou navíc doplněna tam, kde je to vhodné, o analýzu chyby aproximace řešení.

Na závěr je kniha doplněna několika přílohami.

Ostrava,
únor 2015

Petr Beremlijski
Marie Sadovská

Text byl vytvořen v rámci projektu Matematika s radostí (<http://msr.vsb.cz>).



Poděkování

Rádi bychom na tomto místě poděkovali našim kolegům z Katedry aplikované matematiky Fakulty elektrotechniky a informatiky Vysoké školy báňské – Technické univerzity Ostrava za pozorné přečtení celého textu. Velký dík patří zejména J. Bouchalovi a B. Krajcovi za mnohé cenné připomínky ke kapitolám 1, 3 a 6. Dále děkujeme L. Malému, R. Čosićovi a A. Markopoulovi za podněty ke kapitolám 2 a 4–6. M. Theuer nám pomohl vylepšit kapitolu o geometrických transformacích a M. Pětroš z Fakultní nemocnice Ostrava nám ochotně poskytl rady ohledně modelování změny koncentrace léků v krvi pacientů, moc jim za to děkujeme.

P. B., M. S.

Obsah

Úvod	iii
Poděkování	v
Značení	ix
1 Úlohy z matematické analýzy	1
1.1 Několik definic na úvod: limita a derivace funkce	1
1.2 Numerický výpočet určitého integrálu	4
1.3 Sčítáme řady - proč?	9
1.4 Monte Carlo	12
1.5 Numerické řešení rovnic	17
1.6 Jak pracuje kalkulačka – k čemu je dobrý Taylorův polynom?	22
2 Úlohy z lineární algebry	27
2.1 Několik definic na úvod: matice, součin matic a inverze matice	27
2.2 Jednoduchý model vyhledávání v databázích	29
2.3 Geometrické transformace	33
3 Počáteční úlohy	37
3.1 Stručný úvod do počátečních úloh a jak je řešit	37
3.2 Modelování změny koncentrace léků v krvi	39
3.3 Tak je to padělek nebo to není padělek aneb jak poznat stáří některých „Vermeerových“ obrazů?	42
3.4 Populační modely	46
3.5 Změna teploty tělesa	47
3.6 RC obvod	47
3.7 Matematické kyvadlo	49
4 Okrajové úlohy	53
4.1 Jednorozměrné (1D) okrajové úlohy a 1D metoda sítí	53

5	Optimalizační úlohy	57
5.1	Několik definic na úvod: gradient, Hessián	57
5.2	Rovnovážná poloha tělesa	58
6	Řešení úloh	65
6.1	Úlohy kapitoly 1	65
6.2	Úlohy kapitoly 2	85
6.3	Úlohy kapitoly 3	92
6.4	Úlohy kapitoly 4	112
6.5	Úlohy kapitoly 5	115

Značení

$A \setminus B$...	rozdíl množin A a B
\mathbb{N}	...	množina všech přirozených čísel
\mathbb{Z}	...	množina všech celých čísel
\mathbb{R}	...	množina všech reálných čísel
\mathbb{R}^+	...	množina všech kladných reálných čísel
\mathbb{R}^*	...	rozšířená číselná osa, tj. $\mathbb{R}^* = \mathbb{R} \cup \{+\infty, -\infty\}$
\mathbb{R}^n	...	množina všech uspořádaných n -tic reálných čísel
Df	...	definiční obor funkce f
Hf	...	obor hodnot funkce f
$\lim a_n$...	limita posloupnosti (a_n)
$\lim_{x \rightarrow x_0} f(x)$...	limita funkce f v bodě x_0
$f'(x)$...	(první) derivace funkce f v bodě x
$f''(x)$...	druhá derivace funkce f v bodě x
$f^{(n)}(x)$...	derivace n -tého řádu funkce f v bodě x
$\int_a^b f(x) dx$...	určitý integrál funkce f od a do b
$\sum_{n=1}^{\infty} a_n$...	řada reálných čísel
$\mathcal{U}_\delta(x_0)$...	okolí bodu x_0 o poloměru δ
$\ x\ $...	velikost vektoru x

Kapitola 1

Úlohy z matematické analýzy

1.1 Několik definic na úvod: limita a derivace funkce

V této kapitole si velmi stručně připomeneme základní pojmy matematické analýzy, které budeme v dalších kapitolách používat. Níže uvedené pojmy si lze mnohem podrobněji prostudovat např. v [1].

Nejprve si připomeneme definici limity posloupnosti reálných čísel. Řekneme, že posloupnost (a_n) má limitu $a \in \mathbb{R}$, a píšeme

$$\lim a_n = a \quad \text{nebo} \quad a_n \rightarrow a,$$

jestliže¹

$$(\forall \varepsilon \in \mathbb{R}^+)(\exists n_0 \in \mathbb{N})(\forall n \in \mathbb{N}, n > n_0) : |a_n - a| < \varepsilon.$$

Má-li posloupnost (konečnou) limitu, říkáme, že je konvergentní. V opačném případě se jedná o divergentní posloupnost. Řekneme, že posloupnost (a_n) má limitu $+\infty$, a píšeme

$$\lim a_n = +\infty \quad \text{nebo} \quad a_n \rightarrow +\infty,$$

jestliže

$$(\forall k \in \mathbb{R})(\exists n_0 \in \mathbb{N})(\forall n \in \mathbb{N}, n > n_0) : a_n > k.$$

Řekneme, že posloupnost (a_n) má limitu $-\infty$, a píšeme

$$\lim a_n = -\infty \quad \text{nebo} \quad a_n \rightarrow -\infty,$$

¹Připomeňme si, že symbolem \forall označujeme všeobecný kvantifikátor a čteme jej „pro všechna“. Symbol \exists označuje existenční kvantifikátor a čteme jej „existuje“.

jestliže

$$(\forall \ell \in \mathbb{R})(\exists n_0 \in \mathbb{N})(\forall n \in \mathbb{N}, n > n_0) : a_n < \ell.$$

Pomocí limity posloupnosti definujeme limitu reálné funkce jedné reálné proměnné. Vztahu $x_0 \neq x_n \rightarrow x_0$ rozumíme tak, že $x_n \rightarrow x_0$ a $x_n \neq x_0$ pro všechna dost velká $n \in \mathbb{N}$. Obdobně budeme chápat i vztahy $x_0 < x_n \rightarrow x_0$ a $x_0 > x_n \rightarrow x_0$. Řekneme, že funkce $f : \mathbb{R} \mapsto \mathbb{R}$ má v bodě $x_0 \in \mathbb{R}^*$ limitu $a \in \mathbb{R}^*$, a píšeme

$$\lim_{x \rightarrow x_0} f(x) = a,$$

pokud

$$x_0 \neq x_n \rightarrow x_0 \quad \Rightarrow \quad f(x_n) \rightarrow a.$$

Řekneme, že funkce f má v bodě $x_0 \in \mathbb{R}$ limitu zprava $a \in \mathbb{R}^*$, a píšeme

$$\lim_{x \rightarrow x_0+} f(x) = a,$$

pokud

$$x_0 < x_n \rightarrow x_0 \quad \Rightarrow \quad f(x_n) \rightarrow a.$$

Řekneme, že funkce f má v bodě $x_0 \in \mathbb{R}$ limitu zleva $a \in \mathbb{R}^*$, a píšeme

$$\lim_{x \rightarrow x_0-} f(x) = a,$$

pokud

$$x_0 > x_n \rightarrow x_0 \quad \Rightarrow \quad f(x_n) \rightarrow a.$$

Platí, že funkce f má v bodě $x_0 \in \mathbb{R}^*$ nejvýše jednu limitu.

Pojem limita funkce využijeme dále pro definici spojitosti funkce. Řekneme, že funkce f je spojitá v bodě $x_0 \in \mathbb{R}$, jestliže

$$\lim_{x \rightarrow x_0} f(x) = f(x_0).$$

Řekneme, že funkce f je spojitá zprava v bodě $x_0 \in \mathbb{R}$, jestliže

$$\lim_{x \rightarrow x_0+} f(x) = f(x_0).$$

Řekneme, že funkce f je spojitá zleva v bodě $x_0 \in \mathbb{R}$, jestliže

$$\lim_{x \rightarrow x_0-} f(x) = f(x_0).$$

Řekneme, že funkce f je spojitá v intervalu $I \subset \mathbb{R}$, platí-li současně:

- f je spojitá v každém vnitřním bodě intervalu I ,

-
- patří-li počáteční bod intervalu I do I , je v něm f spojitá zprava,
 - patří-li koncový bod intervalu I do I , je v něm f spojitá zleva.

Pomocí limity funkce budeme definovat pojem derivace funkce. Buď $f : \mathbb{R} \mapsto \mathbb{R}$ a $x \in \mathbb{R}$. Existuje-li

$$\lim_{h \rightarrow 0} \frac{f(x+h) - f(x)}{h},$$

nazveme ji derivací funkce f v bodě x a značíme ji $f'(x)$. Není-li řečeno jinak, budeme derivací rozumět vlastní (tj. konečnou) derivaci. Lze ukázat, že má-li funkce f v bodě $x_0 \in \mathbb{R}$ (konečnou) derivaci, je v bodě x_0 spojitá. Derivací funkce f budeme dále rozumět funkci f' definovanou předpisem

$$f'(x) := f'(x).$$

Zdefinujme si ještě derivace vyšších řádů. Předpokládejme nejprve, že funkce f má derivaci v bodě $x_0 \in \mathbb{R}$. Jestliže funkce derivace f' má derivaci v bodě x_0 , definujeme druhou derivací funkce f v bodě x_0 jako

$$f''(x_0) := (f')'(x_0).$$

Podobně postupujeme při definicích třetí, čtvrté, ... derivace funkce f v bodě x_0 . Indukcí tedy definujeme pro $n \in \mathbb{N}$ derivaci n -tého řádu funkce f v bodě x_0 jako

$$f^{(n)}(x_0) := (f^{(n-1)})'(x_0),$$

přičemž $f^{(0)} := f$.

Na závěr této kapitoly si ještě uvedme některá základní pravidla pro derivování funkcí. Buď $f, g : \mathbb{R} \mapsto \mathbb{R}$ a $x \in \mathbb{R}$. Pak platí

- $(f \pm g)'(x) = f'(x) \pm g'(x)$, má-li pravá strana rovnosti smysl,
- $(fg)'(x) = f'(x)g(x) + f(x)g'(x)$, existují-li (vlastní) derivace $f'(x)$ a $g'(x)$,
- $\left(\frac{f}{g}\right)'(x) = \frac{f'(x)g(x) - f(x)g'(x)}{g^2(x)}$, existují-li (vlastní) derivace $f'(x)$ a $g'(x)$ a je-li $g(x) \neq 0$.

Povězme si také, jak derivovat složenou funkci.² Nechť $x \in \mathbb{R}$ a necht' existují konečné derivace $f'(x)$ a $g'(f(x))$. Potom je

$$(g \circ f)'(x) = g'(f(x))f'(x).$$

Čtenáři se navíc mohou hodit předpisy pro derivování některých běžně používaných funkcí:

- $(c)' = 0$, $c \in \mathbb{R}$ (konst.), $x \in \mathbb{R}$,
- $(x^r)' = rx^{r-1}$, $r \in \mathbb{R}$, $x \in \mathbb{R}^+$,³
- $(\sin x)' = \cos x$, $x \in \mathbb{R}$,
- $(\cos x)' = -\sin x$, $x \in \mathbb{R}$,
- $(\operatorname{tg} x)' = \frac{1}{\cos^2 x}$, $x \in \mathbb{R} \setminus \left\{ \frac{\pi}{2} + k\pi : k \in \mathbb{Z} \right\}$,
- $(\operatorname{cotg} x)' = -\frac{1}{\sin^2 x}$, $x \in \mathbb{R} \setminus \{k\pi : k \in \mathbb{Z}\}$,
- $(e^x)' = e^x$, $x \in \mathbb{R}$,
- $(\ln x)' = \frac{1}{x}$, $x \in \mathbb{R}^+$.

1.2 Numerický výpočet určitého integrálu

Několik poznámek k určitému integrálu

Funkci $F: \mathbb{R} \mapsto \mathbb{R}$ nazveme **primitivní funkcí** k funkci $f: \mathbb{R} \mapsto \mathbb{R}$ na otevřeném intervalu $I \subset \mathbb{R}$, pokud pro každé $x \in I$ platí

$$F'(x) = f(x).$$

Lze ukázat, že pokud je f spojitá na otevřeném intervalu I , pak existuje primitivní funkce F k f na tomto intervalu.

Pomocí primitivní funkce zadefinujeme určitý integrál funkce f od a do b , kde $a, b \in \mathbb{R}$ a $a < b$. Jsou-li funkce f a F spojitě na uzavřeném intervalu

²Připomeňme, že pro funkce $f, g: \mathbb{R} \mapsto \mathbb{R}$ takové, že $Hg \subset Df$, definujeme **složenou funkci** $f \circ g$ předpisem $(f \circ g)(x) := f(g(x))$.

³Připomeňme si, že $x^0 := 1$.

$\langle a, b \rangle$ a je-li F primitivní k f na (a, b) , definujeme určitý integrál funkce f od a do b jako

$$\int_a^b f(x) dx := F(b) - F(a). \quad (1.1)$$

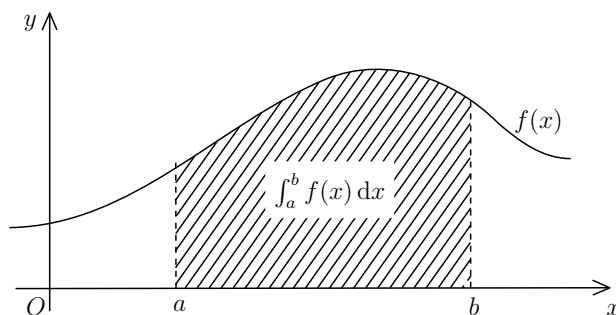
Vztah (1.1) se často nazývá Newtonův-Leibnizův vzorec.^{4,5} Pro zjednodušení zápisu se pro rozdíl hodnot $F(b)$ a $F(a)$ zavádí toto značení:

$$[F(x)]_a^b \stackrel{\text{ozn.}}{=} F(b) - F(a).$$

Popišme si nyní geometrický význam určitého integrálu. Nejprve předpokládejme, že je funkce f spojitá a nezáporná na intervalu $\langle a, b \rangle$. Pak integrál

$$\int_a^b f(x) dx$$

je roven obsahu rovinného obrazce ohraničeného osou x , grafem funkce f a přímkami $x = a$ a $x = b$, viz obr. 1.1. Nyní rozšířme naše úvahy na spojitě



Obrázek 1.1: Význam určitého integrálu

funkce, které mohou být v intervalu $\langle a, b \rangle$ záporné. Necht' je např. funkce f záporná na intervalu $\langle c, d \rangle \subset \langle a, b \rangle$, viz obr. 1.2. Potom také

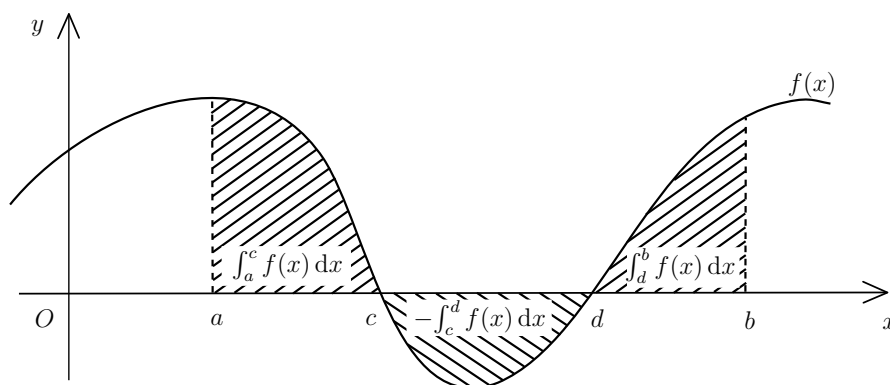
$$\int_c^d f(x) dx < 0.$$

Chceme-li pak pomocí tohoto integrálu vypočítat obsah plochy ohraničené osou x , grafem funkce f a přímkami $x = c$ a $x = d$, musíme na této části vzít integrál s opačným znaménkem. Bude-li nás dále zajímat obsah plochy, kterou ohraničuje osa x , graf funkce f a přímky $x = a$ a $x = b$, a bude-li funkce f protínat v intervalu $\langle a, b \rangle$ osu x , musíme tyto průsečíky najít

⁴Isaac Newton (1643–1727) – významný anglický fyzik, matematik a astronom

⁵Gottfried Wilhelm Leibniz (1646–1716) – významný německý matematik

a rozdělit interval $\langle a, b \rangle$ na intervaly, na nichž má f totéž znaménko. Na těchto intervalech spočteme integrály funkce f . Výsledný obsah plochy pak dostaneme jako součet všech vypočtených integrálů, přičemž integrály na těch intervalech, kde je f nekladná, musíme uvažovat se záporným znaménkem (viz obr. 1.2).



Obrázek 1.2: Obsah rovinného obrazce a určitý integrál

Přibližný výpočet určitého integrálu

V případě, že primitivní funkci nelze vyjádřit elementárními nebo tabelovanými funkcemi, Newtonův-Leibnizův vzorec (1.1) nemůžeme použít. Někdy zase může být hledání primitivní funkce příliš složité či časově náročné. V těchto případech často přistupujeme k tzv. numerickému výpočtu daného určitého integrálu, který nám dá přibližnou hodnotu integrálu s danou přesností. V následujícím textu si přiblížíme dvě základní metody přibližného výpočtu určitého integrálu, a to obdélníkové a lichoběžníkové pravidlo.

Obdélníkové pravidlo

Rozdělme nejprve interval $\langle a, b \rangle$ na n stejných dílků o délce

$$h = \frac{b - a}{n}. \quad (1.2)$$

Krajní body i -tého dílku postupně označme x_{i-1} a x_i ; platí tedy

$$x_0 = a < x_1 < \dots < x_{n-1} < x_n = b. \quad (1.3)$$

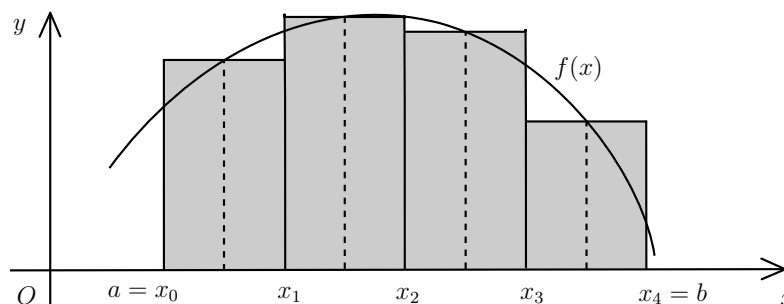
Vypočteme si dále středy jednotlivých dílků:

$$c_i = \frac{x_{i-1} + x_i}{2}, \quad i = 1, 2, \dots, n.$$

Na i -tém dílku pak funkci f nahradíme konstantní funkcí o hodnotě $f(c_i)$ a hledaný integrál budeme aproximovat takto:

$$\int_a^b f(x) dx \approx hf(c_1) + hf(c_2) + \dots + hf(c_n) = h \sum_{i=1}^n f(c_i).$$

Například integrál funkce f od a do b z obr. 1.3 tak nahrazujeme součtem obsahů příslušných obdélníků.



Obrázek 1.3: Aproximace obdélníky ($n = 4$)

Lze ukázat, že pokud existuje spojitá f'' na $\langle a, b \rangle$, potom pro chybu aproximace platí

$$\left| \int_a^b f(x) dx - h \sum_{i=1}^n f(c_i) \right| \leq \frac{(b-a)^3}{24n^2} \max_{x \in \langle a, b \rangle} |f''(x)|. \quad (1.4)$$

Lichoběžníkové pravidlo

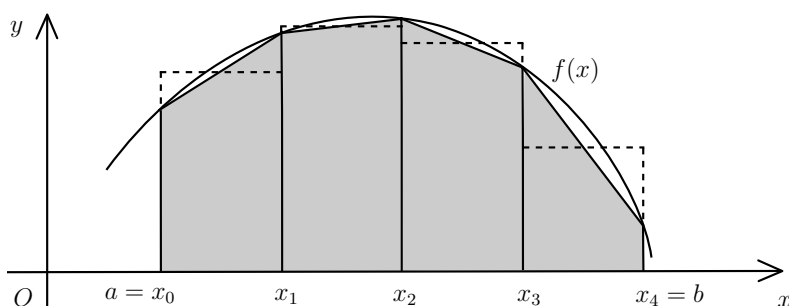
Interval $\langle a, b \rangle$ rozdělíme stejně jako u obdélníkového pravidla na dílky shodné délky, viz (1.2) a (1.3). Hledaný integrál budeme aproximovat takto:

$$\begin{aligned} \int_a^b f(x) dx &\approx h \frac{f(x_0) + f(x_1)}{2} + h \frac{f(x_1) + f(x_2)}{2} + \dots + h \frac{f(x_{n-1}) + f(x_n)}{2} = \\ &= \frac{h}{2} (f(x_0) + 2f(x_1) + \dots + 2f(x_{n-1}) + f(x_n)) = \frac{h}{2} \left(f(x_0) + f(x_n) + 2 \sum_{i=1}^{n-1} f(x_i) \right). \end{aligned}$$

Například integrál funkce f od a do b z obr. 1.4 tak nahrazujeme součtem obsahů příslušných lichoběžníků.

Je možné ukázat, že pokud existuje spojitá f'' na $\langle a, b \rangle$, potom pro chybu aproximace platí

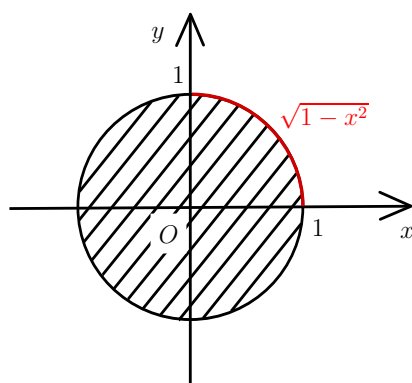
$$\left| \int_a^b f(x) dx - \frac{h}{2} \left(f(x_0) + f(x_n) + 2 \sum_{i=1}^{n-1} f(x_i) \right) \right| \leq \frac{(b-a)^3}{12n^2} \max_{x \in \langle a, b \rangle} |f''(x)|.$$



Obrázek 1.4: Aproximace lichoběžníky ($n = 4$)

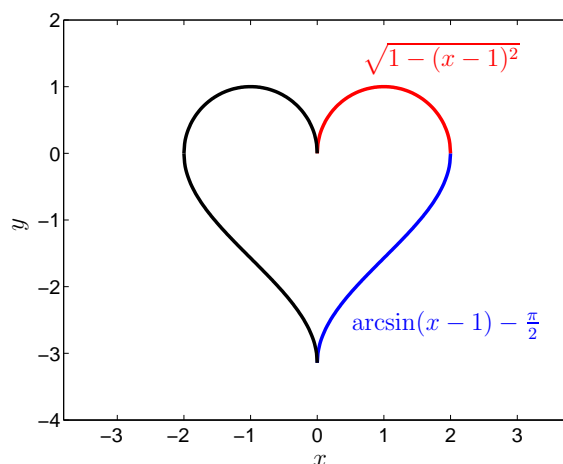
Všimněme si, že u lichoběžníkového pravidla máme odhad chyby aproximace horší než u obdélníkového pravidla, přestože u obdélníkového pravidla užíváme nahrazení konstantními funkcemi a u lichoběžníkového pravidla používáme nahrazení lineárními funkcemi.⁶ Z uvedených vzorců pro odhad chyby aproximace lze určit, jaký počet dílků zaručí požadovanou přesnost aproximace. Jelikož ovšem odhady obsahují druhé derivace, jejichž hodnoty není vždy lehké odhadnout, může být výpočet počtu potřebných dílků náročný a výsledek pesimistický.

Příklad 1 Pomocí obdélníkového a lichoběžníkového pravidla spočítejte přibližně obsah jednotkového kruhu a výsledky porovnejte se skutečným obsahem daného kruhu.



Obrázek 1.5: Jednotkový kruh

⁶Aproximační vlastnosti obdélníkového a lichoběžníkového pravidla si lze detailněji prostudovat např. v [10].



Obrázek 1.6: Matematické srdce

Příklad 2 Pomocí obdélníkového a lichoběžníkového pravidla spočítejte přibližně obsah srdce na obr. 1.6 a výsledky porovnejte se skutečným obsahem srdce.

Příklad 3 Aproximujte hodnotu integrálu

$$\int_0^1 x^2 dx$$

pomocí obdélníkového pravidla tak, aby chyba aproximace byla nejvýše 10^{-4} . Určete počet dílků dělení intervalu $\langle 0, 1 \rangle$, který zaručí dosažení požadované přesnosti.

1.3 Sčítáme řady - proč?

V této kapitole si nejprve povíme, co se skrývá pod pojmy reálná číselná řada a její konvergence. Řadou reálných čísel rozumíme výraz

$$a_1 + a_2 + \dots + a_n + \dots = \sum_{n=1}^{\infty} a_n, \quad (1.5)$$

kde $a_n \in \mathbb{R}$ pro každé $n \in \mathbb{N}$. Například:

i) $\sum_{n=1}^{\infty} (-1)^n = -1 + 1 + (-1) + 1 + \dots$ (příklad tzv. alternující řady⁷),

⁷Alternující řady jsou takové řady, jejichž členy pravidelně střídají znaménka.

- ii) $\sum_{n=1}^{\infty} n = 1 + 2 + 3 + 4 + \dots$ (příklad tzv. aritmetické řady),
- iii) $\sum_{n=1}^{\infty} q^{n-1} = 1 + q + q^2 + q^3 + \dots$, $q \in \mathbb{R}$ (příklad tzv. geometrické řady),
- iv) $\sum_{n=1}^{\infty} \frac{1}{n} = 1 + \frac{1}{2} + \frac{1}{3} + \frac{1}{4} + \dots$ (tzv. harmonická řada).

Číslo a_n nazýváme n -tým členem řady (1.5), posloupnost (s_n) definovanou předpisem

$$s_n := a_1 + a_2 + \dots + a_n = \sum_{k=1}^n a_k$$

nazýváme posloupností částečných součtů řady (1.5). Existuje-li limita

$$\lim s_n = s \in \mathbb{R}^*,$$

nazýváme ji součtem řady (1.5) a píšeme

$$\sum_{n=1}^{\infty} a_n = s.$$

Například:

- i) Součet $\sum_{n=1}^{\infty} (-1)^n$ neexistuje, jelikož $s_n = \begin{cases} 0 & \text{pro } n \text{ sudé,} \\ 1 & \text{pro } n \text{ liché.} \end{cases}$
- ii) Součet $\sum_{n=1}^{\infty} n = +\infty$, jelikož $s_n = \frac{n(n+1)}{2}$.
- iii) Součet $\sum_{n=1}^{\infty} q^{n-1} = \begin{cases} +\infty, & \text{je-li } q \geq 1, \\ \frac{1}{1-q}, & \text{je-li } |q| < 1, \\ \text{neexistuje,} & \text{je-li } q \leq -1, \end{cases}$
- jelikož $s_n = \begin{cases} n & \text{pro } q = 1, \\ \frac{1-q^n}{1-q} & \text{pro } q \neq 1. \end{cases}$
- iv) Lze ukázat, že součet $\sum_{n=1}^{\infty} \frac{1}{n} = +\infty$.

Říkáme dále, že řada **konverguje**, jestliže je její součet roven nějakému reálnému číslu. V opačném případě, tj. pokud má řada součet roven $+\infty$ nebo $-\infty$ nebo pokud řada součet nemá, říkáme, že řada **diverguje**. Například:

i) $\sum_{n=1}^{\infty} (-1)^n$ diverguje.

ii) $\sum_{n=1}^{\infty} n$ diverguje.

iii) $\sum_{n=1}^{\infty} q^{n-1}$ konverguje, pokud $|q| < 1$, a diverguje, pokud $|q| \geq 1$.

iv) $\sum_{n=1}^{\infty} \frac{1}{n}$ diverguje.

Výše uvedené pojmy můžeme nyní využít k řešení následujících úloh.

Příklad 1 Prozrad'me, že řada

$$\sum_{n=1}^{\infty} \frac{1}{n(n+1)}$$

konverguje a její posloupnost částečných součtů konverguje „rychle“ k součtu této řady. Zkuste odhadnout součet této řady.

Příklad 2 Mezi dvěma městy A a B se nachází přímá železniční trať, jejíž délka je 90 km. Z města A do města B vyjede vlak rychlostí 10 km/h. V tu samou chvíli vyjede z města B vlak do města A po té samé koleji stejnou rychlostí. Ve chvíli, kdy se vlaky rozjedou vstříc jisté zkáze, z předního okna lokomotivy vlaku jedoucího z A do B se odrazí moucha rychlostí 100 km/h a letí vstříc druhému vlaku. Ve chvíli, kdy k němu doletí, dotkne se nožkou jeho předního skla a letí zpátky. Takto moucha lítá mezi vlaky, než ji rozmáčknou na placku při jejich srážce. Kolik kilometrů moucha celkem nalétala? Jakou vzdálenost moucha urazila mezi 15. a 16. odražením se od oken lokomotiv (jinými slovy: jak dlouhý byl její 15. let mezi vlaky)?

Příklad 3 Kolik členů harmonické řady musíte nejméně sečíst, aby tento částečný součet řady měl hodnotu alespoň 10 (15, 20)?

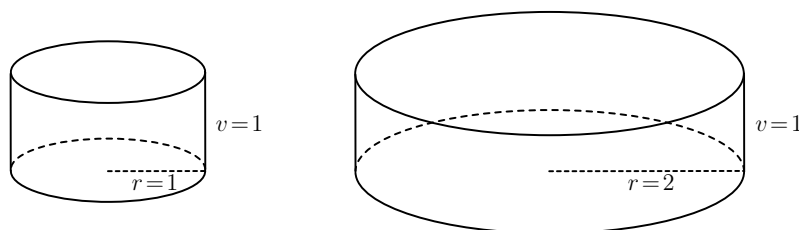
Příklad 4 Prozrad' me, že řada

$$\sum_{n=1}^{\infty} 4 \frac{(-1)^{n+1}}{2n-1}$$

konverguje a její posloupnost částečných součtů konverguje „rychle“ k součtu této řady. Zkuste odhadnout součet této řady.

1.4 Monte Carlo

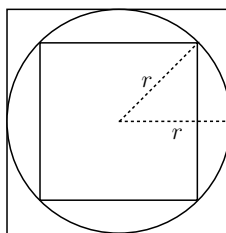
V této kapitole se budeme věnovat odhadům Ludolfova⁸ čísla π . Nejprve hledejme obsah kruhu o poloměru r (představme si na chvíli, že neznáme příslušný vzorec a nic nevíme o číslu π). Prvním nápadem by mohlo být zhotovení válcových nádob s různými poloměry podstav (např. s poloměry o délce 1 a 2 jednotek) a jednotkovou výškou, viz obr. 1.7. Objem vody, který



Obrázek 1.7: Válcové nádoby

se do takových nádob vejde, je roven obsahu podstavy válce, tj. obsahu kruhu s poloměrem r . Rychle si všimneme, že pokud zvětšíme poloměr podstavy dvakrát, zvětší se objem čtyřikrát, a následně pozorujeme, že obsah kruhu je přímo úměrný druhé mocnině poloměru. Také zjistíme, že druhou mocninu poloměru kruhu musíme vynásobit vhodnou konstantou, abychom dostali správnou hodnotu obsahu daného kruhu. Tuto konstantu označíme π . Existuje mnoho možností, jak tuto konstantu odhadnout. Snad nejjednodušším způsobem jak stanovit meze pro π , je vepsat do kruhu o poloměru r čtverec a stejnému kruhu opsat jiný čtverec, viz obr. 1.8. Délka strany většího čtverce je $2r$ a z Pythagorovy věty dále zjistíme, že délka strany menšího čtverce je $\sqrt{2}r$, tudíž obsah většího resp. menšího čtverce je $4r^2$ resp. $2r^2$. Jelikož jsme si už „odvodili“, že obsah kruhu o poloměru r je dán vzorcem πr^2 , pouhým porovnáním obsahů čtverců a kruhu zjistíme, že $\pi \in (2, 4)$. Tuto metodu

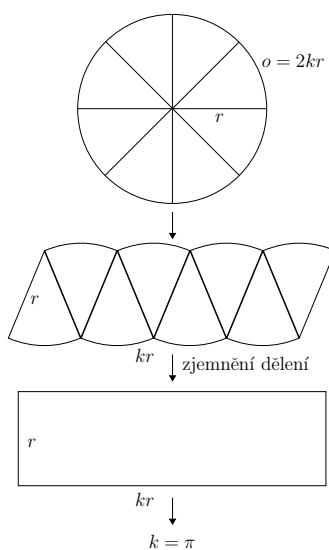
⁸Ludolph van Ceulen (1540–1610) – německý matematik



Obrázek 1.8: Exhaustní metoda pro stanovení mezí pro π

nazýváme vyčerpávací (exhaustní) a pravděpodobně první ji použil Eudoxos.⁹

Než se budeme věnovat dalším odhadům čísla π , podívejme se krátce na výpočet obvodu kruhu. Jistě víme, že obvod kruhu je přímo úměrný dvojnásobku jeho poloměru, ale abychom dostali správnou hodnotu, je nutno $2r$ vynásobit vhodnou konstantou k . Na obr. 1.9 provedeme přeuspořádání kruhu na útvar, který se pro zjemňující se dělení kruhu blíží obdélníku. Porovnáním obsahu kruhu (πr^2) a obsahu vzniklého obdélníku (kr^2) je názorně vidět, že konstanta k je opět rovna π .



Obrázek 1.9: Stanovení vzorce pro obvod kruhu

Nyní si ukážeme několik způsobů, jak nalézt přibližnou hodnotu čísla π .

⁹Eudoxos (410 nebo 408 př. n. l. – 355 nebo 347 př. n. l.) – řecký astronom, matematik a fyzik, student Platóna

Buffonova metoda

Řešením tzv. Buffonova¹⁰ problému s jehlou je aproximace čísla π . Úloha spočívá v opakovaném házení jehly o délce ℓ na rovinu (papír), na které máme vyznačenu síť rovnoběžek se vzdáleností 2ℓ . Jestliže jehlu hodíme n -krát a s -krát nám během těchto pokusů po dopadu zkříží některou z rovnoběžek, pak číslo

$$\frac{n}{s}$$

aproximuje číslo π .

Nyní si uveďme odhad týkající se přesnosti Buffonovy metody. Lze ukázat, že s pravděpodobností 95 procent nemá chyba aproximace hodnotu větší než $9,011/\sqrt{n}$. Tzn. například pro 10^4 pokusů nám s pravděpodobností 95 procent chyba nepřekročí hodnotu 0,09011.

Příklad 1 Implementujte Buffonovu metodu a použijte ji k aproximaci čísla π . Porovnejte vaši aproximaci se „skutečnou“ hodnotou čísla π a určete chybu aproximace.

Metoda Monte Carlo

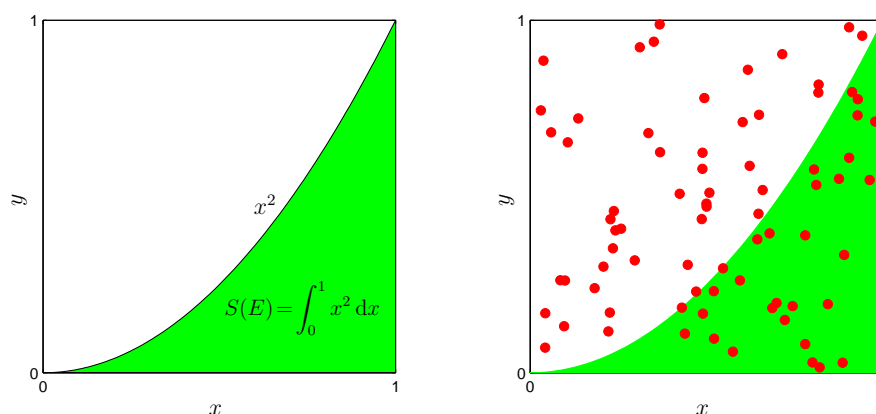
Monte Carlo je třída výpočetních algoritmů založená na provádění náhodných experimentů. Této metody se často používá pro simulaci fyzikálních a matematických systémů. Výsledkem provedení velkého množství experimentů je obvykle pravděpodobnost určitého jevu. Na základě získané pravděpodobnosti a známých vztahů pak spočítáme potřebné výsledky. Protože metoda vyžaduje generování velkého souboru náhodných dat, je vhodné pro její implementaci použití počítače. Metod Monte Carlo se používá v případě, kdy je příliš pracné nebo nemožné nalézt přesný výsledek jiným způsobem. Její výhodou je jednoduchá implementace, nevýhodou relativně malá přesnost. Metoda byla vytvořena skupinou fyziků pracujících na projektu jaderné pumy v Los Alamos, jméno metody bylo navrženo v roce 1940 von Neumannem.¹¹

V matematice se Monte Carlo používá zejména pro výpočet určitých integrálů (zejména vícenásobných určitých integrálů), které je obtížné či nemožné vyčíslit analyticky nebo jinou numerickou metodou. Např. obsah omezené plochy E ohraničené grafem funkce $y = x^2$, osou x a přímkou $x = 1$ (tj. $S(E) = \int_0^1 x^2 dx$, viz obr. 1.10) je možné metodou Monte Carlo vypočítat následujícím způsobem. Nechť náš program generuje náhodně dvojice čísel

¹⁰Georges Louis Leclerc de Buffon (1707–1788) – francouzský přírodovědec

¹¹John von Neumann (1903–1957) – významný maďarský matematik

$[x, y]$, přičemž každé z čísel x a y je vybráno nezávisle z intervalu $\langle 0, 1 \rangle$. Tuto dvojici budeme chápat jako souřadnice bodu, který je náhodně zvolen ve čtverci $\langle 0, 1 \rangle \times \langle 0, 1 \rangle$. Pravděpodobnost toho, že bod leží uvnitř zadaného čtverce, je 1. Pravděpodobnost toho, že bod leží uvnitř podmnožiny E zadaného čtverce, je rovna obsahu plochy E , tj. $S(E)$. Tedy obsah plochy, která je podmnožinou zvoleného čtverce, můžeme odhadnout jako pravděpodobnost, že náhodně zvolený bod z daného čtverce leží v této podmnožině.



Obrázek 1.10: Plocha E a Monte Carlo

Příklad 2 Implementujte metodu Monte Carlo pro přibližný výpočet $\int_0^1 x^2 dx$. Získané výsledky můžete porovnat s analytickým vyčíslením integrálu.

Pro odhad přesnosti metody Monte Carlo lze odvodit například následující tvrzení. S pravděpodobností 75 procent nemá chyba aproximace hodnotu větší než $1/\sqrt{n}$. Tzn. například pro 10^4 pokusů nám s pravděpodobností 75 procent chyba nepřekročí hodnotu 0,01.

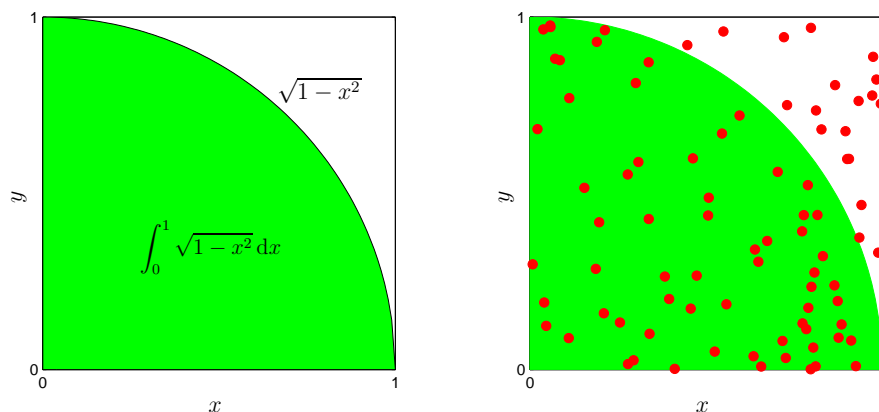
Chceme-li využít metodu Monte Carlo k aproximaci čísla π , vypočteme přibližně touto metodou (například) integrál

$$\int_0^1 \sqrt{1-x^2} dx.$$

Snadno si uvědomíme (viz obr. 1.11), že tímto způsobem získáme aproximaci hodnoty $\pi/4$.

Příklad 3 Implementujte metodu Monte Carlo pro přibližný výpočet integrálu $\int_0^1 \sqrt{1-x^2} dx$ a použijte ji k aproximaci čísla π . Porovnejte vaši

aproximaci se „skutečnou“ hodnotou čísla π a určete chybu aproximace.



Obrázek 1.11: π a Monte Carlo

Připomeňme ještě, že výše uvedené použití metody Monte Carlo je pouze ilustrativní; jednoduchý integrál z příkladu 3 lze velmi efektivně spočítat jinými numerickými integračními metodami.

Aproximace π pomocí číselných řad

Poslední metodou nalezení aproximace čísla π , kterou si v tomto přehledu ukážeme, je využití číselných řad. K této aproximaci použijeme tzv. Leibnizovu řadu, která je rozvojem funkce arkustangens:

$$\operatorname{arctg} x = x - \frac{x^3}{3} + \frac{x^5}{5} - \frac{x^7}{7} + \dots = \sum_{n=1}^{\infty} (-1)^{n+1} \frac{x^{2n-1}}{2n-1}.$$

Tato řada má konečný součet pro $x \in \langle -1, 1 \rangle$ (říkáme, že řada konverguje), navíc platí, že čím více je $|x|$ bližší 0, tím méně členů řady potřebujeme použít k nahrazení $\operatorname{arctg} x$ s „uspokojivou“ přesností. Pro $x = 1$ dostáváme z Leibnizovy řady tzv. Gregoryho¹² řadu

$$1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \dots = \sum_{n=1}^{\infty} \frac{(-1)^{n+1}}{2n-1}.$$

Jelikož

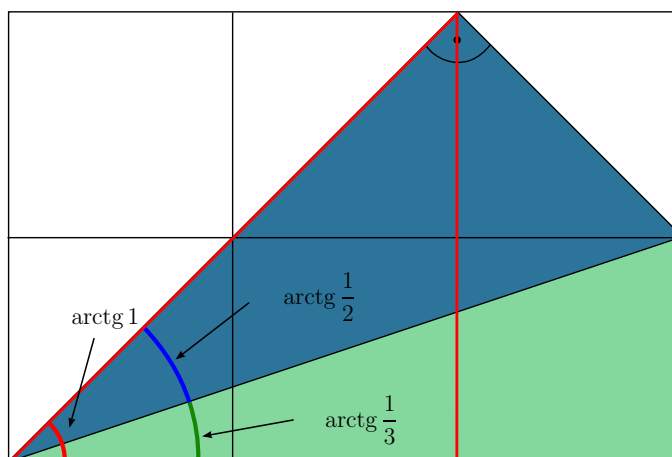
$$\operatorname{arctg} 1 = \frac{\pi}{4},$$

¹²James Gregory (1638–1675) – skotský matematik a astronom

můžeme Gregoryho řadu využít k aproximaci čísla π . Aproximaci s rychlejší konvergencí pak získáme, pokud použijeme rovnost

$$\operatorname{arctg} 1 = \operatorname{arctg} \frac{1}{2} + \operatorname{arctg} \frac{1}{3}, \quad (1.6)$$

viz obr. 1.12.



Obrázek 1.12: Ilustrace k odvození vztahu (1.6)

Příklad 4 Implementujte metodu, která využije Gregoryho, popř. Leibnizovu řadu k nalezení aproximace čísla π .

1.5 Numerické řešení rovnic

Velmi často se (nejen) matematik setkává s úkolem, kdy je třeba vyřešit nějakou nelineární rovnici. Tento úkol bývá většinou složitý a řešení nelineární rovnice často nelze najít analyticky. Proto je užitečné vědět, jak lze rovnice řešit alespoň přibližně pomocí numerických metod. V následujícím textu si ukážeme dvě jednoduché iterační metody, které umožňují nalézt alespoň přibližné řešení nelineární rovnice.

Metoda prosté iterace

Bod $x^* \in \mathbb{R}$ nazýváme pevným bodem funkce $f : \mathbb{R} \mapsto \mathbb{R}$, pokud

$$f(x^*) = x^*.$$

Věta Bud' $\emptyset \neq I \subset \mathbb{R}$ uzavřený omezený interval a $f : \mathbb{R} \mapsto \mathbb{R}$. Nechť f je spojitá v I a nechť f má spojitou derivaci v I . Ať navíc existuje $\lambda \in \mathbb{R}$ takové, že¹³

$$|f'(x)| \leq \lambda < 1 \quad \text{pro všechna } x \in I.$$

Zvolme bod $x_0 \in I$. Jestliže každý bod posloupnosti (x_n) definované předpisem

$$x_{n+1} := f(x_n), \quad n = 0, 1, 2, 3, \dots,$$

leží v I , pak existuje $x^* \in I$ takové, že

$$f(x^*) = x^* = \lim x_n,$$

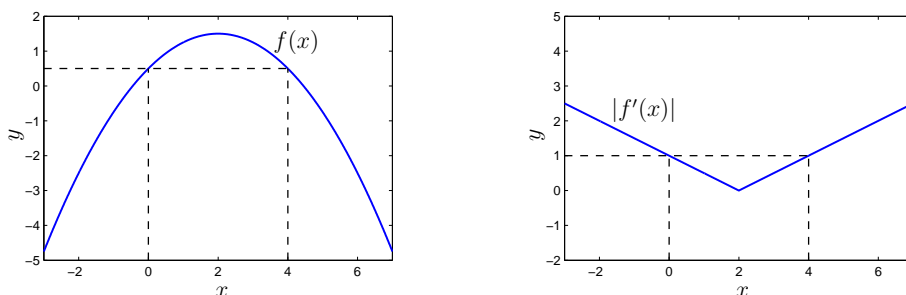
tj. (x_n) konverguje k pevnému bodu f .

Lze ukázat, že čím více je hodnota nejmenšího horního odhadu množiny $\{|f'(x)| : x \in I\}$ bližší 0, tím rychleji posloupnost (x_n) konverguje k pevnému bodu f .

Pro dané x_0 uvažujme posloupnost (x_n) danou předpisem

$$x_{n+1} := x_n - \frac{1}{4}(x_n^2 - 2), \quad n = 0, 1, 2, 3, \dots \quad (1.7)$$

Všimněme si nejprve, že $\pm\sqrt{2}$ jsou pevné body funkce $f(x) := x - \frac{1}{4}(x^2 - 2)$. Lze si rozmyslet (viz také obr. 1.13), že posloupnost (1.7) konverguje k $\sqrt{2}$ pro libovolné $x_0 \in (0, 4)$.



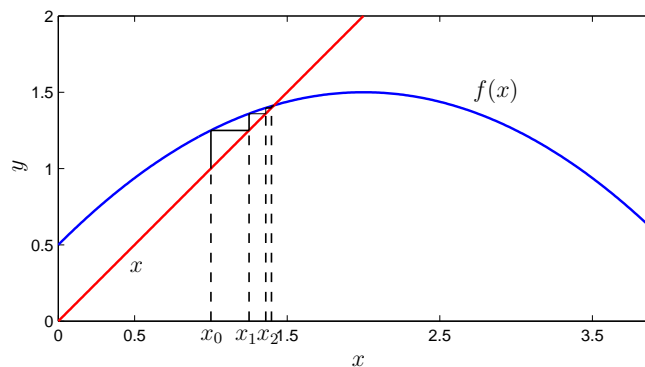
Obrázek 1.13: Grafy funkcí $f(x) = x - \frac{1}{4}(x^2 - 2)$ a $|f'(x)| = |1 - \frac{1}{2}x|$

¹³V pravém, resp. levém krajním bodě intervalu I nepožadujeme existenci derivace, ale pouze existenci derivace zleva, resp. zprava. **Derivaci zleva**, resp. **zprava funkce f v bodě** $x \in \mathbb{R}$ definujeme jako $\lim_{h \rightarrow 0^-} \frac{f(x+h) - f(x)}{h}$, resp. $\lim_{h \rightarrow 0^+} \frac{f(x+h) - f(x)}{h}$.

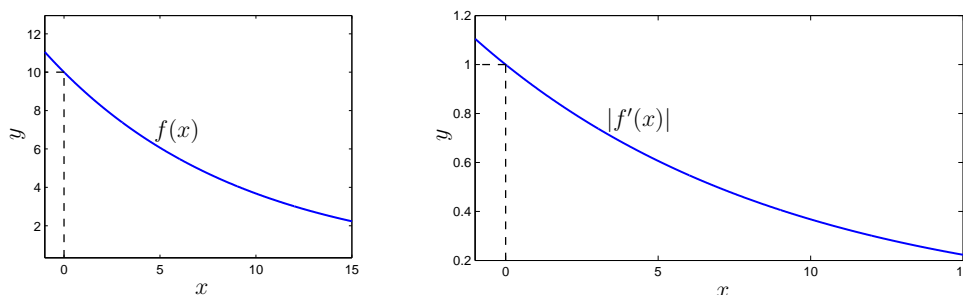
Algoritmus (Metoda prosté iterace)

1. $\varepsilon > 0$ (ukončující podmínka)
 x_0 (počáteční aproximace)
 $n = 0$
 $x_1 = f(x_0)$
2. while $|x_{n+1} - x_n| \geq \varepsilon$
 $n = n + 1$
 $x_{n+1} = f(x_n)$
end
3. x_{n+1} aproximuje řešení rovnice $f(x) = x$

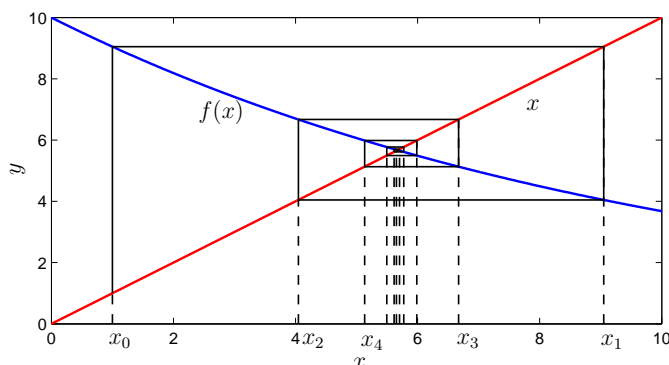
Na obr. 1.14 je ilustrována konvergence posloupnosti (1.7) k pevnému bodu $\sqrt{2}$ funkce f pro $x_0 = 1$.



konverguje k pevnému bodu funkce f pro libovolné $x_0 > 0$. Na obr. 1.16 je ilustrována konvergence této posloupnosti k pevnému bodu funkce f pro $x_0 = 1$.



Obrázek 1.15: Grafy funkcí $f(x) = 10 e^{-\frac{x}{10}}$ a $|f'(x)| = | -e^{-\frac{x}{10}} |$



Obrázek 1.16: Ilustrace konvergence metody prosté iterace k pevnému bodu $x^* \doteq 5,67143290$ funkce $f(x) = 10 e^{-\frac{x}{10}}$

Newtonova metoda

Zabývejme se nyní řešením rovnice

$$g(x) = 0. \tag{1.9}$$

Předpokládejme, že g je spojitá a má spojitou a nenulovou derivaci v nějakém okolí¹⁴ $\mathcal{U}(x^*)$, kde x^* řeší rovnici (1.9). Budeme vycházet z metody prosté

¹⁴Okolím $\mathcal{U}(x^*)$ budeme rozumět nějaký otevřený omezený interval se středem v x^* .

iterace. Protože x^* řeší rovnici (1.9), tak x^* řeší také rovnici

$$x = f(x) := x - \frac{g(x)}{h(x)}, \quad (1.10)$$

kde h je libovolná nenulová a diferencovatelná funkce na okolí $\mathcal{U}(x^*)$. Pak dále platí, že

$$f'(x) = 1 - \frac{g'(x)h(x) - g(x)h'(x)}{h^2(x)} \quad \text{pro všechna } x \in \mathcal{U}(x^*).$$

Pro „dobrou“ konvergenci budeme požadovat $f'(x^*) = 0$, tj.

$$1 - \frac{g'(x^*)}{h(x^*)} = 0.$$

Odtud $h(x^*) = g'(x^*)$. Volíme proto $h(x) = g'(x)$. Dostáváme tak, že posloupnost (x_n) definovaná předpisem

$$x_{n+1} := x_n - \frac{g(x_n)}{g'(x_n)}, \quad n = 0, 1, 2, 3, \dots, \quad (1.11)$$

konverguje k řešení x^* rovnice (1.9), pokud x_0 je dostatečně blízko x^* , funkce f je daná (1.10) a jsou splněny předpoklady věty ze str. 18.

Algoritmus (Newtonova metoda)

1. $\varepsilon > 0$ (ukončující podmínka)
 x_0 (počáteční aproximace)
 $n = 0$
 $x_1 = x_0 - g(x_0)/g'(x_0)$
2. while $|x_{n+1} - x_n| \geq \varepsilon$
 $n = n + 1$
 $x_{n+1} = x_n - g(x_n)/g'(x_n)$
end
3. x_{n+1} aproximuje řešení rovnice $g(x) = 0$

Připomeňme si ještě, že rovnice tečny t grafu funkce g v dotykovém bodě $(x_n, g(x_n))$ má tvar

$$t: \quad y = g'(x_n)(x - x_n) + g(x_n).$$

Hledáme-li průsečík přímky t a osy x , řešíme rovnici

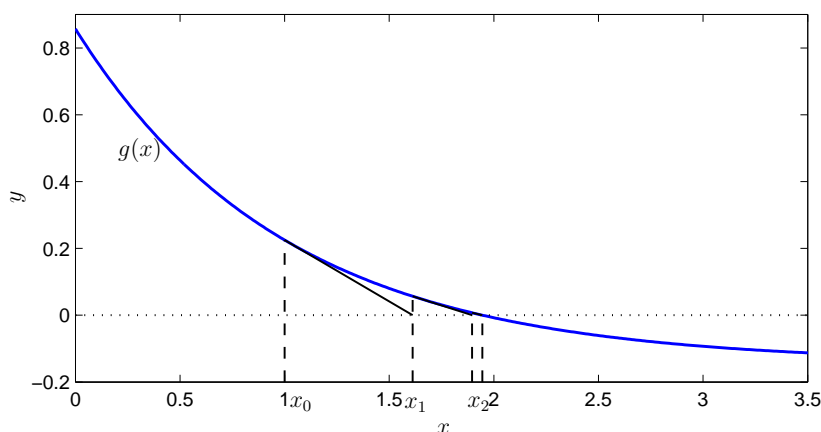
$$g'(x_n)(x - x_n) + g(x_n) = 0. \quad (1.12)$$

Je-li $g'(x_n) \neq 0$, pak lze řešení rovnice (1.12) psát ve tvaru

$$x = x_n - \frac{g(x_n)}{g'(x_n)}.$$

Porovnáme-li pak poslední rovnost s iterační formulí (1.11), zjistíme, že každé x_{n+1} vytvoříme jako průsečík tečny sestavené ke grafu funkce g v dotykovém bodě $(x_n, g(x_n))$ a osy x . Proto se Newtonova metoda často nazývá **metodou tečen**.

Na obr. 1.17 ilustrujeme přibližné řešení rovnice $g(x) := e^{-x} - 1/7 = 0$ pomocí Newtonovy metody pro $x_0 = 1$.



Obrázek 1.17: Ilustrace konvergence Newtonovy metody k řešení $x^* \doteq 1,94591015$ rovnice $g(x) := e^{-x} - 1/7 = 0$

Příklad 1 Pomocí Newtonovy metody aproximujte řešení rovnic

1) $\ln x + (x - 1)^3 = 0$,

2) $x + e^{-x^2} = 1$.

Příklad 2 Pomocí Newtonovy metody najděte přibližné hodnoty čísel $\sqrt{2}$ a π .

1.6 Jak pracuje kalkulačka – k čemu je dobrý Taylorův polynom?

Při řešení různých matematických úloh se setkáváme s potřebou vyčíslit hodnotu nějaké funkce v daném bodě. V některých případech to ale může být

obtížné. Uvažujeme-li například funkci \sqrt{x} , pak určit hodnotu této funkce v bodě 1 je jednoduché: $\sqrt{1} = 1$. Jak ale pomocí desetinného čísla (alespoň přibližně) zapsat hodnotu funkce \sqrt{x} například v bodě 2? Nabízí se řešení: použijeme kalkulačku. Jak ovšem číslo 1,414213... spočetla kalkulačka? V této kapitole si ukážeme postup, jak lze přibližně vypočítat hodnoty „obtížně vyčíslitelných“ funkcí v ruce a simulovat tím práci jednoduché kalkulačky. Nejprve si zavedeme několik pojmů, s nimiž budeme dále pracovat.

Má-li funkce f v bodě x_0 derivace až do řádu $n \in \mathbb{N}$ včetně, definujeme Taylorův¹⁵ polynom řádu n funkce f v bodě x_0 vztahem

$$\begin{aligned} T_n(x) &:= f(x_0) + f'(x_0)(x - x_0) + \frac{f''(x_0)}{2!}(x - x_0)^2 + \dots + \frac{f^{(n)}(x_0)}{n!}(x - x_0)^n \\ &= \sum_{k=0}^n \frac{f^{(k)}(x_0)}{k!}(x - x_0)^k. \end{aligned}$$

Okolím bodu $x_0 \in \mathbb{R}$ o poloměru $\delta > 0$ budeme rozumět otevřený interval $(x_0 - \delta, x_0 + \delta)$ a budeme jej značit $\mathcal{U}_\delta(x_0)$.

Nyní budeme chtít aproximovat funkci f na okolí $\mathcal{U}_\delta(x_0)$ Taylorovým polynomem,¹⁶ čímž se dopustíme určité chyby. Výhodou lokální aproximace funkce pomocí polynomu je například to, že funkční hodnotu polynomu lze spočítat pouze pomocí operací sčítání a násobení. Obvykle čím vyšší stupeň Taylorova polynomu pak budeme při aproximaci uvažovat, tím menší se dopustíme chyby (tj. tím přesnější nahrazení získáme).

Taylorova věta *Předpokládejme, že funkce f má v každém bodě okolí $\mathcal{U}_\delta(x_0)$ všechny derivace až do řádu $n + 1$ včetně, a uvažujme $x \in \mathcal{U}_\delta(x_0)$. Pak existuje číslo ξ ležící mezi x_0 a x takové, že platí*

$$f(x) = T_n(x) + R_{n+1}(x),$$

kde R_{n+1} – tzv. zbytek po n -tém členu – je daný vztahem

$$R_{n+1}(x) = \frac{f^{(n+1)}(\xi)}{(n+1)!}(x - x_0)^{n+1}.$$

Uvedená podoba zbytku se nazývá Lagrangeův¹⁷ tvar zbytku. Existuje více (někdy užitečnějších) tvarů zbytku. Přesnost aproximace hodnoty funkce

¹⁵Brook Taylor (1685–1731) – anglický matematik

¹⁶Toto nahrazení budeme zapisovat takto: $f(x) \approx T_n(x)$.

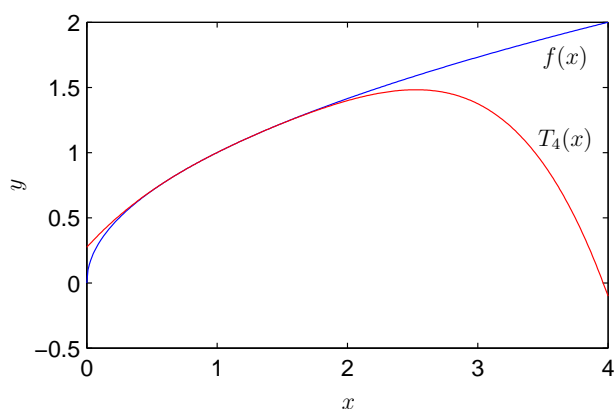
¹⁷Joseph Louis Lagrange (1736–1813) – významný italsko-francouzský matematik a mechanik

f v $x \in \mathcal{U}_\delta(x_0)$ hodnotou Taylorova polynomu T_n v x lze zjistit odhadem velikosti zbytku $R_{n+1}(x)$.

Vraťme se na chvíli zpět k našemu problému jak (přibližně) vyjádřit hodnotu čísla $\sqrt{2}$. Použijeme výše uvedených úvah a funkci $f(x) = \sqrt{x}$ nahradíme na okolí bodu $x_0 = 1$ Taylorovým polynomem například stupně 4. Dostaneme tak (viz také obr. 1.18)

$$\sqrt{x} \approx T_4(x) = f(1) + f'(1)(x-1) + \frac{f''(1)}{2}(x-1)^2 + \frac{f'''(1)}{6}(x-1)^3 + \frac{f^{(4)}(1)}{24}(x-1)^4,$$

a proto pro $x = 2$ máme



Obrázek 1.18: Nahrazení funkce f Taylorovým polynomem T_4 v bodě 1

$$\sqrt{2} \approx T_4(2) = f(1) + f'(1) + \frac{f''(1)}{2} + \frac{f'''(1)}{6} + \frac{f^{(4)}(1)}{24}.$$

Počítejme:

- $f'(x) = \frac{1}{2\sqrt{x}}, f'(1) = \frac{1}{2},$
- $f''(x) = -\frac{1}{4\sqrt{x^3}}, f''(1) = -\frac{1}{4},$
- $f'''(x) = \frac{3}{8\sqrt{x^5}}, f'''(1) = \frac{3}{8},$
- $f^{(4)}(x) = -\frac{15}{16\sqrt{x^7}}, f^{(4)}(1) = -\frac{15}{16}.$

Tedy

$$\sqrt{2} \approx 1 + \frac{1}{2} - \frac{1}{8} + \frac{3}{48} - \frac{15}{384} = 1,3984375.$$

Není těžké v tomto případě odhadnout velikost zbytku $R_5(2)$. Protože

$$f^{(5)}(x) = \frac{105}{32\sqrt{x^9}},$$

platí

$$R_5(2) \leq \frac{f^{(5)}(1)}{5!} (2-1)^5 = \frac{7}{256} = 0,02734375.$$

Lepší přesnosti vyčíslení hodnoty $\sqrt{2}$ bychom dosáhli, pokud bychom zvolili vyšší řád Taylorova polynomu funkce \sqrt{x} v bodě 1.^{18,19}

Příklad 1 Pomocí Taylorova polynomu určete přibližně hodnotu čísla e .²⁰

Příklad 2 Pomocí Taylorova polynomu určete přibližně hodnotu čísla π .²¹

Zkusme se zamyslet, zda by nešlo naši jednoduchou kalkulačku nějak zrychlit, tedy zda můžeme dosáhnout stejného přiblížení k požadované hodnotě s využitím menšího počtu aritmetických operací. Odpověď zní ano – můžeme použít metodu prosté iterace, kterou jsme si již popsali v kapitole 1.5. Připomeňme si v tuto chvíli alespoň daný iterační algoritmus pro nalezení řešení rovnice $f(x) = x$, tj. pro nalezení pevného bodu funkce f , za předpokladu, že f splňuje předpoklady věty ze str. 18.

¹⁸Zkuste si rozmyslet, že $T_n(2) = 1 + \frac{1}{1!} \cdot \frac{1}{2} + \frac{1}{2!} \cdot \left(-\frac{1}{4}\right) + \frac{1}{3!} \cdot \frac{3}{8} + \frac{1}{4!} \cdot \left(-\frac{15}{16}\right) + \dots + \frac{1}{n!} \left(\frac{1}{2} \cdot \left(-\frac{1}{2}\right) \cdot \left(-\frac{3}{2}\right) \cdot \left(-\frac{5}{2}\right) \cdot \dots \cdot \left(-\frac{2n-3}{2}\right)\right) = 1 + \sum_{k=1}^n \frac{(-1)^k}{k!} \cdot \frac{-1 \cdot 1 \cdot 3 \cdot 5 \cdot \dots \cdot (2k-3)}{2^k}$.

¹⁹Leckdy je namísto zvyšování řádu n výhodnější lépe zvolit x_0 , např. v tomto případě je rozumnou volbou $x_0 = 49/25$.

²⁰Nápověda: Použijte Taylorův polynom stupně n funkce $f(x) = e^x$ ve vhodném bodě x_0 .

²¹Nápověda: Použijte Taylorův polynom stupně n funkce $f(x) = \operatorname{arctg} x$ ve vhodném bodě x_0 .

Algoritmus (Metoda prosté iterace)

1. $\varepsilon > 0$ (ukončující podmínka)
 x_0 (počáteční aproximace)
 $n = 0$
 $x_1 = f(x_0)$

2. while $|x_{n+1} - x_n| \geq \varepsilon$
 $n = n + 1$
 $x_{n+1} = f(x_n)$
end

3. x_{n+1} aproximuje řešení rovnice $f(x) = x$

Připomeňme si dále (viz str. 18), že posloupnost (x_n) daná předpisem

$$x_{n+1} := x_n - k(x_n^2 - 2), \quad n = 0, 1, 2, 3, \dots, \quad k := \frac{1}{4}, \quad (1.13)$$

konverguje k $\sqrt{2}$ pro libovolné $x_0 \in (0, 4)$. ($\sqrt{2}$ je pevným bodem funkce $f(x) := x - k(x^2 - 2)$.)

Příklad 3 Změňte koeficient k v (1.13) tak, aby daná posloupnost konvergovala k $\sqrt{2}$ rychleji než stávající posloupnost (1.13), pokud počáteční aproximace je $x_0 = 1$. Změňte koeficient k v (1.13) tak, aby posloupnost konvergovala k $\sqrt{2}$ pro $x_0 = 5$.

Příklad 4 Pomocí metody prostých iterací určete přibližně hodnotu čísla π .

Kapitola 2

Úlohy z lineární algebry

2.1 Několik definic na úvod: matice, součin matic a inverze matice

Uvažujme prvky x_1, x_2, \dots, x_n z dané množiny $\mathcal{F} \subset \mathbb{R}$. Aritmetický vektor (n -rozměrný) je uspořádaná n -tice čísel, jejíž prvky se nazývají složky. Tyto uspořádané n -tice budeme zapisovat do hranatých závorek do řádků nebo sloupců:

$$x = [x_1, x_2, \dots, x_n] \quad \text{nebo} \quad x = \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix}.$$

Stručně píšeme též $x = [x_i]$. Někdy je užitečné zapisovat i -tou složku vektoru x jako $[x]_i$, tedy $x_i = [x]_i$. Pro libovolný skalár α a aritmetický vektor x dále definujeme jejich součin jako

$$\alpha x := [\alpha x_i].$$

Sčítání a odečítání dvou aritmetických vektorů x a y o n složkách je dáno předpisem

$$[x \pm y]_i := [x]_i \pm [y]_i, \quad i = 1, 2, \dots, n$$

a jejich skalární součin definujeme vztahem

$$xy := x_1y_1 + x_2y_2 + \dots + x_ny_n.$$

Uvažujme prvky $a_{11}, a_{12}, \dots, a_{mn}$ z dané množiny $\mathcal{F} \subset \mathbb{R}$. Matice typu

(m, n) je obdélníková tabulka

$$A = \begin{bmatrix} a_{11} & \dots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{m1} & \dots & a_{mn} \end{bmatrix},$$

která má $m \cdot n$ prvků a_{ij} uspořádaných do m řádků r_i^A a n sloupců s_j^A , takže

$$A = \begin{bmatrix} r_1^A \\ \vdots \\ r_m^A \end{bmatrix} = [s_1^A, \dots, s_n^A],$$

$$r_i^A = [a_{i1}, \dots, a_{in}], \quad s_j^A = \begin{bmatrix} a_{1j} \\ \vdots \\ a_{mj} \end{bmatrix}.$$

Stručně píšeme též $A = [a_{ij}]$ a ij -tou složku matice A můžeme alternativně zapsat jako $[A]_{ij}$, tedy $a_{ij} = [A]_{ij}$. Opět pro libovolný skalár α a matici A definujeme jejich součin jako

$$\alpha A := [\alpha a_{ij}].$$

Sčítání a odečítání dvou matic téhož typu je dáno předpisem

$$[A \pm B]_{ij} := [A]_{ij} \pm [B]_{ij}.$$

Nyní si zavedeme operaci násobení matic. Nejdříve ovšem zdefinujeme **součin matice a vektoru**. Součinem matice $A = [a_{ij}]$ typu (m, n) a sloupcového vektoru $x = [x_i]$ o n složkách nazýváme sloupcový vektor y o m složkách definovaný předpisem

$$y = Ax := x_1 s_1^A + \dots + x_n s_n^A.$$

Rozepsáním této definice po složkách dostaneme

$$y_i = [Ax]_i = a_{i1}x_1 + \dots + a_{in}x_n = r_i^A x.$$

Nyní přejdeme k násobení matic. Jestliže A je matice typu (m, ℓ) a B je matice typu (ℓ, n) , pak **součin matic** A a B je matice AB typu (m, n) definovaná předpisem

$$AB := [As_1^B, \dots, As_n^B].$$

Rozepíšeme-li si definici násobení matic po složkách, dostaneme

$$[AB]_{ij} = a_{i1}b_{1j} + \dots + a_{il}b_{lj} = r_i^A s_j^B.$$

Nakonec si zavedeme inverzní matici. Necht' A je čtvercová matice (tzn. jde o matici, která má stejný počet řádků a sloupců). Jestliže existuje matice B tak, že

$$AB = BA = I,^1$$

pak se matice B nazývá inverzní maticí k matici A a značí se A^{-1} . Čtvercová matice, ke které existuje inverzní matice, se nazývá regulární. V opačném případě takovou matici nazýváme singulární. Dá se ukázat, že ke každé regulární matici existuje právě jedna inverzní matice.

Mnohem více detailů o vektorech, maticích a operacích s nimi lze nalézt například v [5].

2.2 Jednoduchý model vyhledávání v databázích

V této kapitole si ukážeme jednoduchou ilustraci toho, jak lze pomocí vektorů a jejich skalárních součinů prohledávat databáze.

Nejprve si připomeňme, že pro skalární součin dvou aritmetických vektorů $x = [x_i]$ a $y = [y_i]$ o n složkách platí vztah

$$xy = \|x\| \|y\| \cos \varphi,$$

kde

$$\|x\| := \sqrt{xx} = \sqrt{x_1^2 + x_2^2 + \dots + x_n^2}$$

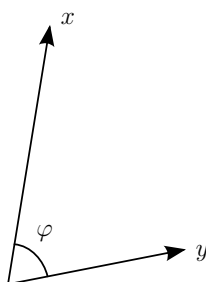
je velikost vektoru x (analogicky definujeme pro y) a φ je úhel, který svírají vektory x a y (viz obr. 2.1).²

Nyní si uvedeme jednu z možných definicí blízkosti vektorů. O dvou aritmetických vektorech řekneme, že jsou si „blízké“, pokud je jejich úhel φ blízký 0, tj. hodnota $\cos \varphi$ je blízká 1. Např. na obr. 2.2 je vektor x „blízký“ y a vektor z „není blízký“ ani x ani y .

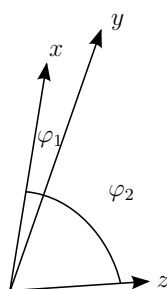
¹Matici I říkáme jednotková matice a definujeme ji takto

$$I := \begin{bmatrix} 1 & 0 & 0 & \dots & 0 & 0 \\ 0 & 1 & 0 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & 0 & 1 \end{bmatrix}.$$

²Úhel φ tedy leží v intervalu $\langle 0, \pi \rangle$.



Obrázek 2.1: Úhel φ mezi vektory x a y



Obrázek 2.2: „Blížkost“ vektorů

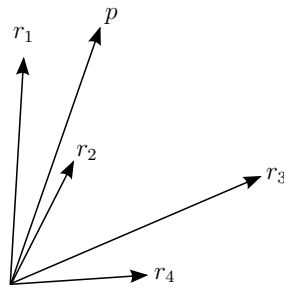
Právě blízkosti vektorů můžeme využít při vyhledávání v databázích. Ilustrujme si to na následujícím příkladě. Mějme kuchařskou knihu, která obsahuje m receptů, kde každý recept obsahuje některou z k surovin. Takovou databázi receptů můžeme popsat m vektory r_1, r_2, \dots, r_m o k složkách. Každý vektor r_i bude obsahovat jen nuly a jedničky (0, resp. 1 na dané pozici v případě, že danou ingredienci neobsahuje, resp. obsahuje). Chceme-li vybrat recept se surovinami, které nejlépe splňují naše požadovaná kritéria, pak můžeme náš požadavek zapsat jako vektor p o k složkách a naším úkolem bude najít vektor r_i z vektorů r_1, r_2, \dots, r_m , který je nejblíže vektoru p , tj. platí, že

$$\cos \varphi_i = \frac{r_i p}{\|r_i\| \|p\|}$$

je největší ze všech

$$\cos \varphi_\ell = \frac{r_\ell p}{\|r_\ell\| \|p\|}, \quad \ell = 1, 2, \dots, m.$$

Na obr. 2.3 ilustrujeme databázi čtyř receptů, přičemž danému požadavku p na suroviny je nejblíže recept r_2 . Vektor p našeho požadavku na suroviny musí mít stejný „formát“ jako naše recepty. Zejména si uvědomme, že 0 na dané pozici vektoru p znamená, že požadujeme, aby recept danou surovinu



Obrázek 2.3: Vektorový model receptů a daný požadavek

neobsahoval. Tato skutečnost nám často může překážet, a proto je vhodné provést „projekci“ receptů (a vektoru p) jen do těch složek, ve kterých je p nenulový.³ Ukážeme si to na příkladu jednoduché databáze tří receptů

$$r_1 = [1, 1, 1, 0, 0], \quad r_2 = [1, 0, 1, 1, 1], \quad r_3 = [1, 0, 0, 1, 1],$$

kde jednotlivé složky postupně odpovídají ingrediencím kuře, vepřové, česnek, cibule, mrkev. Nyní nás bude zajímat, které recepty obsahují kuře a cibuli. Náš požadavek můžeme zapsat jako

$$p = [1, 0, 0, 1, 0],$$

tím ale říkáme, že nechceme, aby recept obsahoval vepřové, česnek a mrkev. Ve skutečnosti nás však obsah ostatních ingrediencí nezajímá a provedeme proto „projekci“ na první a čtvrtou složku⁴

$$r_1 = [1, 0], \quad r_2 = [1, 1], \quad r_3 = [1, 1], \quad p = [1, 1]$$

a pomocí kosínů úhlů mezi „projektovaným“ p a „projektovanými“ recepty zjistíme, že našemu požadavku (na obsah kuřete a cibule) jsou neblíže recepty 2 a 3.

Příklad 1 Popište následující recepty pomocí aritmetických vektorů:

1. Buchty - ingredience: 100 g Hery, 100 g cukru moučka, 2 vejce, 500 g hladké mouky, 1/4 l mléka, 30 g droždí
2. Čínské placičky - ingredience: 3 silnější kuřecí řízky, 3 vejce, 1 drobně nakrájená cibule, 4 lžíce Solamylu, sůl, 4 lžíce oleje, 1 lžíce sojové omáčky, 1 lžíce worcesterové omáčky
3. Čočková polévka - ingredience: 1 lžíce olivového oleje, 1 mrkev, 1 cibule, 4 stroužky česneku, 50 g žitné mouky, 450 g čočky, sůl, 1 kostka zeleninového bujónu, podle chuti chilli nebo cayenský pepř, majoránka, 10 g másla, 2 lžíce plnotučné hořčice, 2 vejce, petrželka

³To, zda „projekci“ provedeme či ne, tedy může ovlivnit výsledek vyhledávání.

⁴Tedy na ty složky, ve kterých je požadavek p nenulový.

4. Domácí buchty - ingredience: 600 g polohrubé mouky, 1/4 l mléka, 2 vejce, 10 lžic oleje, 40 g cukru, 1 prášek do pečiva, 1 kostička droždí (42 g), špetka soli
5. Evíková mňamka - ingredience: 4 kuřecí řízky, 8 plátků anglické slaniny, sýr podle chuti (eidam, blaťácké zlato, hermelín, niva..., ale vždy jen jeden druh), 1 cibule, sůl, pepř, 50 g másla, 1 lžíce sojové omáčky
6. Chléb - ingredience: 1,5 lžíce octa, 3 lžíce olivového oleje, 10 g cukru, 3 lžičky soli, 360 g hladké mouky pšeničné, 140 g žitné mouky, 75 g celozrnné mouky žitné, 75 g celozrnné mouky pšeničné, lžíce kmínu, 15 g sušeného droždí
7. Chlebičková pomazánka - ingredience: 100 g brambor, 1 cibule, 1 lžíce tatarské omáčky, sůl, pepř
8. Kokosová hrníčková bábovka - ingredience: 200 g hladké mouky, 10 lžic kokosu, 100 g cukru krupice, 1/4 l mléka, 6 lžic oleje, 1 prášek do pečiva, 1 vanilkový cukr, 3 vejce
9. Kuřecí kousky v sýrovém těstíčku - ingredience: 4 kuřecích prsíček, sůl
Těstíčko: 1 vejce, 0,05 l bílého vína, 80 g hladké mouky, strouhaný sýr
Dresink: 100 g bílého jogurtu, 5 lžic tatarské omáčky, mletý bílý pepř, sůl, 2 stroužky česneku
10. Pařížské kostky - ingredience: 440 g polohrubé mouky, 220 g cukru, 7 lžic oleje, 1/4 l vlažného mléka, 2-3 lžíce kakaa, 1 prášek do pečiva, 3 celá vejce
11. Perník - ingredience: 1/2 kg polohrubé mouky, 350 g cukru krystal, 1/2 l mléka, 10 lžic oleje, 2 celá vejce, 4 lžíce rozředěných povidel, 2-3 lžíce kakaa, 1 lžička jedlé sody, 1 prášek do pečiva, špetka soli, 1 lžička mleté skořice, lze přidat nejemno nastrohaná 2-3 jablka
12. Pizza těsto - ingredience: 0,5 kg hladké mouky, 1 lžíce olivového oleje, 1 lžíce soli, 15 g droždí
13. Plněné kuře - ingredience: 1 kuře, 1 velká cibule, 3 plátky anglické slaniny, 2 lžíce oleje, sůl
14. Plněný cop - ingredience: 500 g polohrubé mouky, 50 g moučkového cukru, 3 žloutky, 100 g rozpuštěného másla, špetka soli, 0,2 l mléka, 40 g droždí
Náplň: 5 lžic strouhaných lískových oříšků, 100 g cukru, 3 vejce, špetka skořice, vejce na potření, sekané oříšky na posypání
15. Tatranské pracny - ingredience: 300 g hladké mouky, 250 g Hery, 100 g moučkového cukru, 1 vejce, 1 lžíce kakaa, 1 lžička skořice, 6 lžic mletých ořechů, oříšků nebo mandlí (může se i namíchat)
16. Tradiční italské lasagne - ingredience: Boloňská omáčka: 1 lžíce olivového oleje, 1 střední cibule, 1 stroužek česneku, 500 g mletého hovězího (kuřecího, sójového masa, jaké kdo má rád), 250 g oloupaných celých či nakrájených rajčat v plechovce, 2 lžíce rajčatového protlaku, 4 lžíce červeného vína, sůl, čerstvě namletý pepř
Sýrová omáčka: 50 g másla, 50 g mouky, 0,6 l mléka, sýr na strouhání (čedar), 12 listů vaječných těstovin - lasagní
17. Vepřová kýta - ingredience: 6 řízků z vepřové kýty, 1 lžička soli, 1 lžička pepře, 1/2 lžíce solamylu, 1 lžíce worchestrové omáčky, 2 stroužky česneku, 1/2 lžičky majoránky, 2 vejce, 1 lžíce rajčatového protlaku, 1 lžíce oleje

Nyní zkuste na základě výše popsané blízkosti vektorů vyhledat, které recepty obsahují ingredience nejlépe odpovídající těmto požadavkům:

- 1) cukr, droždí, Hera, kakao, mouka a skořice,
- 2) sůl, sýr a worchester,
- 3) česnek, hořčice, kuřecí maso, majoránka, pepř, slanina a sůl.

Příklad 2 Vyhledejte recepty z naší databáze, které neobsahují

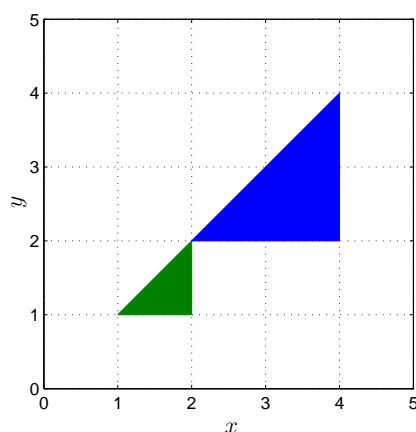
- 1) mléko, jogurt, máslo, sýr ani tatarskou omáčku (recepty pro pacienty s alergií na bílkovinu kravského mléka),
- 2) žádné maso (hovězí, kuřecí, vepřové) ani slaninu (vegetariánské recepty).

2.3 Geometrické transformace

Nejdříve si připomeneme některé pojmy z oblasti geometrických zobrazení. Geometrickým zobrazením rozumíme zobrazení, které geometrickému útvaru U přiřadí geometrický útvar U' . Útvar U je tzv. **vzor** a útvar U' se označuje jako **obraz**. Podívejme se blíže na tři konkrétní typy zobrazení, a to stejnoolehlost, rotaci a posunutí.

Stejnolehlost

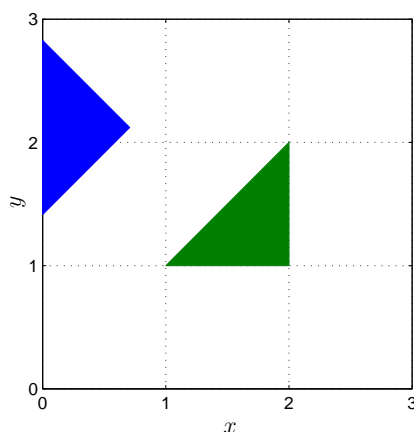
Uvažujme bod $S \in \mathbb{R}^n$, kde $n \in \{2, 3\}$. Geometrické zobrazení, při němž obrazem bodu S je bod S a obrazem každého bodu $A \in \mathbb{R}^n$, $A \neq S$, je takový bod $A' \in \mathbb{R}^n$, že pro vektor SA' platí $SA' = \kappa SA$, kde $\kappa \in \mathbb{R} \setminus \{0, 1\}$ je pevně zvolené, se nazývá stejnolehlostí (nebo také homotetií). Bod S nazýváme **středem stejnoolehlosti** a κ **koefficientem stejnoolehlosti**. Stejnolehlost s $\kappa = -1$ je středovou souměrností.



Obrázek 2.4: Stejnolehlost (v rovině) s koeficientem $\kappa = 2$ a středem v počátku (vzor zeleně, obraz modře)

Rotace

Rotace (otočení) v rovině je geometrické zobrazení, které je charakterizováno tím, že spojnice všech bodů s pevně zvoleným bodem S , tzv. **středem otočení**, se změní o stejný úhel φ , tzv. **úhel otočení**, a vzdálenost bodů od středu otočení zůstává nezměněna.



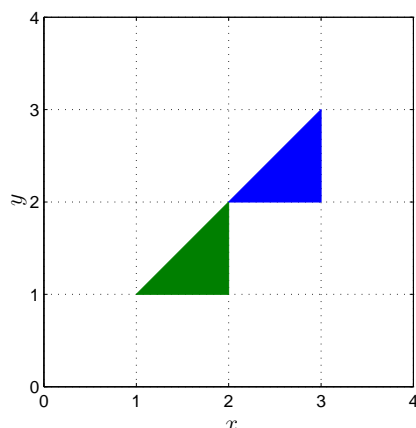
Obrázek 2.5: Rotace s úhlem $\varphi = \pi/4$ a středem v počátku (vzor zeleně, obraz modře)

Posunutí

Posunutí (translace) je geometrické zobrazení, které je charakterizováno tím, že všechny body transformované množiny bodů změní své kartézské souřadnice o stejnou hodnotu, tj. ke každému bodu přičteme stejný **vektor posunutí**.

Nyní si ukážeme, jak lze pomocí elementárních maticových operací zapsat výše uvedená zobrazení. Zapišme souřadnice vzoru geometrického zobrazení, tj. bodu z \mathbb{R}^2 či \mathbb{R}^3 , do sloupcového vektoru. Pokud vzorem zobrazení je více bodů, zapišeme je jako sloupce matice P . Obraz bodů ve stejnolehlosti s koeficientem κ a středem v počátku zapišeme jako součin transformační matice T typu (n, n) a matice P , přičemž n je dimenze prostoru, ve kterém zobrazujeme (tj. 2 nebo 3). Matice T má všechny prvky na hlavní diagonále rovny koeficientu stejnolehlosti κ . Všechny další prvky jsou nulové.⁵ Pro $n = 2$

⁵Tedy $T = \kappa I$, kde I je jednotková matice.



Obrázek 2.6: Posunutí (v rovině) o vektor $[1, 1]$ (vzor zeleně, obraz modře)

zapišeme obrazy bodů v rotaci s úhlem otočení φ a středem otočení v počátku jako součin transformační matice R a matice P , přičemž

$$R = \begin{bmatrix} \cos \varphi & -\sin \varphi \\ \sin \varphi & \cos \varphi \end{bmatrix}.$$

Posunutí s vektorem posunutí p realizujeme tak, že ke každému sloupci matice P přičteme sloupcový vektor p .

Příklad 1 Implementujte geometrická zobrazení stejnolehlost, rotaci a posunutí v rovině. Nalezněte obraz trojúhelníku s vrcholy $[1, -1]$, $[2, 0]$ a $[1, 1]$ ve stejnolehlosti se středem v počátku a s koeficientem stejnolehlosti $\kappa = 2$. Tento obraz transformujte rotací se středem otočení v počátku a úhlem otočení $\varphi = \pi/2$. Nakonec na tento nový obraz aplikujte posunutí s vektorem posunutí $p = [-3, -2]$. Zobrazte vzor (tj. původní trojúhelník) a jednotlivé obrazy (použijte např. matlabovskou funkci `fill`).

Příklad 2 Vytvořte funkci s názvem `transformace`, která transformuje zadaný bod podle vstupních parametrů:

```
transformovany_bod = transformace(bod,  $\kappa$ ,  $\varphi$ ,  $p$ ).
```

Transformaci budeme chápat ve smyslu příkladu 1, tj. nejdříve aplikujeme stejnolehlost se středem v počátku a s koeficientem κ , poté rotaci se středem v počátku a úhlem otočení φ a nakonec posunutí s vektorem posunutí p . Tuto funkci použijte pro vytvoření rutiny, která transformuje n -úhelník zadaný maticí P (sloupce matice jsou tvořeny vektory souřadnic vrcholů).

Příklad 3 Vytvořte rutinu, která v každé iteraci s danou pravděpodobností provede transformaci bodu z předchozí iterace podle tabulky 2.1. Transformovaný bod v každé iteraci vykreslete. Jako počáteční bod volte $[0,5,0,5]$. (Tento příklad byl převzat a upraven z [3].)

pravděpodobnost	κ (stejnolehlost)	φ (rotace)	p (posunutí)
0,85	0,85	-0,05	$[0, 1,5]$
0,07	0,3	1	$[0, 1,5]$
0,07	0,25	-1	$[0, 0,5]$
0,01	transformace definovaná maticí $\begin{bmatrix} 0 & 0 \\ 0 & 0,15 \end{bmatrix}$		

Tabulka 2.1: Krok iteračně definovaného geometrického zobrazení

Kapitola 3

Počáteční úlohy

3.1 Stručný úvod do počátečních úloh a jak je řešit

Na úvod se naučíme pracovat se speciálním typem rovnice – s tzv. diferenciální rovnicí. Pomocí diferenciálních rovnic lze popsat celou řadu zákonitostí, které se objevují v přírodních a společenských vědách. Podrobně se tomuto typu rovnic věnuje text [8]. V této části si také ukážeme, jak lze diferenciální rovnice numericky řešit. A v následujících podkapitolách budeme pomocí diferenciálních rovnic konstruovat modely popisující některé jevy, se kterými se běžně setkáváme. Na závěr poznamenejme, že mnoho dalších zajímavých aplikací obyčejných diferenciálních rovnic lze nalézt v [8, 9].

Obyčejnou diferenciální rovnicí 1. řádu rozumíme rovnici tvaru

$$y'(t) = f(t, y(t)), \quad (3.1)$$

kde $f: \mathbb{R}^2 \mapsto \mathbb{R}$ je zadaná funkce a y je „hledaná“ funkce.

Řešením této rovnice na otevřeném intervalu (a, b) ($a < b$) rozumíme každou funkci $\bar{y}: (a, b) \mapsto \mathbb{R}$ takovou, že pro všechna $t \in (a, b)$ platí

$$\bar{y}'(t) = f(t, \bar{y}(t)).$$

Například funkce $\bar{y}(t) := t$, $t \in (0, +\infty)$, je řešením diferenciální rovnice $y'(t) = \frac{y(t)}{t}$ na intervalu $(0, +\infty)$. Jiným řešením této rovnice na intervalu $(0, +\infty)$ je například funkce $\bar{y}(t) := 2t$, $t \in (0, +\infty)$, či $\bar{y}(t) := 3t$, $t \in (0, +\infty)$. Není těžké ukázat, že pro libovolné $k \in \mathbb{R}$ je funkce $\bar{y}_k(t) := kt$, $t \in (0, +\infty)$, řešením diferenciální rovnice $y'(t) = \frac{y(t)}{t}$ na intervalu $(0, +\infty)$. Naše úloha má nekonečně mnoho řešení. Zkusme navíc přidat k naší rovnici

například podmínku $y(1) = 2$, tj. chceme nalézt funkci, která řeší naši rovnici a navíc její funkční hodnota v $t = 1$ je rovna 2. Lze ukázat, že taková úloha má na $(0, +\infty)$ pouze jediné řešení $\bar{y}(t) = 2t$.

Úlohu, která se skládá z hledání řešení obyčejné diferenciální rovnice 1. řádu, které má navíc splňovat tzv. počáteční podmínku $y(t_0) = y_0$, nazýváme Cauchyovou úlohou¹ a zapisujeme ji obecně takto:

$$\left. \begin{aligned} y'(t) &= f(t, y(t)), \\ y(t_0) &= y_0. \end{aligned} \right\} \quad (3.2)$$

Pokud má funkce $f(t, y(t))$ speciální tvar,² pak existují metody, jak analyticky řešit výše popsanou Cauchyovu úlohu. Často však analytické řešení nalézt nelze nebo by jeho nalezení bylo příliš náročné. V takovém případě se nabízí použití některé z numerických metod pro přibližné řešení Cauchyových úloh. Podívejme se nyní na jednu z těchto metod – Eulerovu metodu.³ O funkci f budeme dále předpokládat, že je v množině $D := \{(t, y) \in \mathbb{R}^2 : t_0 \leq t\}$ spojitá a také jsou v D spojitě derivace této funkce podle proměnné y .

Eulerova metoda

Eulerova metoda je nejjednodušším způsobem numerického řešení Cauchyových úloh. Výstup Eulerovy metody nám aproximuje řešení Cauchyovy úlohy (3.2) na intervalu $\langle t_0, t_N \rangle$ (tento interval specifikujeme níže). Metoda využívá aproximace derivace⁴ funkce y v bodě t pomocí tzv. dopředné difference y v tomto bodě

$$y'(t) \approx \frac{y(t+h) - y(t)}{h},$$

kde h je „malé“ a kladné.

Po jednoduché úpravě dostaneme

$$y(t+h) \approx y(t) + hy'(t).$$

Nahradíme-li funkci $y'(t)$ funkcí $f(t, y(t))$ (viz diferenciální rovnici úlohy (3.2)), získáme vztah

$$y(t+h) \approx y(t) + hf(t, y(t)). \quad (3.3)$$

¹Pojmenována podle francouzského matematika Augustina Louise Cauchyho (1789–1857).

²Například f závisí lineárně na y , tj. $f(t, y) = a(t)y + b(t)$, kde a a b jsou reálné funkce.

³Publikoval ji významný švýcarský matematik a fyzik Leonhard Euler (1707–1783) v roce 1768.

⁴Pro jistotu připomeňme, že derivace funkce y v bodě t je definována jako $\lim_{h \rightarrow 0} \frac{y(t+h) - y(t)}{h}$.

Dále zvolme „dostatečně malé“ pevné $h > 0$ a sestrojme posloupnost

$$t_0, t_1 := t_0 + h, t_2 := t_0 + 2h, \dots, t_N := t_0 + Nh.$$

Označme pomocí y_n aproximaci hodnoty přesného řešení $y(t_n)$. Po dodání počáteční podmínky dostáváme z (3.3) rekurzivní vztah

$$\left. \begin{aligned} y_0 &= y(t_0), \\ y_{n+1} &= y_n + hf(t_n, y_n), \quad n = 0, 1, \dots, N-1, \end{aligned} \right\} \quad (3.4)$$

který použijeme pro numerické řešení Cauchyovy úlohy (3.2). Dá se ukázat, že odhad chyby aproximace řešení Cauchyovy úlohy pomocí Eulerovy metody v bodě t_1 je přímo úměrný druhé mocnině kroku h . Odhad chyby aproximace řešení Cauchyovy úlohy v bodě t_N je přímo úměrný kroku h . Platí totiž, že absolutní hodnotu chyby aproximace řešení Cauchyovy úlohy v bodě t_N můžeme odhadnout shora výrazem

$$\frac{M}{2K} (e^{K(t_N-t_0)} - 1) h \quad (3.5)$$

(tento odhad je možné použít jen při splnění podmínky $Kh < 1$), kde

$$K := \max_{t \in \langle t_0, t_N \rangle, y \in \langle y_0, y_N \rangle} \left| \frac{\partial f}{\partial y}(t, y) \right|$$

a

$$M := \max_{t \in \langle t_0, t_N \rangle, y \in \langle y_0, y_N \rangle} \left| \left(\frac{\partial f}{\partial t} + f \frac{\partial f}{\partial y} \right) (t, y) \right|. ^5$$

3.2 Modelování změny koncentrace léků v krvi

V této části se pokusíme vytvořit matematický model, který popisuje, jakým způsobem se mění koncentrace léku v krvi v závislosti na čase. Nechť funkce $c(t) \geq 0$ udává okamžitou koncentraci látky (vhodného léku) v krvi v čase t (v $\mu g/ml$). Lékařskými pokusy bylo zjištěno, že pokles koncentrace této látky v krvi je přímo úměrný její samotné koncentraci, tj. platí:

$$c'(t) = -kc(t), ^6$$

⁵Symboly $\frac{\partial f}{\partial y}$ a $\frac{\partial f}{\partial t}$ označují parciální derivaci funkce f podle proměnné y , resp. t . O parciálních derivacích se více dozvíte v kapitole 5.1.

⁶Uvědomme si, že koncentrace léku v krvi klesá. Proto změna koncentrace léku popsaná funkcí c' je záporná.

kde $k > 0$ je konstanta.⁷ Tato konstanta popisující úbytek dané látky je určena dvěma farmakokinetickými parametry, kterými jsou **clearance** (míra schopnosti organismu eliminovat látku) a **distribuční objem** (míra kapacity zdánlivého prostoru, který je v organismu pro tuto látku k dispozici). Clearance budeme dále značit cl a její jednotky budou ml/min , distribuční objem označíme v_d a jeho jednotky budou l . Pokud zvolíme jako jednotku času hodiny, můžeme konstantu k popsat následující závislostí

$$k = \frac{cl \frac{60}{1000}}{v_d}.$$

Dále pro jednoduchost předpokládejme, že látka je distribuována do krve intravenózní injekcí a rozšiřuje se do krve okamžitě. Předpokládejme, že tímto způsobem bylo v čase $t = 0$ dodáno do krve takové množství látky, že její koncentrace v krvi měla hodnotu c_0 . Tím jsme získali jednoduchý model, popisující hodnoty koncentrace látky v krvi po její intravenózní aplikaci:

$$\left. \begin{aligned} c'(t) &= -kc(t), \\ c(0) &= c_0. \end{aligned} \right\} \quad (3.6)$$

U léčiv je dalším významným farmakokinetickým parametrem **účinná koncentrace**. Ta udává hodnotu koncentrace či interval hodnot koncentrací, při kterých látka působí prospěšně na organismus. Pokud známe hodnotu výše zmíněných farmakokinetických parametrů pro konkrétní léčivo, můžeme využít řešení úlohy (3.6) pro odpověď na otázku, jak často je třeba lék obsahující tuto látku pacientovi aplikovat pro zajištění úspěšné léčby. V následujících příkladech předpokládáme, že pacientem je průměrná osoba s tělesnou hmotností 70 kg. Hodnoty farmakokinetických parametrů pro léčiva z těchto příkladů byly převzaty z [6], kde se lze seznámit s oblastí farmakokinetiky daleko detailněji.

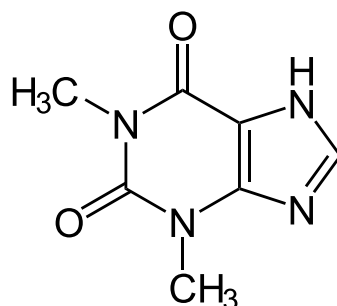
Příklad 1 Pro léčbu nemocného s průduškovým astmatem se používá theofylin. Jeho farmakokinetické parametry jsou:

$$cl = 48 \text{ ml/min}, \quad v_d = 35 \text{ l},$$

účinná koncentrace se pohybuje v intervalu $\langle 10, 20 \rangle \mu g/ml$.

Toto léčivo musíme pacientovi podávat v pravidelných časových intervalech tak, aby jeho koncentrace v krvi léčené osoby nepřesáhla horní mez účinné koncentrace a neklesla pod její dolní mez. Na začátku léčby byla aplikována

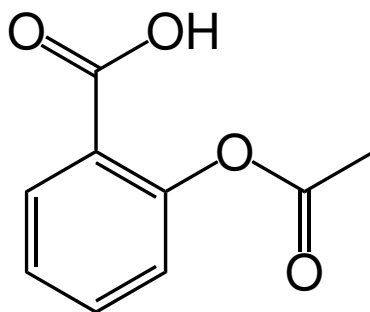
⁷Hodnota k závisí na léku a pacientovi.



Obrázek 3.1: Molekula theofylinu

zaváděcí dávka, která způsobila, že v čase $t = 0$ byla koncentrace theofylinu v krvi $c_0 = 20 \mu\text{g}/\text{ml}$. Další dávky chceme aplikovat vzhledem k pohodlí pacienta tak, aby interval podávání léku byl co největší. Zjistěte, po jakém čase T (v hodinách) je nutné lék obsahující theofylin znovu podat pacientovi. Připomeňme, že aby byla léčba účinná, je třeba lék pacientovi podat dříve, než koncentrace theofylinu klesne pod dolní mez účinné koncentrace. Odhadněte chybu aproximace koncentrace léku v čase T .

Příklad 2 Pro léčbu horečnatých stavů, bolesti hlavy, svalů či kloubů se používá kyselina acetylsalicylová.⁸ Její farmakokinetické parametry jsou:



Obrázek 3.2: Molekula kyseliny acetylsalicylové

$$cl = 650 \text{ ml}/\text{min}, \quad v_d = 11 \text{ l},$$

účinná koncentrace se pohybuje v intervalu $\langle 150, 300 \rangle \mu\text{g}/\text{ml}$.

Toto léčivo musíme pacientovi podávat v pravidelných časových intervalech tak, aby jeho koncentrace v krvi léčené osoby nepřesáhla horní mez účinné koncentrace a neklesla pod její dolní mez. Na začátku léčby byla aplikována

⁸Kyselina acetylsalicylová je hlavní složkou léků jako je Aspirin, Acylpirin či Anopyrin.

zaváděcí dávka, která způsobila, že v čase $t = 0$ byla koncentrace kyseliny acetylsalicylové v krvi $c_0 = 300 \mu\text{g}/\text{ml}$. Další dávky chceme aplikovat vzhledem k pohodlí pacienta tak, aby interval podávání léku byl co největší. Zjistěte, po jakém čase T (v hodinách) je nutné lék obsahující kyselinu acetylsalicylovou znovu podat pacientovi. Aby byla léčba účinná, je třeba lék pacientovi podat dříve, než koncentrace kyseliny acetylsalicylové klesne pod dolní mez účinné koncentrace. Odhadněte chybu aproximace koncentrace léku v čase T .

3.3 Tak je to padělek nebo to není padělek aneb jak poznat stáří některých „Vermeerových“ obrazů?

V této části se pokusíme vytvořit matematický model, který popisuje rozpad atomů radioaktivních prvků. Než se začneme zabývat tvorbou dalšího matematického modelu, vrátíme se v čase do doby krátce po konci druhé světové války. Těsně po válce zjistila nizozemská policie, že během války bylo prodáno několik Vermeerových⁹ obrazů německému ministrovi letectví Hermannu Göringovi. Tuto transakci zprostředkoval nizozemský malíř Han van Meegeren. Na základě těchto zjištěných faktů byl 29. 5. 1945 van Meegeren zadržen a obviněn z kolaborace s nepřítelem. 12. 7. 1945 van Meegeren vydal prohlášení, že Göringovi nikdy žádný Vermeerův obraz neprodal. Naopak Göringa napálil, protože obrazy, které mu prodal, jsou podvrhy Vermeerových obrazů a sám je vytvořil.

A aby dokázal své tvrzení, začal jeden z „Vermeerových“ obrazů¹⁰ napodobovat. Van Meegeren přizvaným znalčům předvedl způsob, jakým vytváří barvy, jak připravuje plátno, či jak zařídí, aby povrch malby vypadal jako u několik set let starého obrazu. Těsně před dokončením podvrhu Vermeerova obrazu se van Meegeren dozvěděl, že obvinění z kolaborace bude nahrazeno obviněním z padělatelství, a tak odmítl tuto kopii dokončit. I tak ale většina přizvaných odborníků uznala, že obrazy prodané Göringovi jsou pravděpodobně falzum a van Meegeren byl 12. 11. 1947 odsouzen za padělatelství na rok do vězení, ve kterém 30. 12. 1947 na infarkt zemřel.

I přesto, že komise, která posuzovala pravost „Vermeerových“ obrazů uznala, že to jsou pravděpodobně podvrhy vytvořené van Meegerenem, zůstávali odborníci u některých obrazů, k jejichž autorství se také van Meegeren přihlásil, na pochybách. Zejména zpochybňování pravosti obrazu Emauzští učedníci, který zakoupilo muzeum v Rotterdamu za 170 000 dolarů, vy-

⁹Jan Vermeer (1632–1675) byl nizozemský malíř.

¹⁰Konkrétně šlo o obraz Ježíš mezi znalci Písma.

volávalo velké spory. Proto se přistoupilo u tohoto obrazu v roce 1967 k metodě radioaktivního datování, která měla tyto pochyby rozhodnout.

Metoda radioaktivního datování využívá toho, že některé tzv. radioaktivní prvky jsou nestabilní a část jejich atomů se samovolně rozpadá na atomy jiných prvků. Experimenty bylo zjištěno, že rychlost rozpadu atomů radioaktivních prvků je přímo úměrná počtu těchto atomů. Pokud funkci udávající počet atomů radioaktivního prvku v čase t v gramu látky označíme jako $N(t)$, pak výše zmíněnou závislost můžeme popsat diferenciální rovnicí

$$N'(t) = -\lambda N(t),^{11} \quad (3.7)$$

kde λ je konstanta, která popisuje rychlost rozpadu atomů daného radioaktivního prvku. Tato konstanta je dána pro každý radioaktivní prvek tímto vztahem

$$\lambda = \frac{\ln 2}{\text{poločas rozpadu prvku v rocích}}.$$

Čas t v našem modelu budeme měřit v rocích a jednotkou konstanty λ je rok^{-1} .

Metoda radioaktivního datování je založena na jednoduchém pozorování. Pokud bychom věděli, kolik atomů radioaktivního prvku měla látka v jednom svém gramu při svém vzniku (tzn. známe hodnotu N_0 , pro kterou platí $N(0) = N_0$), a znali bychom také aktuální počet těchto atomů v gramu látky, mohli bychom řešením úlohy

$$\left. \begin{array}{l} N'(t) = -\lambda N(t), \\ N(0) = N_0 \end{array} \right\} \quad (3.8)$$

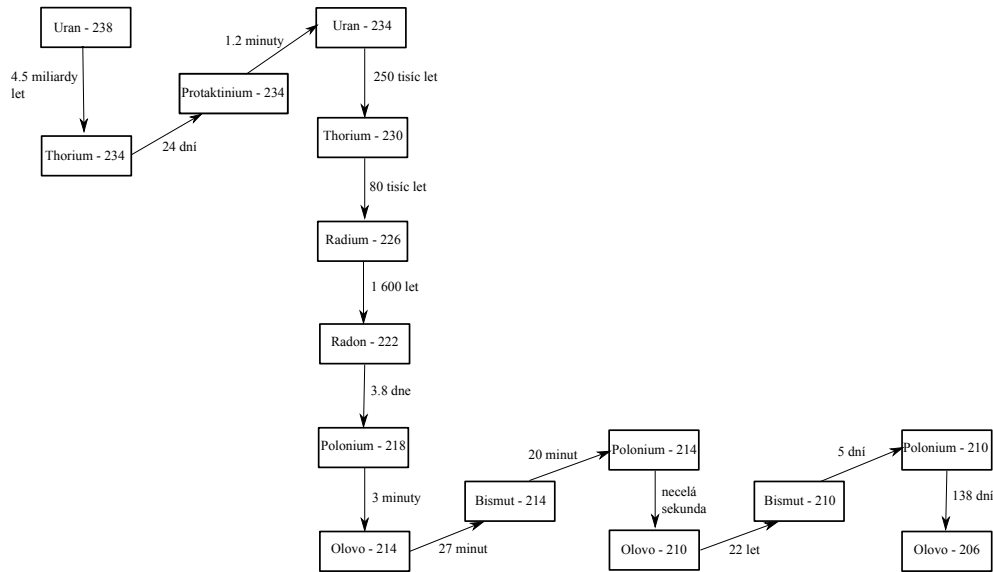
zjistit, jak je tato látka stará.

Než se začneme zabývat datováním „Vermeerových“ obrazů, uvědomme si, že všechny horniny na Zemi obsahují malé množství radioaktivního uranu, který se rozpadá na atomy dalšího prvku. Tyto atomy se opět samovolně mění na další atomy atd. Viz obr. 3.3 (časy u šipek udávají poločasy rozpadu¹² jednotlivých radioaktivních prvků).

Dále je známo, že olovnatá běloba používaná na malbách obsahuje oxid olovnatý, který obsahuje malé množství olova-210 a ještě menší množství radia-226. V okamžiku, kdy je barva obsahující oxid olovnatý vyrobena, začnou se atomy olova-210 velmi rychle rozpadat s poločasem rozpadu 22

¹¹Uvědomme si, že počet atomů radioaktivního prvku klesá v důsledku jejich samovolného rozpadu. Proto je změna počtu atomů popsána funkcí N' záporná.

¹²Poločas rozpadu je doba, za kterou počet atomových jader ve vzorku klesne na polovinu.



Obrázek 3.3: Uranová rozpadová řada

let a množství olova-210 v této barvě klesá. Na druhé straně vzniká malé množství olova-210 rozpadem radia-226 (a prvků, které následují v rozpadové řadě za ním). Tento proces můžeme popsat následující počáteční úlohou

$$\left. \begin{aligned} N'(t) &= -\lambda N(t) + r(t), \\ N(0) &= N_0, \end{aligned} \right\} \quad (3.9)$$

kde $N(t)$ je funkce udávající počet atomů olova-210 v čase t v gramu látky, $r(t)$ je funkce udávající počet atomů olova-210, které vzniknou v čase t v gramu oxidu olovnatého za rok.

Protože poločas rozpadu radia-226 je 1600 let a metodu radioaktivního datování chceme použít pro rozpoznání stáří obrazů, které měly v roce 1967 přibližně buď 300 let nebo 20 let, můžeme funkci $r(t)$ považovat za konstantní. Pak $r(t) = r = \text{konst.}$ a úlohu (3.9) můžeme nahradit počáteční úlohou

$$\left. \begin{aligned} N'(t) &= -\lambda N(t) + r, \\ N(0) &= N_0. \end{aligned} \right\} \quad (3.10)$$

Mnohem více podrobností o metodě radioaktivního datování může čtenář nalézt v [2].

Také v případě úlohy (3.10) jsme schopni, pokud známe počet atomů olova-210 v gramu oxidu olovnatého v době výroby olovnaté běloby, určit stáří obrazu, na kterém je tato barva použita. K řešení této úlohy můžeme

opět použít Eulerovu metodu, se kterou jste se seznámili v kapitole 3.1. V naší úloze počet atomů olova-210 v gramu oxidu olovnatého v době výroby barvy bohužel neznáme. I přesto jsme schopni rozlišit obraz, jehož stáří je 300 let, od obrazu, který má 20 let. Je totiž známo, jaké bývají koncentrace radioaktivního olova-210 v rudách, ze kterých se vyrábí oxid olovnatý. Je naprosto nemožné, aby počet atomů olova-210 v gramu rudy, ze které se oxid olovnatý vyrobil přesáhl počet $5 \cdot 10^{11}$. Proto můžeme zjistit, pokud známe potřebné parametry, zda je možné, aby bylo stáří obrazu 300 let.

Příklad 1 Určete, zda je možné,¹³ aby byl obraz Emauzští učedníci opravdu starý 300 let a byl tedy pravý, pokud bylo měřením zjištěno, že v čase měření t (tzn. v roce 1967) platí

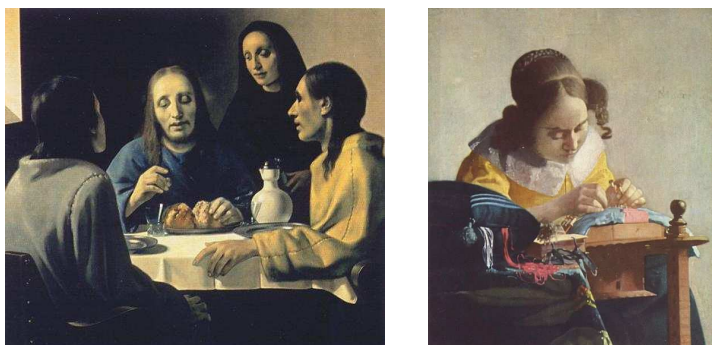
$$N(t) = 1,42 \cdot 10^8, \quad r = 420480.$$

Odhadněte chybu aproximace $N(t)$ v čase vytvoření obrazu.

Příklad 2 Určete, zda je možné, aby byl obraz Krajkářka opravdu starý 300 let a byl tedy pravý, pokud bylo měřením zjištěno, že v čase měření t (tzn. v roce 1967) platí

$$N(t) = 0,25 \cdot 10^8, \quad r = 735840.$$

Odhadněte chybu aproximace $N(t)$ v čase vytvoření obrazu.



Obrázek 3.4: Obrazy Emauzští učedníci a Krajkářka

¹³V tomto i v následujícím příkladě nám může pomoci, pokud pro řešení použijeme tzv. zpětnou diferenci $y'(t) \approx \frac{y(t) - y(t-h)}{h}$, kde h je „malé“ a kladné.

3.4 Populační modely

Nyní se pokusíme o odhad vývoje počtu obyvatel. K tomu se potřebujeme naučit pracovat s jednoduchými populačními modely. Nechť $y(t)$ označuje velikost populace v čase t a y_0 popisuje velikost populace v čase t_0 . Nechť p je kladná konstanta, která udává přírůstek populace. Pak jednoduchý model vývoje populace nám poskytne následující populační rovnice s počáteční podmínkou

$$\left. \begin{aligned} y'(t) &= py(t), \\ y(t_0) &= y_0. \end{aligned} \right\} \quad (3.11)$$

Řešení úlohy (3.11) poměrně dobře aproximuje vývoj populace, která má dostatečně velké zásoby potravy a dalších zdrojů a může neomezeně růst. Lepší model dává následující populační rovnice s počáteční podmínkou, ve které se navíc objevuje člen $qy^2(t)$ (q je kladná konstanta)

$$\left. \begin{aligned} y'(t) &= py(t) - qy^2(t), \\ y(t_0) &= y_0. \end{aligned} \right\} \quad (3.12)$$

Řešení úlohy (3.12) dobře aproximuje vývoj populace, která už je dostatečně velká, má omezené zásoby potravy i dalších zdrojů a mezi členy populace dochází k soupeření o tyto zdroje (to popisuje člen $-qy^2(t)$).

Příklad 1 Použijte populační rovnici s počáteční podmínkou (3.12) k modelování vývoje počtu obyvatel USA v letech 1790–1950. Konstanty p , q byly odhadnuty takto: $p = 0,03134$, $q = 1,5887 \cdot 10^{-10}$. Čas t je v rocích. Navíc víte, že počet obyvatel v USA v roce 1790 byl 3 929 000. Spočítané hodnoty můžete porovnat se skutečnými hodnotami v následující tabulce. Odhadněte chybu aproximace počtu obyvatel USA v roce 1950.

Rok	Počet obyvatel
1790	3 929 000
1800	5 308 000
1850	23 192 000
1900	75 995 000
1950	150 697 000

Tabulka 3.1: Populace USA v letech 1790–1950

3.5 Změna teploty tělesa

Nastal čas na šálek kávy. Čerstvě uvařená káva má teplotu T_k a je ochlazovaná vzduchem v místnosti o konstantní teplotě T_v . Je rozumné modelovat teplotu



kávy jako funkci $T_k(t)$, která se mění¹⁴ přímo úměrně rozdílu teploty vzduchu a kávy s koeficientem $k > 0$. Nechť teplota kávy v čase t_0 je dána hodnotou T_{k0} . Dostáváme tedy následující model:

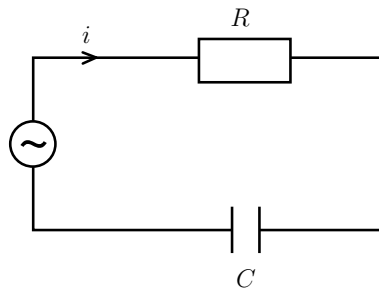
$$\left. \begin{aligned} T_k'(t) &= k(T_v - T_k(t)), \\ T_k(t_0) &= T_{k0}. \end{aligned} \right\} \quad (3.13)$$

Příklad 1 Použijte řešení úlohy (3.13) k modelování ochlazování kávy, která měla v čase $t_0 = 0$ teplotu 100°C . Teplota vzduchu má hodnotu $T_v = 20^\circ\text{C}$ a konstanta $k = 0,04$. Čas t je v minutách. Zjistěte, za jak dlouho se káva ochladí na teplotu nižší než 50°C . Odhadněte chybu aproximace řešení úlohy (3.13) v čase, kdy se káva ochladí na požadovanou teplotu.

3.6 RC obvod

V této kapitole vytvoříme matematický model, který popisuje, jakým způsobem se mění náboj v jednoduchém obvodu s rezistorem a kondenzátorem v závislosti na čase. Uvažujme nyní elektrický obvod, ve kterém je sériově zapojen rezistor a kondenzátor. V obvodu je také v sérii zapojen elektrický zdroj, který do sítě dodává elektromotorické napětí. Schéma obvodu je na obr. 3.5. Nyní vytvoříme jednoduchý model, který popíše hodnotu elektrického náboje q v obvodu v závislosti na čase t . Odpor rezistoru označme R a kapacitu kondenzátoru popíšme parametrem C . Elektrický zdroj dodává do obvodu elektromotorické napětí $E(t)$ (často $E(t) = A \sin(\omega t)$, kde $\omega, A \in \mathbb{R}$).

¹⁴Tuto změnu matematicky popisuje derivace funkce $T_k(t)$.



Obrázek 3.5: Schéma elektrického obvodu

Matematicky lze uvedený systém popsat následující Cauchyovou úlohou

$$\left. \begin{aligned} Rq'(t) + \frac{1}{C}q(t) &= E(t), \\ q(t_0) &= q_0, \end{aligned} \right\} \quad (3.14)$$

kde q_0 označuje počáteční hodnotu elektrického náboje. Protože elektrický proud i v čase t je definován jako okamžitá změna náboje q a tento vztah můžeme matematicky zapsat jako $i(t) = q'(t)$, lze derivováním diferenciální rovnice v (3.14) snadno získat rovnici pro hodnotu elektrického proudu i procházejícího obvodem v závislosti na čase t :

$$Ri'(t) + \frac{1}{C}i(t) = E'(t).$$

Příklad 1 Pomocí úlohy (3.14) popíšeme náboj v elektrickém obvodu, ve kterém je sériově zapojen rezistor s odporem $R = 2 \Omega$, kondenzátor s kapacitou $C = 1 \text{ F}$ a elektrický zdroj, který do sítě dodává elektromotorické napětí $E(t) = 10 \text{ V}$. Hodnota elektrického náboje je na počátku nulová, tj. $q(0) = 0 \text{ C}$. Pomocí Eulerovy metody aproximujte chování elektrického náboje v časovém intervalu $\langle 0, 10 \rangle$ s. Odhadněte chybu aproximace elektrického náboje v čase $t = 10 \text{ s}$.

Příklad 2 Uvažujme obvod bez elektrického zdroje se sériově zapojeným rezistorem a kondenzátorem (se stejnými parametry jako v předchozím příkladě). Kondenzátor je na počátku nabit nábojem o hodnotě 10 C , tj. $q(0) = 10 \text{ C}$. S využitím modelu (3.14) aproximujte pomocí Eulerovy metody chování elektrického náboje v časovém intervalu $\langle 0, 10 \rangle$ s. Odhadněte chybu aproximace elektrického náboje v čase $t = 10 \text{ s}$.

3.7 Matematické kyvadlo

V této části vytvoříme matematický model, který popisuje pohyb matematického kyvadla v závislosti na čase. Abychom mohli takovýto model sestavit, potřebujeme pracovat se speciálním typem rovnice – s tzv. diferenciální rovnicí druhého řádu. Navíc si povíme, jak lze diferenciální rovnici druhého řádu numericky řešit.

Pro práci s dalšími typy diferenciálních rovnic si zavedeme pojem vektorové funkce. **Vektorovou funkcí n reálných proměnných o m složkách** nazýváme každé zobrazení z \mathbb{R}^n do \mathbb{R}^m , $m, n \in \mathbb{N}$. Vektorová funkce $f = (f_1, \dots, f_m)$ n reálných proměnných o m složkách je tedy předpis, který každému

$$x = [x_1, x_2, \dots, x_n] \in Df \subset \mathbb{R}^n$$

přiřadí jednoznačně hodnotu

$$f(x) = [f_1(x_1, x_2, \dots, x_n), f_2(x_1, x_2, \dots, x_n), \dots, f_m(x_1, x_2, \dots, x_n)] \in Hf \subset \mathbb{R}^m,$$

kde $f_i : \mathbb{R}^n \mapsto \mathbb{R}$ pro $i = 1, \dots, m$. Skutečnost, že f je vektorovou funkcí n reálných proměnných, zapisujeme pomocí

$$f : \mathbb{R}^n \mapsto \mathbb{R}^m.$$

Soustava diferenciálních rovnic 1. řádu

V případě, že v rovnici (3.1) jsou funkce $y = (y_1, \dots, y_n)$ a $f = (f_1, \dots, f_n)$ vektorové, tzn., že $y : (a, b) \mapsto \mathbb{R}^n$ a $f : \mathbb{R}^{n+1} \mapsto \mathbb{R}^n$, pak můžeme tuto rovnici přepsat do tvaru

$$\left. \begin{aligned} y_1'(t) &= f_1(t, y_1(t), y_2(t), \dots, y_n(t)), \\ y_2'(t) &= f_2(t, y_1(t), y_2(t), \dots, y_n(t)), \\ &\vdots \\ y_n'(t) &= f_n(t, y_1(t), y_2(t), \dots, y_n(t)), \end{aligned} \right\} \quad (3.15)$$

který nazýváme **soustavou obyčejných diferenciálních rovnic 1. řádu**. Pokud má navíc systém (3.15) splňovat počáteční podmínky

$$y(t_0) = [y_{10}, \dots, y_{n0}] \in \mathbb{R}^n,$$

tzn.

$$y_1(t_0) = y_{10}, \quad y_2(t_0) = y_{20}, \quad \dots, \quad y_n(t_0) = y_{n0},$$

nazveme takovou úlohu opět Cauchyovou úlohou. K získání přibližného řešení takové úlohy lze použít rekursivní vztah (3.4), kde $y_0, y_1, \dots, y_k, \dots$ jsou n -rozměrné vektory a $f : \mathbb{R}^{n+1} \mapsto \mathbb{R}^n$ je vektorová funkce.

Obyčejná diferenciální rovnice n -tého řádu

Obyčejnou diferenciální rovnici n -tého řádu (v explicitním tvaru) rozumíme rovnicí tvaru

$$y^{(n)}(t) = f(t, y(t), y'(t), \dots, y^{(n-1)}(t)), \quad (3.16)$$

kde $f: \mathbb{R}^{n+1} \mapsto \mathbb{R}$ je zadaná funkce. Pokud k rovnici (3.16) přidáme n počátečních podmínek

$$y(t_0) = y_0, \quad y'(t_0) = y_1, \quad \dots, \quad y^{(n-1)}(t_0) = y_{n-1},$$

($y_k \in \mathbb{R}$ pro $k = 0, \dots, n-1$, $t_0 \in \mathbb{R}$), získáme Cauchyovu úlohu pro diferenciální rovnici n -tého řádu. Tuto úlohu můžeme s využitím substituce $z_1(t) = y'(t)$, $z_2(t) = y''(t)$, \dots , $z_{n-1}(t) = y^{(n-1)}(t)$ přepsat na systém

$$\left. \begin{aligned} y'(t) &= z_1(t), \\ z_1'(t) &= z_2(t), \\ &\vdots \\ z_{n-2}'(t) &= z_{n-1}(t), \\ z_{n-1}'(t) &= f(t, y(t), z_1(t), \dots, z_{n-1}(t)) \end{aligned} \right\} \quad (3.17)$$

s počátečními podmínkami

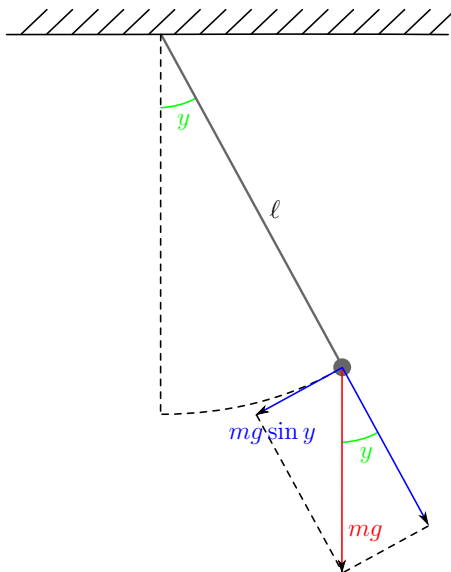
$$y(t_0) = y_0, \quad z_1(t_0) = y_1, \quad \dots, \quad z_{n-1}(t_0) = y_{n-1}.$$

Matematické kyvadlo

Nyní sestavme obyčejnou diferenciální rovnici pro popis matematického kyvadla. Zabývejme se zkoumáním chování hmotného bodu zavěšeného na tenkém, pevném a neprůtažném vlákně zanedbatelné hmotnosti, zanedbává se odpor vzduchu při pohybu kyvadla i tření v závěsu a gravitační pole se považuje za homogenní. Tíhové zrychlení působené tímto polem označíme g . Délka vlákna je ℓ , hmotný bod zavěšený na vlákně má hmotnost m . Náčrtek kyvadla je na obr. 3.6.

Vytvořme matematický model kyvadla popisující jeho výchylku y od rovnovážné polohy v závislosti na čase t . Výchylku y budeme měřit v radiánech. S využitím Newtonova zákona¹⁵ lze uvedený systém po určitém zjed-

¹⁵Přesněji druhého Newtonova pohybového zákona, který je matematicky vyjádřen vztahem $F = ma$.



Obrázek 3.6: Matematické kyvadlo

nodušení¹⁶ popsat následující Cauchyovou úlohou

$$\left. \begin{aligned} y''(t) &= (-g/\ell) y(t), \\ y(t_0) &= y_0, y'(t_0) = dy_0, \end{aligned} \right\} \quad (3.18)$$

kde $y_0 \in \mathbb{R}$ označuje počáteční výchylku závaží (v radiánech) a $dy_0 \in \mathbb{R}$ značí počáteční (úhlovou) rychlost závaží.

Obdobně jako rovnici (3.16) můžeme pomocí substituce přepsat na soustavu (3.17), lze s využitím substituce $z(t) = y'(t)$ transformovat úlohu (3.18) na následující systém:

$$\left. \begin{aligned} y'(t) &= z(t), \\ z'(t) &= (-g/\ell) y(t), \\ y(t_0) &= y_0, z(t_0) = dy_0. \end{aligned} \right\} \quad (3.19)$$

K přibližnému řešení této soustavy můžeme využít dříve uvedenou Eulerovu metodu, tj. vektorový tvar formule (3.4).

Příklad 1 Pomocí úlohy (3.19) popíšeme kyvadlo, jehož délka je $\ell = 1$ m a tíhové zrychlení g považujeme rovné 10 m/s^{-2} . Výchylka kyvadla je na počátku rovna $0,1$ (tj. $y(0) = 0,1$ rad) a počáteční úhlová rychlost kyvadla

¹⁶Toto zjednodušení spočívá v aproximaci $y \approx \sin y$ pro „malé“ y .

je nulová (tj. $z(0) = 0$ rad/s). Pomocí Eulerovy metody aproximujte pohyb kyvadla v časovém intervalu $\langle 0, 10 \rangle$ s. Odhadněte chybu aproximace výchylky kyvadla v čase $t = 10$ s.

Příklad 2 Uvažujme opět matematické kyvadlo (se stejnými parametry a stejným modelem jako v předchozím cvičení). Výchylka kyvadla je na počátku nulová (tj. $y(0) = 0$ rad) a počáteční úhlová rychlost kyvadla je rovna 1 (tj. $z(0) = 1$ rad/s). Pomocí Eulerovy metody aproximujte pohyb kyvadla v časovém intervalu $\langle 0, 10 \rangle$ s. Odhadněte chybu aproximace výchylky kyvadla v čase $t = 10$ s.

Kapitola 4

Okrajové úlohy

4.1 Jednorozměrné (1D) okrajové úlohy a 1D metoda sítí

V této kapitole si ukážeme jednu z numerických metod pro přibližné řešení 1D okrajových úloh, a to tzv. metodu sítí. Okrajové úlohy v jedné dimenzi sestávají z příslušné diferenciální rovnice předepsané na daném otevřeném intervalu a dále z okrajových podmínek předepsaných v krajních bodech tohoto intervalu. V dalším výkladu se omezíme pouze na tzv. Dirichletovu okrajovou úlohu 2. řádu, kdy nejvyšší řád derivace vyskytující se v diferenciální rovnici je roven dvěma a okrajové podmínky jsou předepsány ve tvaru hodnoty hledaného řešení v krajních bodech intervalu.

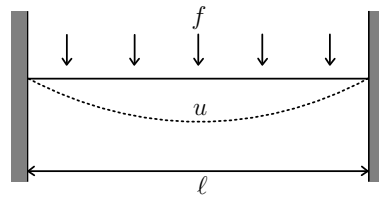
1D Dirichletova úloha

Ať k a ℓ jsou libovolné kladné reálné konstanty a f je reálná spojitá funkce. Uvažujme následující Dirichletovu¹ úlohu: najděte funkci u takovou, že

$$\left. \begin{aligned} -k u''(x) &= f(x) \quad \text{pro } x \in (0, \ell), \\ u(0) &= u(\ell) = 0. \end{aligned} \right\} \quad (4.1)$$

Řešení u tohoto problému si lze představit jako průhyb struny délky ℓ , která je uchycena na obou koncích a na níž příčně působí síla s hustotou f . Konstanta k souvisí s tuhostí struny. Takovouto interpretaci úlohy (4.1) si můžeme prohlédnout na obr. 4.1.

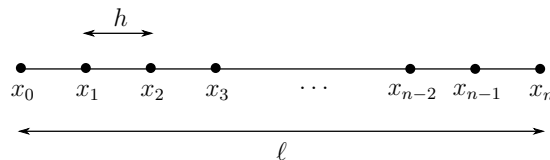
¹Johann Peter Gustav Lejeune Dirichlet (1805–1859) – významný německý matematik



Obrázek 4.1: Průhyb příčně zatížené a na obou koncích uchycené struny

Metoda sítí pro Dirichletovu úlohu (4.1)

Na intervalu $(0, \ell)$ zvolíme pravidelnou síť² uzlů $x_0, x_1, \dots, x_{n-1}, x_n$, $n \in \mathbb{N}$, viz obr. 4.2. Vzdálenost dvou sousedních uzlů vyjádříme číslem $h = \ell/n$. Označíme si dále $f_i = f(x_i)$ pro $i = 1, \dots, n-1$.

Obrázek 4.2: Pravidelná síť uzlů na struně délky ℓ

Nyní se budeme snažit aproximovat hodnoty funkce u v daných uzlech, tj. budou nás zajímat čísla $u_i \approx u(x_i)$:

- počáteční uzel: $u_0 = 0$,
- vnitřní uzly: derivaci druhého řádu vyskytující se v diferenciální rovnici nahradíme diferenčními podíly²

$$\begin{aligned} u''(x_i) &\approx \frac{u'(x_i + \frac{h}{2}) - u'(x_i - \frac{h}{2})}{h} \approx \frac{\frac{u(x_i+h) - u(x_i)}{h} - \frac{u(x_i) - u(x_i-h)}{h}}{h} = \\ &= \frac{u(x_{i+1}) - 2u(x_i) + u(x_{i-1}))}{h^2}; \end{aligned}$$

²V Eulerově metodě jsme na str. 38 použili k nahrazení derivace tzv. dopředné diference. Zde použijeme tzv. centrální diferenci:

$$u'(x) \approx \frac{u(x + \frac{h}{2}) - u(x - \frac{h}{2})}{h},$$

přičemž centrální diference použijeme pro nahrazení druhé i první derivace funkce u v daných uzlech. Výsledná aproximace $u''(x_i)$ pomocí diferencních podílů je přímo úměrná h^2 .

rovnosti $-ku''(x_i) = f(x_i)$ tak nahradíme systémem rovnic

$$\frac{k}{h^2}[-u_{i+1} + 2u_i - u_{i-1}] = f_i, \quad i = 1, 2, \dots, n-1,$$

c) koncový uzel: $u_n = 0$.

Dostáváme tak soustavu $n-1$ lineárních algebraických rovnic o $n-1$ neznámých, kterou si můžeme maticově zapsat takto:

$$\underbrace{\frac{k}{h^2} \begin{bmatrix} 2 & -1 & 0 & \cdots & 0 \\ -1 & 2 & -1 & & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & & -1 & 2 & -1 \\ 0 & \cdots & 0 & -1 & 2 \end{bmatrix}}_{=: K} \underbrace{\begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_{n-2} \\ u_{n-1} \end{bmatrix}}_{=: u} = \underbrace{\begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ f_{n-2} \\ f_{n-1} \end{bmatrix}}_{=: f},$$

zkráceně pak

$$Ku = f.$$

Matice K se často nazývá **maticí tuhosti**. Všimněme si také, že matice K je třídiagonální.³ Poznamenejme ještě, že jelikož metoda sítí využívá vždy k nahrazení derivací vhodné diferenční podíly, říká se jí také často **metoda konečných diferencí**.

Analytické řešení úlohy (4.1) pro konstantní f

Předpokládejme, že $f(x) := c \in \mathbb{R}$ pro všechna $x \in (0, \ell)$ a že u je řešením úlohy (4.1). Pak pro všechna $x \in (0, \ell)$ platí:

$$-ku''(x) = c,$$

$$u''(x) = -\frac{c}{k},$$

$$u'(x) = \int -\frac{c}{k} dx = -\frac{c}{k}x + a, \quad a \in \mathbb{R},$$

$$u(x) = \int -\frac{c}{k}x + a dx = -\frac{c}{2k}x^2 + ax + b, \quad a, b \in \mathbb{R}.$$

Nyní vezměme v úvahu okrajové podmínky a najděme hodnoty konstant a

³Třídiagonální matice má nenulové prvky pouze na hlavní diagonále, první diagonále pod hlavní diagonálou a první diagonále nad hlavní diagonálou.

a b.

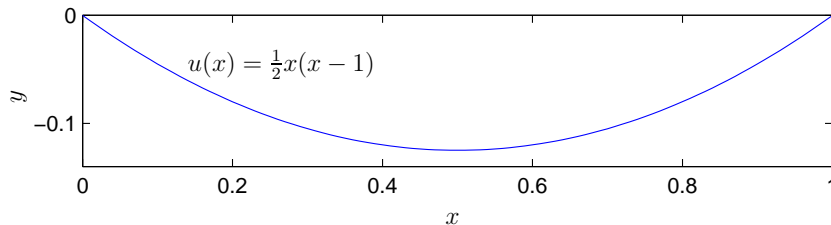
$$u(0) = 0 : \quad 0 = -\frac{c}{2k} \cdot 0^2 + a \cdot 0 + b \Rightarrow b = 0$$

$$u(\ell) = 0 : \quad 0 = -\frac{c}{2k} \cdot \ell^2 + a \cdot \ell \Rightarrow a = \frac{c}{2k} \ell$$

Získali jsme analytické řešení Dirichletovy úlohy (4.1) ve tvaru

$$u(x) = \frac{c}{2k} x(\ell - x) \quad \text{pro } x \in \langle 0, \ell \rangle,$$

viz obr. 4.3.



Obrázek 4.3: Analytické řešení úlohy (4.1) pro volbu $c = -1$, $k = 1$ a $\ell = 1$

Příklad 1 Naimplementujte metodu sítí pro modelování průhybu struny o tuhosti k a délce ℓ , která je uchycená na obou koncích a zatížená příčně silou s konstantní hustotou c . Své výsledky získané pomocí metody sítí můžete srovnat se známým analytickým řešením.

Příklad 2 Svou implementaci metody sítí z předchozího příkladu jednoduše modifikujte pro případ nekonstantního zatížení s hustotou $f(x) := (x - 1/3)^2 - 1$. Vyzkoušejte si, jak implementace funguje například pro volby $k = 2$, $\ell = 2$ a $n = 300$.

Příklad 3 Uvažujte strunu délky $\ell = 2$, která je uchycená na obou koncích a pod níž se ve vzdálenosti $d = 0,2$ nachází rovinná překážka. S přesností 10^{-1} určete nejmenší možnou tuhost struny⁴ tak, aby při příčném zatížení silou o hustotě $f(x) := (x - 1/3)^2 - 1$ nedošlo ke kontaktu s překážkou. K řešení použijte metodu sítí, přičemž budete volit $n = 300$.

⁴Určete tedy nejmenší možnou hodnotu koeficientu k .

Kapitola 5

Optimalizační úlohy

5.1 Několik definic na úvod: gradient, Hessián

Než se začneme věnovat optimalizačním úlohám, seznámíme se s některými pojmy matematické analýzy funkcí více proměnných.

Nejprve si zavedeme gradient a Hessián pro funkci více proměnných. Buď $f : \mathbb{R}^n \mapsto \mathbb{R}$,¹ $x \in \mathbb{R}^n$ je vnitřní bod Df a $k \in \{1, 2, \dots, n\}$. Má-li funkce $g_k : \mathbb{R} \mapsto \mathbb{R}$ definovaná

$$g_k(t) := g(x_1, x_2, \dots, x_{k-1}, t, x_{k+1}, \dots, x_n)$$

vlastní derivaci v bodě x_k , nazýváme ji parciální derivací funkce f podle k -té proměnné v bodě x a značíme ji

$$\frac{\partial f(x)}{\partial x_k} \quad \left(\text{tj. } \frac{\partial f(x)}{\partial x_k} := g'_k(x_k) \right).$$

Poznamenejme, že podobně můžeme definovat parciální derivace vyšších řádů.

Buď $f : \mathbb{R}^n \mapsto \mathbb{R}$ funkce, která má v bodě x spojitě všechny parciální derivace prvního řádu. Vektor

$$\nabla f(x) := \left[\frac{\partial f(x)}{\partial x_1}, \frac{\partial f(x)}{\partial x_2}, \dots, \frac{\partial f(x)}{\partial x_n} \right]$$

nazýváme gradient funkce f v bodě x .

¹Takové funkci říkáme reálná funkce více reálných proměnných.

Bud' $f : \mathbb{R}^n \mapsto \mathbb{R}$ funkce, která má v bodě x spojité všechny parciální derivace druhého řádu. Matici

$$\nabla^2 f(x) := \begin{bmatrix} \frac{\partial^2 f(x)}{\partial x_1^2} & \frac{\partial^2 f(x)}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 f(x)}{\partial x_1 \partial x_n} \\ \frac{\partial^2 f(x)}{\partial x_2 \partial x_1} & \frac{\partial^2 f(x)}{\partial x_2^2} & \cdots & \frac{\partial^2 f(x)}{\partial x_2 \partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 f(x)}{\partial x_n \partial x_1} & \frac{\partial^2 f(x)}{\partial x_n \partial x_2} & \cdots & \frac{\partial^2 f(x)}{\partial x_n^2} \end{bmatrix}$$

nazýváme Hessiánem nebo Hessovou maticí funkce f v bodě x .

5.2 Rovnovážná poloha tělesa

Nyní se seznámíme s tzv. optimalizačními úlohami, ukážeme si jejich praktickou aplikaci a také jak je lze numericky řešit. Mnohem více mohou čtenáři nalézt v textu [4].

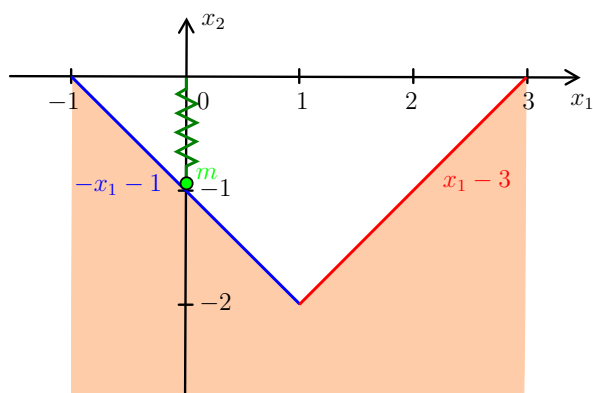
Optimalizační úlohy vznikají často při řešení praktických úloh. Matematicky jsou tyto úlohy obvykle formulovány jako problémy hledání extrémů cenové funkce, což je funkce $f : \mathbb{R}^n \mapsto \mathbb{R}$. Extrémy cenové funkce často hledáme na přípustné množině $\Omega \subset Df$. Jelikož bod, v němž funkce f nabývá svého minima, je stejný jako bod, v němž funkce $-f$ nabývá svého maxima, můžeme za obecnou úlohu optimalizace považovat problém najít $\bar{x} \in \Omega$ tak, aby platilo

$$f(\bar{x}) \leq f(x), \quad x \in \Omega.^2 \quad (5.1)$$

Pokud je $\Omega = Df$, mluvíme o optimalizaci bez omezení. Pokud pro Ω platí $\Omega \subset Df$ a $\Omega \neq Df$, mluvíme o optimalizaci s omezením.

Příkladem optimalizační úlohy bez omezení je například problém nalezení rovnovážné polohy kuličky o hmotnosti m volně zavěšené na pružině. Příkladem optimalizační úlohy s omezením je například problém nalezení rovnovážné polohy kuličky o hmotnosti m zavěšené na pružině, která visí nad překážkou. Uvažujme, že pružina je uchycena v obou případech v bodě $[0, 0]$. Tyto problémy můžeme vyjádřit jako minimalizaci funkce potenciální energie tohoto systému. V prvním případě je $\Omega = \mathbb{R}^2$, v druhém případě přípustná množina Ω popisuje oblast, ve které se kulička může pohybovat a která leží mimo překážku. Tyto úlohy pak zapíšeme jako úlohu najít $\bar{x} \in \Omega$ tak, aby platila nerovnost (5.1), kde $f(x) = x_1^2 + x_2^2 + mx_2$ a $\Omega = \mathbb{R}^2$ nebo $\Omega = \{x \in \mathbb{R}^2 : x_1 + x_2 + 1 \geq 0, -x_1 + x_2 + 3 \geq 0\}$ (pro druhou úlohu viz obr. 5.1).

²Tuto úlohu můžeme zadat také takto: „Řešte úlohu $\min_{x \in \Omega} f(x)$.“



Obrázek 5.1: Příklad rovnovážného stavu kuličky na pružině (úloha s omezením)

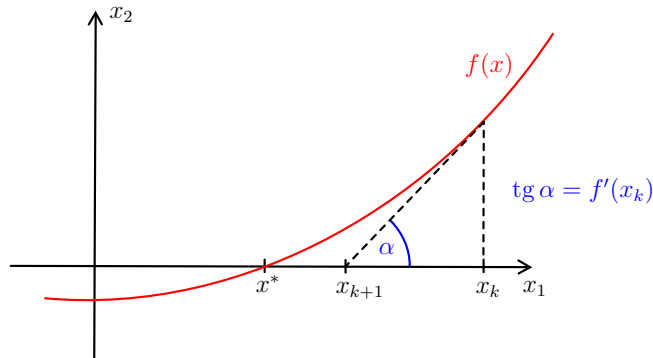
Odkazy na obecně formulované optimalizační úlohy najdeme již ve starověku. Patří mezi ně i Dídónina³ úloha. Dídó údajně dostala po přistání na africkém pobřeží od místního vládce kůži z jednoho býka. Se svým doprovodem měla právo usadit se na pozemku, který lze ohraničit touto kůží. Kůži rozřezala na úzké proužky a pomocí nich ohraničili území, na kterém založila město Kartágo. Snaha získat co největší pozemek vede na problém najít spojitou křivku dané délky, která ohraničuje co největší plochu.

Dále se budeme zabývat numerickým řešením optimalizačních úloh.

Metody řešení úloh bez omezení

Pro úlohu (5.1) existuje řada přístupů, jak ji numericky řešit. Pro „pěkné“ funkce je velmi efektivní Newtonova metoda. Než se začneme věnovat odvození Newtonovy metody pro řešení optimalizační úlohy (5.1), odvodíme si iterační metodu pro hledání řešení rovnice. Tato metoda se také nazývá Newtonova. Připomeňme, že jsme již tuto metodu odvodili v kapitole 1.5. Tam jsme ale pro odvození použili metodu prostých iterací. Nyní předpokládejme, že v k -tém kroce iterační metody máme bod aproximující řešení x^* rovnice $f(x) = 0$. Tento bod označíme x_k . Dále předpokládejme, že známe funkční hodnotu funkce f v bodě x_k a derivaci funkce f v bodě x_k . Nyní chceme učinit $(k+1)$ -ní krok metody a nalézt bod x_{k+1} , který „lépe“ aproximuje řešení rovnice $f(x) = 0$. Využitím vlastnosti derivace dostaneme z pravoúhlého trojúhelníku v obr. 5.2 následující vztah

³Dle legendy byla Dídó fénická princezna, sestra krále Pygmalióna a zakladatelka starověkého Kartága.



Obrázek 5.2: Newtonova metoda

$$f'(x_k) = \frac{f(x_k)}{x_k - x_{k+1}}. \quad (5.2)$$

Rovnici (5.2) snadno upravíme do tvaru

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}. \quad (5.3)$$

Tím dostáváme předpis, jehož opakovaným použitím dostáváme zpřesňující se aproximaci řešení x^* rovnice $f(x) = 0$. Pomocí tohoto předpisu získáme následující algoritmus pro iterační řešení rovnice $f(x) = 0$.

Algoritmus (Newtonova metoda (pro řešení rovnice s jednou neznámou))

1. $\varepsilon > 0$ (ukončující podmínka)
 x_0 (počáteční bod iteračního procesu)
 $k = 0$
 $x_1 = x_0 - f(x_0)/f'(x_0)$
2. while $|x_{k+1} - x_k| \geq \varepsilon$ ($|f(x_{k+1})| \geq \varepsilon$)
 $k = k + 1$
 $x_{k+1} = x_k - f(x_k)/f'(x_k)$
end
3. x_{k+1} aproximuje řešení rovnice $f(x) = 0$

Nyní využijeme předpisu (5.3) pro hledání extrému funkce f . Protože pro funkce f , které mají v celém svém definičním oboru derivaci, platí, že extrémy této funkce jsou uvnitř definičního oboru v bodech splňujících $f'(x) = 0$,⁴

⁴Této podmínce říkáme nutná podmínka existence lokálního extrému.

budeme numericky hledat řešení rovnice $f'(x) = 0$. Použitím předpisu (5.3) obdržíme

$$x_{k+1} = x_k - \frac{f'(x_k)}{f''(x_k)}. \quad (5.4)$$

Pokud bude funkce f navíc konvexní v celém svém definičním oboru, pak uvedený postup nalezne minimum funkce. Vše shrneme do následujícího algoritmu.

Algoritmus (Newtonova metoda (pro minimalizaci funkce jedné proměnné))

1. $\varepsilon > 0$ (ukončující podmínka)
 x_0 (počáteční bod iteračního procesu)
 $k = 0$
 $x_1 = x_0 - f'(x_0)/f''(x_0)$
2. while $|x_{k+1} - x_k| \geq \varepsilon$ ($|f'(x_{k+1})| \geq \varepsilon$)
 $k = k + 1$
 $x_{k+1} = x_k - f'(x_k)/f''(x_k)$
end
3. x_{k+1} aproximuje minimum funkce $f(x)$

Zobecněním předpisu (5.4) pro funkce více proměnných získáme metodu pro minimalizaci funkce více proměnných. Zobecněním první derivace je vektor všech prvních parciálních derivací, který nazýváme gradient (označujeme jej, jak jsme již uvedli v úvodu této kapitoly, jako $\nabla f(x)$). Zobecněním druhé derivace je matice všech druhých parciálních derivací, kterou nazýváme Hessova matice nebo Hessián (značíme jej $\nabla^2 f(x)$). Absolutní hodnotu $|x|$ zobecníme na velikost vektoru $\|x\| := \sqrt{x_1^2 + x_2^2 + \dots + x_n^2}$. Vše opět přepíšeme do algoritmu.

Algoritmus (Newtonova metoda (pro minimalizaci funkce více proměnných))

```

1.  $\varepsilon > 0$  (ukončující podmínka)
    $x_0$  (počáteční bod iteračního procesu)
    $k = 0$ 
    $x_1 = x_0 - (\nabla^2 f(x_0))^{-1} \cdot \nabla f(x_0)$ 
2. while  $\|x_{k+1} - x_k\| \geq \varepsilon$  ( $\|\nabla f(x_{k+1})\| \geq \varepsilon$ )
    $k = k + 1$ 
    $x_{k+1} = x_k - (\nabla^2 f(x_k))^{-1} \cdot \nabla f(x_k)$ 
end
3.  $x_{k+1}$  aproximuje minimum funkce  $f(x)$ 

```

Poznamenejme, že $(\nabla^2 f(x_k))^{-1}$ značí inverzní matici k matici $\nabla^2 f(x_k)$.

Metody řešení úloh s omezeními

Velmi jednoduchá metoda převedení úlohy s omezením na úlohu bez omezení spočívá v zahrnutí nerovnostních podmínek obsažených v popisu přípustné množiny Ω do cenové funkce tak, že k ní přičteme vhodný člen, který penalizuje porušení nerovnostních podmínek. Tuto metodu nazýváme metodou kvadratické penalty a použijeme ji pro úlohu s omezením, která je ve tvaru

$$\min_{x \in \Omega} f(x), \quad \Omega = \{x \in \mathbb{R}^n : g(x) \leq o\}, \quad (5.5)$$

kde g je vektorová funkce⁵ a o označuje nulový vektor $[0, \dots, 0]$. Metoda kvadratické penalty aproximuje řešení \bar{x} úlohy (5.5) řešením \bar{x}_ϱ úlohy

$$\min_{x \in \mathbb{R}^n} f_\varrho(x), \quad f_\varrho(x) = f(x) + \frac{1}{2}\varrho\|\alpha(x)\|^2, \quad \alpha(x) = \max\{g(x), o\}, \quad (5.6)$$

kde $\varrho > 0$ je **penalizační parametr** a $\|\alpha(x)\|^2$ je **penalizační funkce**.

Intuitivně je zřejmé, že když je penalizační parametr ϱ velký, tak řešení \bar{x}_ϱ , ve kterém je dosaženo minimum penalizované funkce f_ϱ , nemůže být daleko od přípustné množiny. Dokonce platí, že kdyby $\varrho = \infty$, pak by bylo minimum f_ϱ řešením původní úlohy. Můžeme tedy očekávat, že pro dostatečně velké hodnoty penalizačního parametru ϱ bude řešení \bar{x} úlohy (5.5) blízko \bar{x}_ϱ . Je zřejmé, že řešení penalizované úlohy je typicky blízko přípustné

⁵Tento pojem jsme si zavedli již v úvodu kapitoly 3.7.

⁶Funkce \max znamená maximum „po složkách“, tzn. funkční hodnotou této funkce je vektor, který v i -té složce obsahuje $\max\{g_i(x), 0\}$.

množiny, avšak nepatří do ní. Proto se naše penalizační metoda nazývá také metoda externí penalty.

Příklad 1 Nalezněte rovnovážnou polohu kuličky hmotnosti m volně zavěšené na pružině (pružina je uchycena v bodě $[0, 0]$), tj. najděte řešení následující minimalizační úlohy

$$\min_{x \in \mathbb{R}^2} x_1^2 + x_2^2 + mx_2, \quad (5.7)$$

kde hmotnost m volte postupně jako $m = 1$, $m = 2$, $m = 3$, $m = 4$, $m = 5$ a $m = 10$.

Příklad 2 Nalezněte rovnovážnou polohu kuličky hmotnosti m zavěšené na pružině (pružina je uchycena v bodě $[0, 0]$), která visí nad překážkou, tj. najděte řešení následující minimalizační úlohy

$$\min_{x \in \Omega} x_1^2 + x_2^2 + mx_2, \quad (5.8)$$

$$\Omega = \{x \in \mathbb{R}^2 : x_1 + x_2 + 1 \geq 0, -x_1 + x_2 + 3 \geq 0\}, \quad (5.9)$$

kde hmotnost m volte postupně jako $m = 1$, $m = 2$, $m = 3$, $m = 4$, $m = 5$ a $m = 10$ (viz obr. 5.1).

Kapitola 6

Řešení úloh

6.1 Úlohy kapitoly 1

Řešení ke kapitole 1.2

Příklad 1 Z obr. 1.5 vidíme, že pro obsah S jednotkového kruhu ($S = \pi$) platí

$$S = 4 \int_0^1 \sqrt{1-x^2} dx.$$

Výše uvedený integrál nyní přibližně vypočteme pomocí obdélníkového a lichoběžníkového pravidla. Nejprve si rozdělme interval $\langle 0, 1 \rangle$ na n stejných dílků.

```
h = 1 / n; % delka dilku
x = 0 : h : 1; % krajni body jednotlivych dilku
```

Nyní si uveďme kód, který pomocí obdélníkového pravidla vypočte aproximaci S_{obd} obsahu S .

```
c = 0.5 * (x(1:n) + x(2:n+1)); % stredy jednotlivych dilku
f = sqrt(1 - c.^2);
S_obd = 4 * h * sum(f);
```

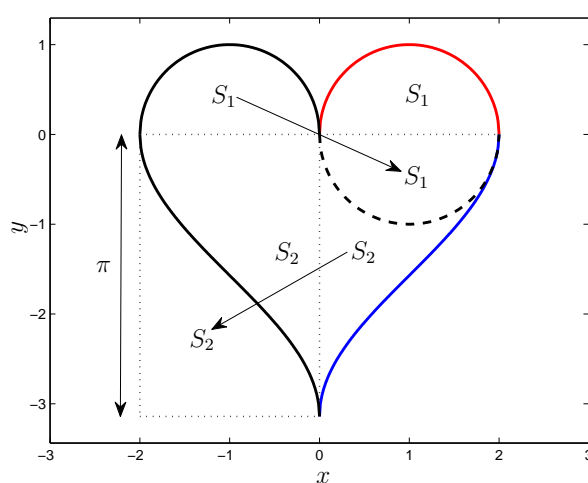
Kód níže vypočte pomocí lichoběžníkového pravidla aproximaci S_{lich} obsahu S .

```
f = sqrt(1 - x.^2);
S_lich = 4 * h/2 * ( f(1) + f(n+1) + 2*sum(f(2:n)) );
```

Následující tabulka uvádí získané chyby aproximace pro různá n :

n	$ S_{obd} - \pi $	$ S_{lich} - \pi $
10^2	$3,44 \cdot 10^{-4}$	$1,18 \cdot 10^{-3}$
10^3	$1,09 \cdot 10^{-5}$	$3,72 \cdot 10^{-5}$
10^4	$3,44 \cdot 10^{-7}$	$1,18 \cdot 10^{-6}$

Příklad 2 Nejprve se podívejme na obr. 6.1. Pro obsah S srdce platí



Obrázek 6.1: Obsah matematického srdce

$$S = \underbrace{2S_1}_{\pi} + \underbrace{2S_2}_{2\pi} = 3\pi.$$

Nyní si S vyjádříme jako¹

$$S = 2(S_1 + S_2) = 2 \left(\int_0^2 \sqrt{1 - (x-1)^2} dx - \int_0^2 \left(\arcsin(x-1) - \frac{\pi}{2} \right) dx \right),$$

přičemž pro přibližné vyčíslení těchto integrálů použijeme obdélníkové a lichoběžníkové pravidlo. Nejprve si rozdělme interval $\langle 0, 2 \rangle$ na n stejných dílků.

`h = 2 / n; % delka dilku`

`x = 0 : h : 2; % krajni body jednotlivych dilku`

¹Uvědomme si, že plocha S_2 leží pod osou x .

Nyní vypočteme pomocí obdélníkového pravidla přibližný obsah srdce S_{obd} :

```

c = 0.5 * (x(1:n) + x(2:n+1)); % stredy jednotlivych dilku
f1 = sqrt(1 - (c-1).^2);
S1 = h * sum(f1);
f2 = asin(c-1) - pi/2;
S2 = - h * sum(f2);
S_obd = 2 * (S1 + S2);

```

Pomocí níže uvedeného kódu získáme přibližný obsah srdce S_{lich} vypočtený lichoběžníkovým pravidlem:

```

f1 = sqrt(1 - (x-1).^2);
S1 = h/2 * ( f1(1) + f1(n+1) + 2*sum(f1(2:n)) );
f2 = asin(x-1) - pi/2;
S2 = - h/2 * ( f2(1) + f2(n+1) + 2*sum(f2(2:n)) );
S_lich = 2 * (S1 + S2);

```

V následující tabulce uvádíme získané chyby aproximace pro různá n :

n	$ S_{obd} - 3\pi $	$ S_{lich} - 3\pi $
10^2	$9,73 \cdot 10^{-4}$	$3,32 \cdot 10^{-3}$
10^3	$3,08 \cdot 10^{-5}$	$1,05 \cdot 10^{-5}$
10^4	$9,74 \cdot 10^{-7}$	$3,33 \cdot 10^{-6}$

Příklad 3 K nalezení počtu potřebných dílků n lze využít odhad (1.4). Jelikož platí $(x^2)'' = (2x)' = 2$, můžeme spočítat n jako řešení nerovnice

$$\frac{(1-0)^3}{24n^2} \cdot 2 \leq 10^{-4},$$

tedy

$$n \geq \frac{1}{\sqrt{12 \cdot 10^{-4}}} \doteq 28,9.$$

Dělení intervalu $\langle 0, 1 \rangle$ na 29 (a více) stejných dílků nám tak zaručí, že chyba aproximace bude nejvýše 10^{-4} .

K nalezení počtu potřebných dílků n můžeme využít i Matlab. Nejprve si uvědomme, že díky Newtonovu-Leibnizovu vzorci máme

$$\int_0^1 x^2 dx = \left[\frac{x^3}{3} \right]_0^1 = \frac{1}{3}.$$

Níže uvedený kód najde minimální počet dílků n , pro který chyba aproximace err nebude větší než 10^{-4} .

```

n = 0; err = 1; % inicializace

while err > 1e-4
    n = n + 1;

    h = 1 / n; % delka dilku
    x = 0 : h : 1; % krajni body jednotlivych dilku
    c = 0.5 * (x(1:n) + x(2:n+1)); % stredy jednotlivych dilku
    f = c.^2;
    S_obd = h * sum(f);

    err = abs(S_obd - 1/3);
end

```

Po průchodu smyčkou programu dostáváme $n = 29$ a $err = 9,91 \cdot 10^{-5}$. Je dobré si uvědomit, že výše uvedený program sám o sobě nezaručí, že pro $n > 29$ nebude chyba aproximace větší než 10^{-4} . Tuto informaci nám poskytnete, jak už jsme uvedli, odhad (1.4).

Řešení ke kapitole 1.3

Příklad 1 Následující kód pro dané n vypočte částečný součet s_n řady

$$\sum_{k=1}^{\infty} \frac{1}{k(k+1)}. \quad (6.1)$$

```

s_n = 0; % inicializace
for k = 1 : n
    s_n = s_n + 1/k/(k + 1);
end

```

Několik spuštění programu nám napovídá, že by se součet řady (6.1) mohl rovnat jedné - dostáváme například:

n	s_n
10^2	0,990099
10^3	0,999001
10^4	0,999900
10^5	0,999990

Uvedeme si nyní důkaz, že tomu tak opravdu je. Nejprve si uvědomme, že

$$\frac{1}{k(k+1)} = \frac{1+k-k}{k(k+1)} = \frac{1}{k} - \frac{1}{k+1}.$$

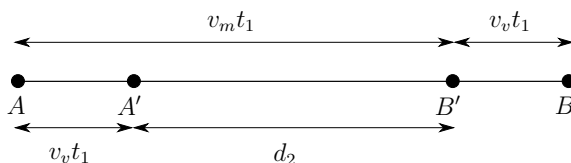
Takže

$$\begin{aligned} s_n &= \sum_{k=1}^n \frac{1}{k(k+1)} = \sum_{k=1}^n \left(\frac{1}{k} - \frac{1}{k+1} \right) = \\ &= \left(\frac{1}{1} - \frac{1}{2} \right) + \left(\frac{1}{2} - \frac{1}{3} \right) + \left(\frac{1}{3} - \frac{1}{4} \right) + \cdots + \left(\frac{1}{n} - \frac{1}{n+1} \right) = 1 - \frac{1}{n+1}. \end{aligned}$$

Jelikož $\lim [1/(n+1)] = 0$, dostáváme

$$\sum_{k=1}^{\infty} \frac{1}{k(k+1)} = \lim s_n = 1.$$

Příklad 2 Nejprve si připomeňme, že těleso pohybující se rovnoměrně přímočaře rychlostí v urazí za čas t vzdálenost $s = vt$. Nahlédněme nyní na obr. 6.2. Moucha vyletěla rychlostí $v_m = 100$ km/h z města A současně s vlakem a



Obrázek 6.2: Vlaky a moucha

letí vstříc druhému vlaku, který vyjel z města B . Oba vlaky jedou rychlostí $v_v = 10$ km/h. Počáteční vzdálenost mezi vlaky (tj. vzdálenost měst A a B) je $d_1 = 90$ km. Za čas t_1 se moucha setká s vlakem z města B , přičemž místo setkání mouchy a vlaku označíme B' . Počáteční vzdálenost d_1 mezi vlaky si můžeme rozepsat jako

$$d_1 = v_m t_1 + v_v t_1,$$

přičemž první člen součtu značí dráhu prvního letu mouchy (tj. vzdálenost míst A a B') a druhý člen součtu značí dráhu vlaku, kterou ujel z města B za čas t_1 (tj. vzdálenost míst B' a B). Odtud

$$t_1 = \frac{d_1}{v_m + v_v} = \frac{90}{100 + 10} = \frac{9}{11}.$$

Vlak z města A dojel za čas t_1 do místa A' . Vzdálenost míst A a A' je rovna $v_v t_1$. Nyní se moucha odráží od vlaku v místě B' a letí vstříc druhému

vlaků. Vzdálenost vlaků v tomto okamžiku (tedy vzdálenost míst A' a B') lze vyjádřit jako

$$d_2 = d_1 - 2v_v t_1 = 90 - 2 \cdot 10 \cdot \frac{9}{11} = \frac{810}{11}.$$

Použitím stejných úvah jako výše si odvodíme čas druhého letu mouchy, a to

$$t_2 = \frac{d_2}{v_m + v_v} = \frac{\frac{810}{11}}{100 + 10} = \left(\frac{9}{11}\right)^2.$$

Obdobně bychom postupovali dále a zjistili bychom, že pro dobu n -tého letu mouchy platí

$$t_n = \left(\frac{9}{11}\right)^n.$$

Délku n -tého letu mouchy pak vyjádříme jako

$$s_n = v_m t_n = 100 \left(\frac{9}{11}\right)^n,$$

a proto

$$s_{15} = 100 \left(\frac{9}{11}\right)^{15} \doteq 4,93,$$

tj. moucha mezi 15. a 16. odražením uletěla přibližně 4,93 km. Celkovou délku letu mouchy získáme jako součet délek všech² letů mouchy, tj. jako součet geometrické řady³

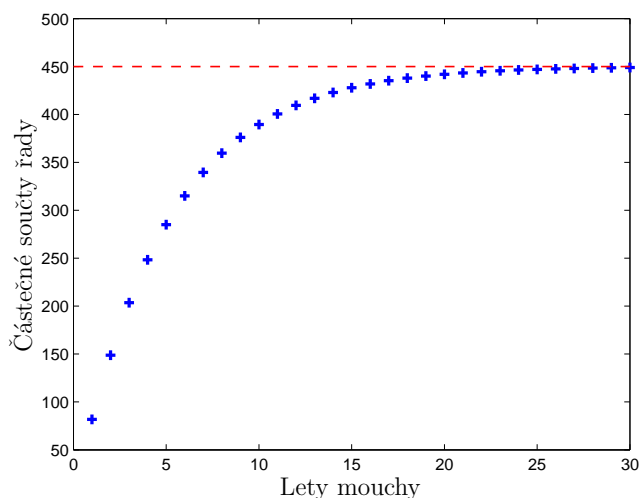
$$\sum_{n=1}^{\infty} s_n = 100 \sum_{n=1}^{\infty} \left(\frac{9}{11}\right)^n = 100 \frac{\frac{9}{11}}{1 - \frac{9}{11}} = 450.$$

Celkem tedy moucha nalétala 450 km. Pro zajímavost je na obr. 6.3 znázorněna posloupnost částečných součtů řady $\sum_{n=1}^{\infty} s_n$.

Poznamenejme ještě, že pokud by nás nezajímala délka dílčích letů mouchy, ale jen celková vzdálenost, kterou moucha nalétala, úloha by se dala řešit jednoduše následujícím způsobem. Jelikož se vlaky srazí přesně v polovině trati (tj. oba ujedou 45 km), dostáváme, že se vlaky srazí za čas $t_s = 45/v_v = 4,5$ hodiny. Celkem tedy moucha nalétala $v_m t_s = 100 \cdot 4,5 = 450$ km.

²Letů mouchy je nekonečně mnoho.

³V následujícím výpočtu využijeme toho, že $\sum_{n=1}^{\infty} q^n = \frac{q}{1-q}$ pro $|q| < 1$.



Obrázek 6.3: Celková uražená dráha mouchy v závislosti na letech

Příklad 3 Uvádíme program, který vypočte, kolik členů harmonické řady musíme nejméně sečíst, aby částečný součet s_n harmonické řady měl hodnotu alespoň h , kde h je „malé“.

```
s_n = 0; n = 0; % inicializace
while s_n < h
    n = n + 1;
    s_n = s_n + 1/n;
end
```

Po ukončení cyklu je v proměnné n uložen počet členů harmonické řady, pro něž dostáváme nejmenší částečný součet s hodnotou alespoň h . Posloupnost částečných součtů této řady roste jen velmi pomalu:

h	n
5	83
10	12367
15	1835421
20	272400600

Je dokázáno, že například pro $h = 100$ bychom potřebovali sečíst alespoň

15092688622113788323693563264538101449859497

členů harmonické řady. Tento výsledek ovšem nelze získat pomocí výše uvedeného kódu. Pokud bychom se totiž tolik členů harmonické řady pokusili

sečíst na počítači, museli bychom na výsledek čekat řádově 10^{17} miliard let (za předpokladu, že našemu počítači trvá přičtení každého nového členu 10^{-9} sekund).

Uveďme si ještě pro zajímavost náznak důkazu divergence harmonické řady:

$$\begin{aligned} \sum_{n=1}^{\infty} \frac{1}{n} &= 1 + \frac{1}{2} + \frac{1}{3} + \frac{1}{4} + \frac{1}{5} + \frac{1}{6} + \frac{1}{7} + \frac{1}{8} + \frac{1}{9} + \frac{1}{10} + \cdots + \frac{1}{16} + \cdots \\ &> 1 + \frac{1}{2} + \underbrace{\frac{1}{4} + \frac{1}{4}}_{1/2} + \underbrace{\frac{1}{8} + \frac{1}{8} + \frac{1}{8} + \frac{1}{8}}_{1/2} + \underbrace{\frac{1}{16} + \frac{1}{16} + \cdots + \frac{1}{16}}_{1/2} + \cdots \\ &= \infty. \end{aligned}$$

Příklad 4 Následující kód pro dané n vypočte částečný součet s_n řady

$$\sum_{k=1}^{\infty} 4 \frac{(-1)^{k+1}}{2k-1}. \quad (6.2)$$

```
s_n = 0; % inicializace
for k = 1 : n
    s_n = s_n + (-1)^(k+1)/(2*k - 1);
end
s_n = 4 * s_n;
```

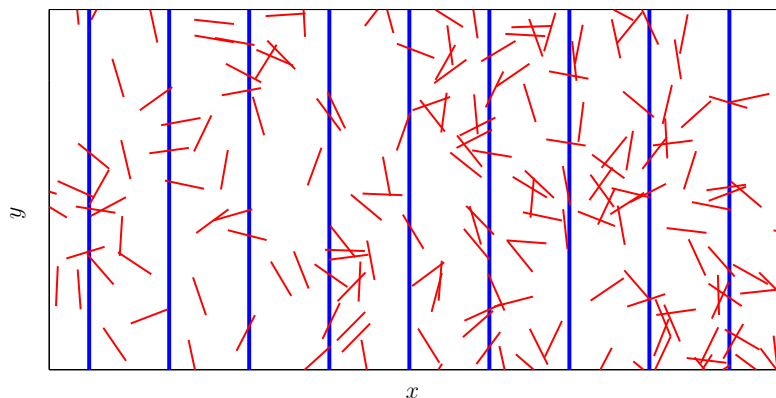
Několik spuštění programu nám napovídá, že by se součet řady (6.2) mohl rovnat π – dostáváme například:

n	s_n
10^3	3,140593
10^4	3,141493
10^5	3,141583
10^6	3,141592

Více o této aproximaci čísla π pomocí číselné řady najdete na konci kapitoly 1.4.

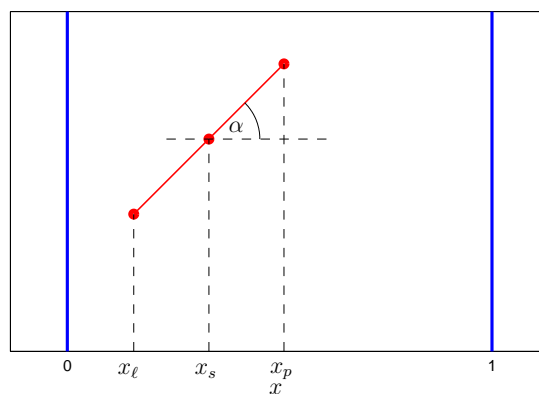
Řešení ke kapitole 1.4

Příklad 1 Uvažujme rovinu a na ní vyznačenu síť rovnoběžek o vzdálenosti $2\ell = 1$. Na rovinu házíme n -krát jehlu o délce $\ell = 1/2$, viz obr. 6.4. Nejdříve si uvědomme, že se v další analýze můžeme bez újmy na obecnosti omezit pouze na dvě sousední rovnoběžky (volme například přímky $x = 0$ a



Obrázek 6.4: Házení jehel na síť rovnoběžek

$x = 1$) a také že y -ová souřadnice jehly nebude ovlivňovat to, zda jehla protne či neprotne některou z přímek. Inspirujme se nyní obr. 6.5 a konstruujme náhodný hod. Nejprve si vygenerujeme náhodné číslo x_s z intervalu $\langle 0, 1 \rangle$.



Obrázek 6.5: Náhodně hozená jehla

Toto číslo bude představovat x -ovou souřadnici středu jehly. Dále si vygenerujeme náhodné číslo α z intervalu $\langle -\pi/2, \pi/2 \rangle$, což bude úhel, který svírá jehla s kladným směrem osy x . Vypočteme nyní hodnoty x_ℓ a x_p představující x -ové souřadnice levého a pravého konce jehly:

$$x_\ell = x_s - \frac{1}{4} \cos \alpha \quad \text{a} \quad x_p = x_s + \frac{1}{4} \cos \alpha.$$

Jehla protne přímkou $x = 0$ v případě, že $x_\ell \leq 0$, nebo přímkou $x = 1$ v případě, že $x_p \geq 1$. Nyní si uveďme program, který implementuje výše zmíněné úvahy. Do proměnné s se načítá počet hodů, při kterých jehla protla některou z přímek, a získaná aproximace čísla π se uloží do proměnné π_{approx} .

```
s = 0; % inicializace

for i = 1 : n
    x_s = rand;
    alfa = -pi/2 + rand * pi;

    x_l = x_s - 1/4*cos(alfa);
    x_p = x_s + 1/4*cos(alfa);

    if x_l <= 0 | x_p >= 1
        s = s + 1; % jehla protne nekterou z primek
    end
end

pi_aprox = n / s;
```

Na závěr poznamenejme, že náš program vytvořený k aproximaci π je pouze ilustrativní, jelikož sám π obsahuje. Uveďme si ještě tabulku výsledků pro jedno konkrétní spuštění programu.⁴

n	$ \pi_{approx} - \pi $	$9,011/\sqrt{n}$
10^2	$2 \cdot 10^{-1}$	$9,01 \cdot 10^{-1}$
10^3	$3,3 \cdot 10^{-2}$	$2,85 \cdot 10^{-1}$
10^4	$8,89 \cdot 10^{-4}$	$9,01 \cdot 10^{-2}$
10^5	$1,61 \cdot 10^{-2}$	$2,85 \cdot 10^{-2}$
10^6	$7,56 \cdot 10^{-3}$	$9,01 \cdot 10^{-3}$
10^7	$7,19 \cdot 10^{-4}$	$2,85 \cdot 10^{-3}$
10^8	$8,67 \cdot 10^{-4}$	$9,01 \cdot 10^{-4}$

Příklad 2 Nejprve počítejme pomocí Newtonova-Leibnizova vzorce:

$$\int_0^1 x^2 dx = \left[\frac{x^3}{3} \right]_0^1 = \frac{1}{3}.$$

⁴Uvědomme si, že při dalším spuštění obdržíme, vzhledem k přítomnosti náhodně generovaných čísel, výsledky jiné; jak jsme si však uvedli v kapitole 1.4, chyba aproximace není s pravděpodobností 95 procent větší než $9,011/\sqrt{n}$.

Nyní si uvedeme kód, který pomocí metody Monte Carlo přibližně vypočte $\int_0^1 x^2 dx$ a spočtenou hodnotu uloží do proměnné *obsah*. Proměnná *n* udává počet náhodně generovaných bodů v jednotkovém čtverci a do proměnné *s* se načítá, kolik z těchto bodů leží pod grafem či na grafu funkce $y = x^2$.

```
s = 0; % inicializace

for i = 1 : n
    x = rand; y = rand; % nahodny bod [x,y] z (0,1)x(0,1)
    if y <= x^2
        s = s + 1;
    end
end

obsah = s / n;
```

Uveďme si ještě tabulku výsledků pro jedno konkrétní spuštění programu.⁵

n	$ obsah - 1/3 $	$1/\sqrt{n}$
10^2	$3,33 \cdot 10^{-3}$	$1 \cdot 10^{-1}$
10^3	$6,33 \cdot 10^{-3}$	$3,16 \cdot 10^{-2}$
10^4	$1,03 \cdot 10^{-3}$	$1 \cdot 10^{-2}$
10^5	$1,44 \cdot 10^{-3}$	$3,16 \cdot 10^{-3}$
10^6	$2,04 \cdot 10^{-4}$	$1 \cdot 10^{-3}$
10^7	$1,23 \cdot 10^{-4}$	$3,16 \cdot 10^{-4}$
10^8	$7,32 \cdot 10^{-6}$	$1 \cdot 10^{-4}$

Příklad 3 Následující program implementuje metodu Monte Carlo pro přibližný výpočet integrálu $\int_0^1 \sqrt{1-x^2} dx$ a vypočte aproximaci π_{approx} čísla π . Jedná se o jednoduchou modifikaci programu z příkladu 2.

```
s = 0; % inicializace

for i = 1 : n
    x = rand; y = rand; % nahodny bod [x,y] z (0,1)x(0,1)
    if x^2 + y^2 <= 1
        s = s + 1;
    end
end
```

⁵Při dalším spuštění obdržíme (vzhledem k přítomnosti náhodně generovaných čísel) výsledky jiné. Chyba aproximace není s pravděpodobností 75 procent větší než $1/\sqrt{n}$.

```
pi_aprox = 4 * s / n;
```

Uvedme si ještě tabulku výsledků pro jedno konkrétní spuštění programu.⁶

n	$ \pi_{aprox} - \pi $
10^2	$5,84 \cdot 10^{-2}$
10^3	$2,24 \cdot 10^{-2}$
10^4	$5,21 \cdot 10^{-3}$
10^5	$2,83 \cdot 10^{-3}$
10^6	$3,61 \cdot 10^{-4}$
10^7	$9,51 \cdot 10^{-5}$
10^8	$1,27 \cdot 10^{-4}$

Příklad 4 Uvedeme kód, který do proměnné π_{aprox} uloží aproximaci čísla π vypočtenou pomocí m prvních členů Gregoryho řady.

```
s = 0; % inicializace

for n = 1 : m
    s = s + (-1)^(n+1)/(2*n - 1);
end

pi_aprox = 4 * s;
```

V následující tabulce jsou výsledky po spuštění programu:

m	$ \pi_{aprox} - \pi $
10^2	10^{-2}
10^3	10^{-3}
10^4	10^{-4}
10^5	10^{-5}
10^6	10^{-6}

Můžeme také použít vztah (1.6) pro výrazné zrychlení konvergence a nahradit tak cyklus ve výše uvedeném programu cyklem

```
for n = 1 : m
    k = 2*n - 1;
    s = s + (-1)^(n+1)/k * ( 1/2^k + 1/3^k );
end
```

⁶Opět si uvědomme, že při dalším spuštění obdržíme výsledky jiné.

Získané výsledky uvádíme v následující tabulce:

m	$ \pi_{approx} - \pi $
5	$1,49 \cdot 10^{-4}$
10	$7,4 \cdot 10^{-8}$
15	$4,87 \cdot 10^{-11}$
20	$3,69 \cdot 10^{-14}$

Řešení ke kapitole 1.5

Příklad 1 Máme za úkol nejprve vyřešit rovnici

$$g(x) := \ln(x) + (x - 1)^3 = 0$$

pomocí Newtonovy metody. Po chvilce přemýšlení nás možná napadne, že řešením naší rovnice je číslo 1. Spočtème nyní derivaci funkce g v bodě x :

$$g'(x) = \frac{1}{x} + 3(x - 1)^2.$$

Následující program pak realizuje řešení zadané rovnice Newtonovou metodou pro dané ε a počáteční aproximaci x_0 . Řešení rovnice program nakonec uloží do proměnné x .

```
x = x_0;

g = log(x) + (x-1)^3;
dg = 1/x + 3*(x-1)^2;

x_n = x - g/dg;

while abs(x_n - x) >= epsilon
    x = x_n;

    g = log(x) + (x-1)^3;
    dg = 1/x + 3*(x-1)^2;

    x_n = x - g/dg;
end

x = x_n;
```

Například pro $\varepsilon = 10^{-4}$ a $x_0 = 5$ dostáváme řešení $x \doteq 0,99999999997277$, pro které platí $g(x) \doteq -2,72 \cdot 10^{-12}$.

Dalším úkolem je vyřešit pomocí Newtonovy metody rovnici

$$g(x) := 1 - x - e^{-x^2} = 0.$$

Zamyslíme-li se opět nad řešením naší rovnice, mohlo by nás napadnout, že řešením je číslo 0. Spočtěme dále derivaci funkce g v bodě x :

$$g'(x) = -1 + 2xe^{-x^2}.$$

Program pro numerickou realizaci použijeme stejný jako výše, až na rozdílné definice $g(x)$ a $g'(x)$:

```
g = 1 - x - exp(-x^2);
dg = -1 + 2*x*exp(-x^2);
```

Například opět pro $\varepsilon = 10^{-4}$ a $x_0 = 5$ dostáváme řešení $x \doteq -4,22 \cdot 10^{-10}$, pro které platí $g(x) \doteq 4,22 \cdot 10^{-10}$.

Příklad 2 Budeme postupovat velmi podobně jako v řešení příkladu 1. Nejprve pomocí Newtonovy metody zkusme aproximovat číslo $\sqrt{2}$. Uvažujme (například) rovnici

$$g(x) := x^2 - 2 = 0$$

a počítejme $g'(x) = 2x$. Program použijeme stejný jako v příkladu 1, až na rozdílné definice $g(x)$ a $g'(x)$:

```
g = x^2 - 2;
dg = 2*x;
```

Pro volby $\varepsilon = 10^{-4}$ a $x_0 = 1$ dostáváme řešení $x \doteq 1,414213562374690$, pro něž platí $|x - \sqrt{2}| \doteq 1,59 \cdot 10^{-12}$.

Nyní pomocí Newtonovy metody zkusme aproximovat číslo π . Uvažujme (například) rovnici

$$g(x) := \sin x = 0$$

a počítejme $g'(x) = \cos x$. Program opět použijeme stejný jako v příkladu 1, až na rozdílné definice $g(x)$ a $g'(x)$:

```
g = sin(x);
dg = cos(x);
```

Pro volby $\varepsilon = 10^{-3}$ a $x_0 = 3$ dostáváme řešení $x \doteq 3,141592653300477$, pro něž platí $|x - \pi| \doteq 2,89 \cdot 10^{-10}$.

Řešení ke kapitole 1.6

Příklad 1 Funkci e^x nahradíme na okolí bodu $x_0 = 0$ Taylorovým polynomem stupně $n \in \mathbb{N}$:⁷

$$\begin{aligned}e^x \approx T_n(x) &= e^0 + e^0(x-0) + \frac{e^0}{2!}(x-0)^2 + \cdots + \frac{e^0}{n!}(x-0)^n \\ &= 1 + x + \frac{x^2}{2!} + \cdots + \frac{x^n}{n!}.\end{aligned}$$

Proto pro $x = 1$ dostáváme

$$e \approx T_n(1) = 1 + 1 + \frac{1}{2!} + \cdots + \frac{1}{n!}.$$

Nyní si uvedeme kód pro aproximaci čísla e dle výše uvedeného vztahu.

```
% inicializace
T_n = 1;
fact = 1; % promenna pro vypocet faktorialu

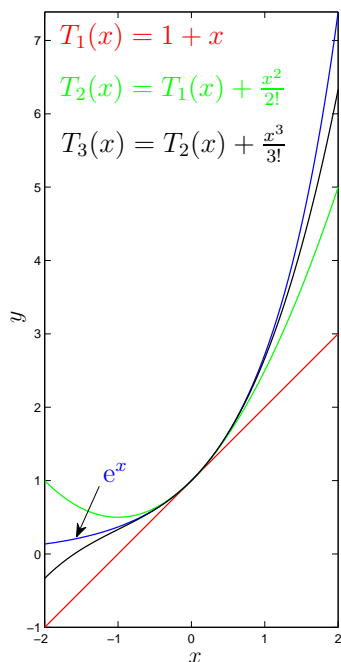
for k = 1 : n
    fact = fact * k;
    T_n = T_n + 1/fact;
end
```

Po doběhnutí programu máme (pro dané n) v proměnné T_n uloženu hodnotu $T_n(1)$ aproximující číslo e . Následující tabulka ilustruje získané výsledky

n	$R_{n+1}(1) = e - T_n(1)$
5	$1,62 \cdot 10^{-3}$
10	$2,73 \cdot 10^{-8}$
15	$5,06 \cdot 10^{-14}$

a na obr. 6.6 můžeme vidět, jak se na okolí bodu 0 Taylorovy polynomy postupně přibližují k funkci e^x .

⁷Připomeňme, že $(e^x)' = e^x$.



Obrázek 6.6: Nahrazení funkce e^x Taylorovými polynomy v bodě $x_0 = 0$

Příklad 2 Naším úkolem bude nahradit funkci $\operatorname{arctg} x$ na okolí bodu $x_0 = 0$ Taylorovým polynomem stupně $n \in \mathbb{N}$:

$$\operatorname{arctg} x \approx T_n(x),$$

přičemž

$$T_n(x) = \operatorname{arctg}(0) + \operatorname{arctg}'(0)(x-0) + \frac{\operatorname{arctg}''(0)}{2!}(x-0)^2 + \dots + \frac{\operatorname{arctg}^{(n)}(0)}{n!}(x-0)^n.$$

Platí, že $\operatorname{arctg}(0) = 0$, a derivováním zjistíme, že

n	$\operatorname{arctg}^{(n)}(x)$	$\operatorname{arctg}^{(n)}(0)$
1	$\frac{1}{1+x^2}$	1
2	$-\frac{2x}{(1+x^2)^2}$	0
3	$-2\frac{1-3x^2}{(1+x^2)^3}$	$-2 = -(2!)$
4	$24\frac{x(1-x^2)}{(1+x^2)^4}$	0
5	$24\frac{1-10x^2+5x^4}{(1+x^2)^5}$	$24 = 4!$
6	$-240\frac{x(3-10x^2+3x^4)}{(1+x^2)^6}$	0
7	$-720\frac{1-21x^2+35x^4-7x^6}{(1+x^2)^7}$	$-720 = -(6!)$
\vdots	\vdots	\vdots

Odtud lze odvodit, že

$$\operatorname{arctg}^{(n)}(0) = \begin{cases} 0 & \text{pro } n \text{ sudé,} \\ (-1)^{\frac{n+1}{2}+1}[(n-1)!] & \text{pro } n \text{ liché.} \end{cases}$$

Pro sudá n tedy platí, že $T_n(x) = T_{n-1}(x)$, a pro lichá n dostáváme⁸

$$\begin{aligned} T_n(x) &= x - \frac{2!x^3}{3!} + \frac{4!x^5}{5!} - \frac{6!x^7}{7!} + \dots + (-1)^{\frac{n+1}{2}+1} \frac{(n-1)!x^n}{n!} \\ &= x - \frac{x^3}{3} + \frac{x^5}{5} - \frac{x^7}{7} + \dots + (-1)^{\frac{n+1}{2}+1} \frac{x^n}{n} \\ &= \sum_{k=1}^{\frac{n+1}{2}} (-1)^{k+1} \frac{x^{2k-1}}{2k-1}. \end{aligned} \tag{6.3}$$

Proto pro $x = 1$ a lichá n máme

$$\frac{\pi}{4} = \operatorname{arctg}(1) \approx T_n(1) = \sum_{k=1}^{\frac{n+1}{2}} \frac{(-1)^{k+1}}{2k-1}$$

a tedy

$$\pi \approx 4 \sum_{k=1}^{\frac{n+1}{2}} \frac{(-1)^{k+1}}{2k-1}.$$

⁸Srovnejte s Leibnizovou řadou na str. 16.

Nyní si uvedeme program, který pro dané liché n vypočte aproximaci π_{approx} čísla π dle výše uvedeného vztahu.

```
T_n = 0; % inicializace

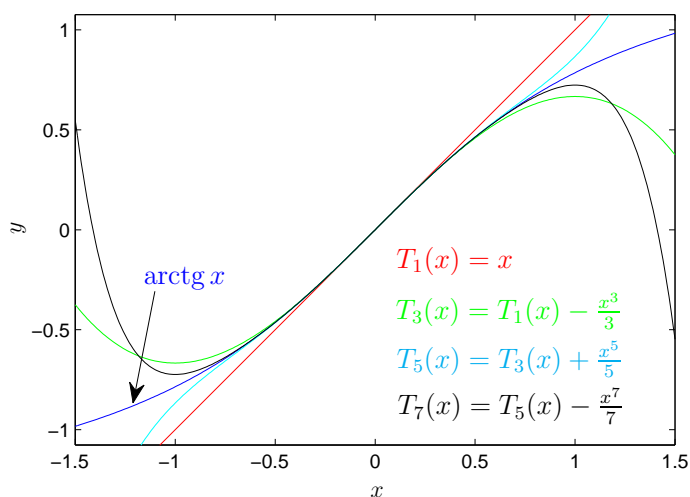
for k = 1 : (n + 1)/2
    T_n = T_n + (-1)^(k + 1)/(2*k - 1);
end

pi_aprox = 4 * T_n;
```

Následující tabulka ilustruje získané výsledky

n	$ \pi_{approx} - \pi $
$10^3 - 1$	$2 \cdot 10^{-3}$
$10^4 - 1$	$2 \cdot 10^{-4}$
$10^5 - 1$	$2 \cdot 10^{-5}$
$10^6 - 1$	$2 \cdot 10^{-6}$

a na obr. 6.7 můžeme vidět, jak se na okolí bodu 0 Taylorovy polynomy $T_n(x)$ postupně přibližují k funkci $\arctg x$.



Obrázek 6.7: Nahrazení funkce $\arctg x$ Taylorovými polynomy v bodě $x_0 = 0$

Příklad 3 Uvažujme funkci

$$f(x) := x - k(x^2 - 2), \quad k > 0.$$

Platí $f'(x) = 1 - 2kx$ a také, že existuje číslo λ takové, že⁹

$$\forall x \in I : |f'(x)| \leq \lambda < 1,$$

kde $I \subset (0, 1/k)$ je libovolný uzavřený interval. Aby pevný bod $\sqrt{2}$ funkce f náležel intervalu $(0, 1/k)$, předepíšeme

$$k \in (0, 1/2).$$

Použijeme-li pak větu na str. 18, dostáváme,¹⁰ že posloupnost (x_n) definovaná předpisem

$$x_{n+1} := f(x_n) = x_n - k(x_n^2 - 2), \quad n = 0, 1, 2, 3, \dots, \quad (6.4)$$

konverguje k $\sqrt{2}$ pro libovolné $x_0 \in (0, 1/k)$.

Nyní si uvedeme kód metody prostých iterací, který pro dané k , danou ukončující podmínku ε a počáteční aproximaci x_0 vydá aproximaci $\sqrt{2}_{approx}$ pevného bodu $\sqrt{2}$ funkce f . Proměnná n slouží k načítání počtu potřebných iterací, tj. k měření rychlosti konvergence.

```
% inicializace
n = 1;
x = x_0;
x_n = x - k*(x^2 - 2);

while abs(x_n - x) >= epsilon
    n = n + 1;
    x = x_n;
    x_n = x - k*(x^2 - 2);
end

sqrt2_aprox = x_n;
```

V následující tabulce pak uvádíme rychlost konvergence (tj. počet potřebných iterací n) pro $\varepsilon = 10^{-8}$ a $x_0 = 1$.

⁹Vyřešte si nerovnici $|f'(x)| < 1$.

¹⁰Platí, že $f'(x) = 1 - 2kx = 0$ pro $x = \frac{1}{2k}$. Grafem kvadratické funkce f je parabola. Vrchol této paraboly má souřadnice $[\frac{1}{2k}, \frac{1}{4k} + 2k]$ a její ramena směřují dolů. Dále platí $f(0) = f(1/k) = 2k$, a proto

$$f((0, 1/k)) = \left(2k, \frac{1}{4k} + 2k\right) \subset (0, 1/k).$$

Volíme-li tedy uzavřený interval $I \subset (0, 1/k)$ tak, aby $(2k, \frac{1}{4k} + 2k) \subset I$ a $x_0 \in I$, pak posloupnost (6.4) leží v I .

k	n	$ \sqrt{2}_{approx} - \sqrt{2} $
1/2	21	$2,02 \cdot 10^{-9}$
1/3	8	$2,48 \cdot 10^{-10}$
1/4	16	$2 \cdot 10^{-9}$

Tedy například pro $k = 1/3$ dostáváme rychlejší konvergenci než v případě $k = 1/4$.

Pro konvergenci i pro $x_0 = 5$ volíme¹¹ například $k = 1/6$, aby interval $(0, 1/k) = (0, 6)$ obsahoval x_0 . Pro $\varepsilon = 10^{-8}$ dostáváme:

k	n	$ \sqrt{2}_{approx} - \sqrt{2} $
1/6	28	$9,83 \cdot 10^{-9}$
1/7	34	$9,65 \cdot 10^{-9}$
1/8	41	$1,32 \cdot 10^{-8}$
1/9	47	$1,89 \cdot 10^{-8}$
1/10	53	$2,35 \cdot 10^{-8}$

Příklad 4 Budeme postupovat podobně jako v řešení příkladu 3. Pevným bodem funkce

$$f(x) := x + \sin x$$

je číslo π . Platí $f'(x) = 1 + \cos x$ a také, že existuje číslo λ takové, že

$$\forall x \in I : |f'(x)| \leq \lambda < 1,$$

kde $I \subset \left(\frac{\pi}{2}, \frac{3}{2}\pi\right)$ je libovolný uzavřený interval. Použijeme-li pak větu na str. 18, dostáváme,¹² že posloupnost (x_n) definovaná předpisem

$$x_{n+1} := f(x_n) = x_n + \sin x_n, \quad n = 0, 1, 2, 3, \dots, \quad (6.5)$$

konverguje k π pro libovolné $x_0 \in \left(\frac{\pi}{2}, \frac{3}{2}\pi\right)$.

Nyní si uvedeme kód metody prostých iterací, který pro dané ε a danou počáteční aproximaci x_0 vydá aproximaci π_{approx} pevného bodu π funkce f . Proměnná n slouží k načítání počtu potřebných iterací.

% inicializace

¹¹Experimentálně získáme dobré výsledky i pro volbu $k = 1/5$ či $k = 1/4$, konvergence metody však již není zaručena větou na str. 18.

¹²Platí

$$f\left(\left(\frac{\pi}{2}, \frac{3}{2}\pi\right)\right) = \left(f\left(\frac{\pi}{2}\right), f\left(\frac{3}{2}\pi\right)\right) = \left(\frac{\pi}{2} + 1, \frac{3}{2}\pi - 1\right) \subset \left(\frac{\pi}{2}, \frac{3}{2}\pi\right).$$

Volíme-li tedy uzavřený interval $I \subset \left(\frac{\pi}{2}, \frac{3}{2}\pi\right)$ tak, aby $\left(\frac{\pi}{2} + 1, \frac{3}{2}\pi - 1\right) \subset I$ a $x_0 \in I$, pak posloupnost (6.5) leží v I .

```

n = 1;
x = x_0;
x_n = x + sin(x)

while abs(x_n - x) >= epsilon
    n = n + 1;
    x = x_n;
    x_n = x + sin(x);
end

pi_aprox = x_n;

```

V následující tabulce pak uvádíme výsledky pro $\varepsilon = 10^{-1}$:

x_0	n	$ \pi_{approx} - \pi $
2	3	$1,51 \cdot 10^{-9}$
3	2	$1,76 \cdot 10^{-11}$
4	3	$8,89 \cdot 10^{-13}$

6.2 Úlohy kapitoly 2

Řešení ke kapitole 2.2

Příklad 1 Nejprve si očísľujme jednotlivé suroviny (ingredience) použité v našich receptech:

1	brambory	16	kuřecí maso	31	rajský protlak
2	bujón	17	lasagne	32	skořice
3	cibule	18	majoránka	33	slanina
4	cukr	19	máslo	34	soda
5	česnek	20	mléko	35	sójová omáčka
6	čočka	21	mouka	36	solamyl
7	droždí	22	mrkev	37	sůľ
8	Hera	23	ocet	38	sýr
9	hořčice	24	olej	39	tatarka
10	hovězí maso	25	oříšky	40	vanilkový cukr
11	jablko	26	pepř	41	vejce
12	jogurt	27	petrželka	42	vepřové maso
13	kakao	28	povidla	43	víno
14	kmín	29	prášek do pečiva	44	worchesterová omáčka
15	kokos	30	rajčata		

Nyní si uvedeme kód pro vytvoření naší databáze receptů. Ta je reprezentována incidenční maticí *inc*, jejíž řádky odpovídají jednotlivým receptům a sloupce odpovídají jednotlivým ingrediencím. Matice *inc* obsahuje na *ij*-té pozici jedničku, pokud *i*-tý recept obsahuje ingredienci s číslem *j*; v opačném případě je na *ij*-té pozici nula.

```
function inc = recepty()
% function inc = recepty()
%
% inc ... matice incidence receptu a surovin

pocet_surovin = 44;
pocet_receptu = 17;

inc = zeros(pocet_receptu,pocet_surovin);

inc(1,[4,7,8,20,21,41]) = 1;
inc(2,[3,16,24,35,36,37,41,44]) = 1;
inc(3,[2,3,5,6,9,18,19,21,22,24,26,27,37,41]) = 1;
inc(4,[4,7,20,21,24,29,37,41]) = 1;
inc(5,[3,16,19,26,33,35,37,38]) = 1;
inc(6,[4,7,14,21,23,24,37]) = 1;
inc(7,[1,3,26,37,39]) = 1;
inc(8,[4,15,20,21,24,29,40,41]) = 1;
inc(9,[5,12,16,21,26,37,38,39,41,43]) = 1;
inc(10,[4,13,20,21,24,29,41]) = 1;
inc(11,[4,11,13,20,21,24,28,29,32,34,37,41]) = 1;
inc(12,[7,21,24,37]) = 1;
inc(13,[3,16,24,33,37]) = 1;
inc(14,[4,7,19,20,21,25,32,37,41]) = 1;
inc(15,[4,8,13,21,25,32,41]) = 1;
inc(16,[3,5,10,17,19,20,21,24,26,30,31,37,38,43]) = 1;
inc(17,[5,18,24,26,31,36,37,41,42,44]) = 1;
```

Dále si uvedeme program, který posoudí blízkost dotazu s recepty v databázi a vypíše recept (popř. recepty), který je nejbližší danému dotazu. V programu je použita projekce do těch složek (surovin), ve kterých je vektor požadavku (dotazu) nenulový.¹³

¹³Uvažujme např. projekci do $n = 4$ složek. Projektovaný recept zapišme jako $r_p = [s_1, s_2, s_3, s_4]$ a projektovaný požadavek jako $p_p = [1, 1, 1, 1]$. Kosínus jejich úhlu pak

```

function blizkost(inc)
% function blizkost(inc)
%
% Zjistuje blizkost pozadavku (vektoru danych surovin)
% a receptu z databaze
%
% inc ... matice incidence projektovana na zadane
% slozky (suroviny)14

% radky matice normujeme15
for i = 1 : size(inc,1)
    norm_radek = norm(inc(i,:));
    if norm_radek > 0
        inc(i,:) = inc(i,:) / norm_radek;
    end
end

% velikost projektovaneho pozadavku
norm_pozadavek = sqrt(size(inc,2));

% urceni blizkosti projektovaneho pozadavku
% a projektovanych receptu
kosinus_uhlu = sum(inc,2) / norm_pozadavek;16

shoda = max(kosinus_uhlu);
indexy = find(kosinus_uhlu == shoda);

disp(['Dotazu nejlepe vyhovuje recept cislo: ', int2str(indexy)])
disp(['Mira shody je: ', num2str(shoda), ' (max: 1, min: 0)'])

```

A konečně dotazy nad naší databází receptů provedeme pomocí následujícího

vypočteme jako

$$\cos \varphi = \frac{r_p p_p}{\|r_p\| \|p_p\|} = \frac{s_1 + s_2 + s_3 + s_4}{\|r_p\| \sqrt{n}}.$$

¹⁴Hledáme-li například recepty obsahující brambory a bujón, projektujeme `inc` na první dva její sloupce.

¹⁵Tj. každý nenulový řádek matice vynásobíme převrácenou hodnotou jeho velikosti. Velikost řádku matice získáme pomocí funkce `norm`.

¹⁶V proměnné `kosinus_uhlu` je uložen vektor všech kosínů úhlů projektovaných receptů a projektovaného požadavku.

skriptu:

```
inc = recepty();

% dotaz 1
suroviny = [4,7,8,13,21,32];
blizkost(inc(:,suroviny))

% dotaz 2
suroviny = [37,38,44];
blizkost(inc(:,suroviny))

% dotaz 3
suroviny = [5,9,16,18,26,33,37];
blizkost(inc(:,suroviny))
```

Uvedme si také výsledky našeho prohledávání:

- dotazu 1 (cukr, droždí, Hera, kakao, mouka a skořice) odpovídá nejlépe recept č. 15 (míra schody je 0,91),
- dotazu 2 (sůl, sýr a worchester) odpovídají nejlépe recepty č. 2, 5, 9, 16 a 17 (míra schody je 0,82),
- dotazu 3 (česnek, hořčice, kuřecí maso, majoránka, pepř, slanina a sůl) odpovídá nejlépe recept č. 3 (míra schody je 0,85).

Příklad 2 Postupujeme podobně jako v příkladu 1. Recepty neobsahující zadané ingredience vyhledáme pomocí negovaných hodnot matice incidence:

```
inc = recepty();

% dotaz 1 (ABKM)
suroviny = [8,12,19,20,38,39];
blizkost(double(~(inc(:,suroviny))))

% dotaz 2 (vegetarianske)
suroviny = [10,16,33,42];
blizkost(double(~(inc(:,suroviny))))
```

Výsledky prohledávání jsou:

-
- mléko, Heru, jogurt, máslo, sýr a tatarskou omáčku neobsahují recepty č. 2, 6, 12, 13 a 17,
 - žádné maso ani slaninu neobsahují recepty č. 1, 3, 4, 6, 7, 8, 10, 11, 12, 14 a 15.

Řešení ke kapitole 2.3

Příklad 1 Uvádíme kód řešení:

```
a = [1,-1]; b = [2,0]; c = [1,1]; % vrcholy trojhelniku abc

k = 2; % stejnolehlost se stredem v [0,0] a koeficientem k
fi = pi / 2; % rotace se stredem v [0,0] a uhlem fi
p = [-3,-2]'; % posunuti s vektorem posunuti p

P = [a' b' c'];

figure; hold on; axis equal; grid on; box on

% vykresleni puvodniho trojhelniku abc
fill(P(1,:),P(2:,:),'red');

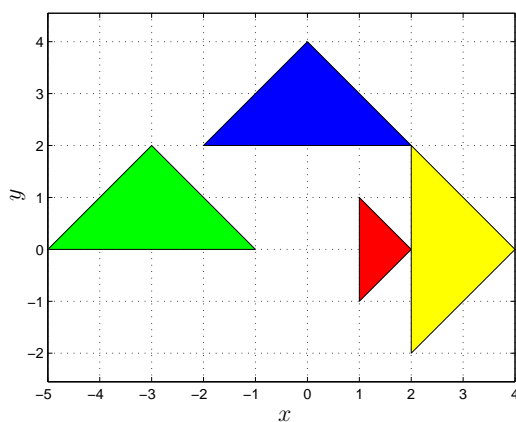
% stejnolehlost
Ps = k * P;17
fill(Ps(1,:),Ps(2:),'yellow');

% rotace
R = [cos(fi),-sin(fi); sin(fi),cos(fi)]; % matice rotace
Pr = R * Ps;
fill(Pr(1,:),Pr(2:),'blue');

% posunuti
Pp = Pr + [p p p];
fill(Pp(1,:),Pp(2:),'green');
```

Na obr. 6.8 je znázorněn grafický výstup programu.

¹⁷Jelikož pro transformační matici platí $T = \begin{bmatrix} k & 0 \\ 0 & k \end{bmatrix} = kI$, kde I je jednotková matice, pak $P_s = TP = kIP = kP$. (Platí totiž $IP = PI = P$.)



Obrázek 6.8: Vzorový trojúhelník (červeně) a jednotlivé obrazy

Příklad 2 Uvedme nejprve kód funkce transformace:

```
function at = transformace(a,k,fi,p)
% function at = transformace(a,k,fi,p)
%
% Funkce transformuje zadany bod a.
%
% a ... a = [x;y] zadany bod
% k ... koeficient stejnolehlosti
% fi ... uhel otoceni
% p ... vektor posunutí
% at ... at = [xt;yt] transformovany bod

% matice rotace
R = [cos(fi),-sin(fi);sin(fi),cos(fi)];

% aplikace transformaci
at = R*(k*a) + p;
```

A dále uvedeme kód transformující podle zadaných parametrů k , fi a p n -úhelník zadaný maticí P .

```
figure; hold on; axis equal; grid on
```

```
% zjisteni poctu vrcholu n-uhelniku
```

```

n = size(P,2);

% prealokace vysledne matice
Pt = zeros(2,n);

% vykresleni vzoru
fill(P(1,:),P(2:),'red');

% aplikace transformaci
for i = 1 : n
    Pt(:,i) = transformace(P(:,i),k,fi,p);
end

% vykresleni vysledku
fill(Pt(1,:),Pt(2:),'green');

```

Příklad 3 Uvádíme program, který pro daný počet iterací n provádí geometrickou transformaci dle tabulky 2.1. Tento program využívá funkci `transformace` z příkladu 2.

```

x = [0.5; 0.5]; % pocatecni bod

figure; hold on;

h = plot(x(1),x(2));

set(h,'markersize',3,'color',[0 0.3 0],'erasemode','none');
axis([-3 3 0 10])

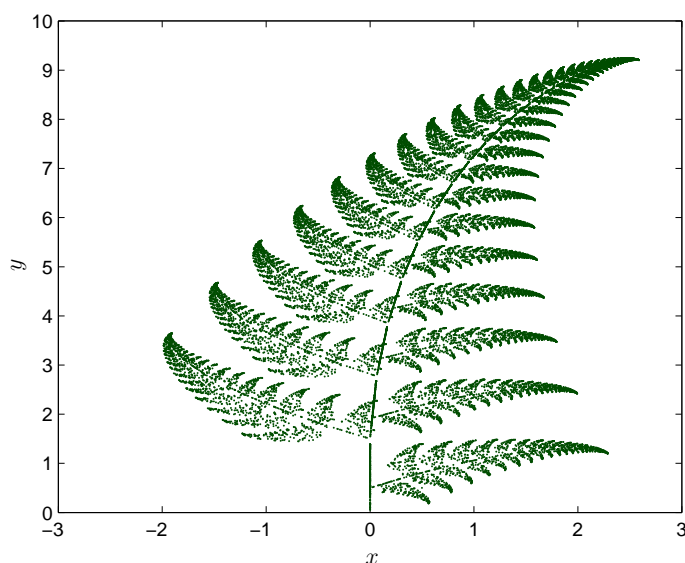
for i = 1 : n
    r = rand;

    if r < 0.85
        x = transformace(x,0.85,-0.05,[0;1.5]);
    elseif r < 0.92
        x = transformace(x,0.3,1,[0;1.5]);
    elseif r < 0.99
        x = transformace(x,0.25,-1,[0;0.5]);
    else
        x = [0 0;0 0.15]*x;
    end
end

```

```
% vykreslení bodu
set(h,'xdata',x(1),'ydata',x(2));
drawnow
end
```

Na obr. 6.9 je znázorněn výsledný obrazec pro 30000 iterací.¹⁸



Obrázek 6.9: Matematická kapradina

6.3 Úlohy kapitoly 3

Řešení ke kapitole 3.2

Příklad 1 Následující kód nalezne přibližně pomocí Eulerovy metody čas v hodinách, po kterém je nutno podat další dávku léku pacientovi trpícím astmatem. Za konstantu c_1 dosadíme pro theofylin 48 ml/min , za v_d dosadíme 35 l , za c_0 volíme $20 \mu\text{g/ml}$ a za c_{dolni} dosadíme $10 \mu\text{g/ml}$. Výsledný čas je v proměnné cas .

¹⁸Jelikož se jedná o transformaci zadanou pomocí pravděpodobnosti, dostaneme i při neměnném počtu iterací při každém spuštění lehce jiný výsledek.

```

tN = 10;19
n = 1e5;20
k = c1*60/1000/v_d;
h = tN / n;
t = 0 : h : tN;
i = 1;
c(1) = c0;
while c(i) >= c_dolni
    c(i+1) = c(i) - h*k*c(i);
    i = i+1;
end
cas = t(i-1);

```

Pro dostatečně velké n (např. pro námi zvolené $n = 10^5$), zjistíme, že další dávku léku musíme pacientovi podat po 8,423 hodinách (tj. cca po 8 hodinách 25 minutách 23 sekundách).

Jinou možností, jak vyřešit náš příklad, je použití vztahu (3.4) pro nalezení hodnot funkce $c(t)$, vykreslení grafu této funkce a následný odhad času, ve kterém je třeba podat další lék (viz obr. 6.10).

Pro úplnost si uveďme ještě analytické řešení naší úlohy. Po dosažení zadaných hodnot dostaneme Cauchyovu úlohu

$$\left. \begin{aligned} c'(t) &= -\frac{2,88}{35}c(t), \\ c(0) &= 20. \end{aligned} \right\} \quad (6.6)$$

Derivováním se lze přesvědčit, že řešením úlohy (6.6) je funkce

$$\bar{c}(t) = 20e^{-\frac{2,88}{35}t}.$$

Z této funkce sestavíme rovnici s neznámou t , která popisuje čas do další aplikace léku

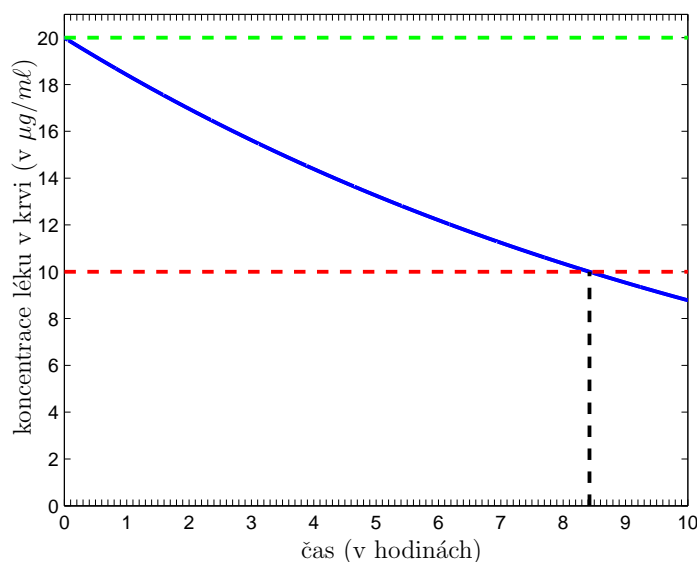
$$10 = 20e^{-\frac{2,88}{35}t}.$$

Po další úpravě získáme rovnici

$$\ln \frac{1}{2} = -\frac{2,88}{35}t.$$

¹⁹Čas tN jsme „uhádli“ tak, aby příliš nepřevyšoval potřebný čas pro podání léku. Toto „hádání“ se dá vyřešit např. prvotním spuštěním tohoto programu s dostatečně velkým tN .

²⁰Poznamenejme, že n je rozumné volit dostatečně velké, protože h je nepřímo úměrné n . V kapitole 3.1 jsme si uvedli, že chyba aproximace řešení pomocí Eulerovy metody závisí přímo úměrně na kroku h .



Obrázek 6.10: Pokles koncentrace theofylinu v krvi v závislosti na čase

Z ní už snadno vypočítáme, že další dávku theofylinu je nutno podat za přibližně 8,424 hodin, tedy za 8 hodin 25 minut 26 sekund.

V následující tabulce si ještě ukážeme, jak závisí chyba aproximace koncentrace theofylinu v čase $T = 8,423$ hodin získaná Eulerovou metodou na volbě n , a tuto chybu porovnáme s odhadem (3.5). Vyčíslení odhadu (3.5) je náročnější, a proto v tabulce uvádíme jen hodnotu tohoto výrazu a podrobný výpočet přenecháme zvědavému čtenáři.

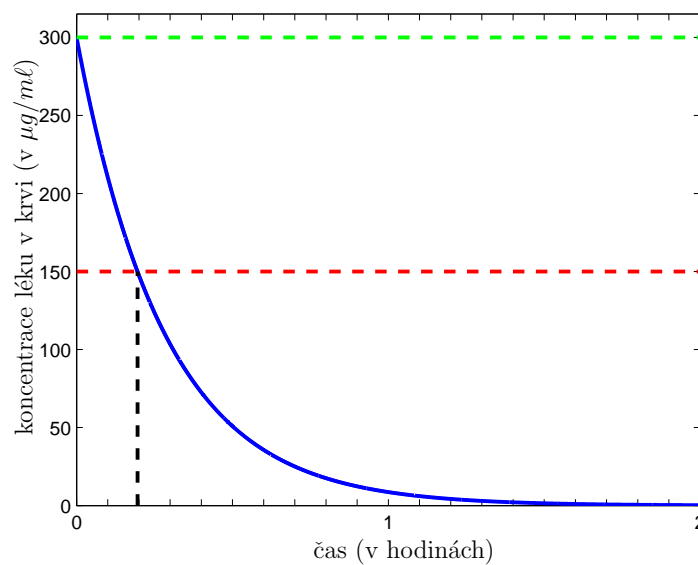
n	h	$ c_{approx}(T) - \bar{c}(T) $	$\frac{M}{2K} (e^{K(T-t_0)} - 1) h$
10^4	10^{-2}	$2,9 \cdot 10^{-3}$	$8,2 \cdot 10^{-3}$
10^5	10^{-3}	$2,9 \cdot 10^{-4}$	$8,2 \cdot 10^{-4}$
10^6	10^{-4}	$2,9 \cdot 10^{-5}$	$8,2 \cdot 10^{-5}$

Příklad 2 Pomocí stejného kódu jako v příkladu 1 najdeme přibližně čas v hodinách, po kterém je nutno podat další dávku léku pacientovi s horečkami, bolestmi hlavy, svalů či kloubů. Za konstantu $c1$ dosadíme pro kyselinu acetylsalicylovou 650 ml/min , za v_d dosadíme 11 l , za $c0$ volíme 300 µg/ml a za c_{dolni} dosadíme 150 µg/ml . Výsledný čas je opět v proměnné cas . Jedinou změnu provedeme v nastavení proměnné tN .

$t_N = 2;$ ²¹
 \vdots
 $\text{cas} = t(i-1);$

Pro dostatečně velké n (např. pro námi zvolené $n = 10^5$), zjistíme, že další dávku léku musíme pacientovi podat po 0,195 hodinách (tj. cca po 11 minutách 42 sekundách).

Jinou možností, jak vyřešit náš příklad, je použití vztahu (3.4) pro nalezení hodnot funkce $c(t)$, vykreslení grafu této funkce a následný odhad času, ve kterém je třeba podat další lék (viz obr. 6.11).



Obrázek 6.11: Pokles koncentrace kyseliny acetylsalicylové v krvi v závislosti na čase

Nakonec si uveďme opět analytické řešení naší úlohy. Po dosazení zadaných hodnot dostaneme Cauchyovu úlohu

$$\left. \begin{aligned} c'(t) &= -\frac{39}{11}c(t), \\ c(0) &= 300. \end{aligned} \right\} \quad (6.7)$$

Derivováním se lze přesvědčit, že řešením úlohy (6.7) je funkce

$$\bar{c}(t) = 300e^{-\frac{39}{11}t}.$$

²¹Čas t_N jsme opět „uhádli“ tak, aby příliš nepřevyšoval potřebný čas pro podání léku.

Z této funkce sestavíme rovnici s neznámou t , která popisuje čas do další aplikace léku

$$150 = 300e^{-\frac{39}{11}t}.$$

Po další úpravě získáme rovnici

$$\ln \frac{1}{2} = -\frac{39}{11}t.$$

Z ní už snadno vypočítáme, že další dávku kyseliny acetylsalicylové je nutno podat za přibližně 0,196 hodin, tedy za 11 minut 46 sekund.

V následující tabulce si ještě ukážeme, jak závisí chyba aproximace koncentrace kyseliny acetylsalicylové v čase $T = 0,195$ hodin získaná Eulerovou metodou na volbě n , a tuto chybu porovnáme s odhadem (3.5). Vyčíslení odhadu (3.5) je náročnější, a proto v tabulce uvádíme jen hodnotu tohoto výrazu a podrobný výpočet přenecháme zvědavému čtenáři.

n	h	$ c_{approx}(T) - \bar{c}(T) $	$\frac{M}{2K} (e^{K(T-t_0)} - 1) h$
10^4	10^{-2}	1,9	5,1
10^5	10^{-3}	$1,9 \cdot 10^{-1}$	$5,3 \cdot 10^{-1}$
10^6	10^{-4}	$1,8 \cdot 10^{-2}$	$5,3 \cdot 10^{-2}$

Řešení ke kapitole 3.3

Příklad 1 Abychom numericky vyřešili náš příklad, budeme se nyní zabývat úpravou Eulerovy metody, kterou jsme se zabývali v kapitole 3.1. Ale zatímco rekurzivní vztah (3.4) nám umožňuje nalézt hodnoty funkce y pro časy $t > t_0$, potřebujeme nyní odvodit obdobný vztah pro časy $t < t_0$. Hodnota t_0 je pro nás čas, ve kterém známe hodnotu hledané funkce popsané diferenciální rovnicí. Např. v našem případě je t_0 rok 1967, ve kterém jsme změřili počet atomů olova-210 v gramu rudy v barvě na zkoumaném obrazu.

K řešení úlohy na intervalu $\langle t_{-N}, t_0 \rangle$ použijeme opět aproximaci derivace funkce y v bodě t pomocí tzv. diference y v tomto bodě. Ale místo tzv. dopředné diference, kterou jsme použili v kapitole 3.1, nyní použijeme tzv. zpětnou diferenci

$$y'(t) \approx \frac{y(t) - y(t-h)}{h},$$

kde h je „malé“ a kladné.

Po úpravě dostaneme

$$y(t-h) \approx y(t) - hy'(t).$$

Použijeme-li diferenciální rovnici z (3.2), získáme vztah

$$y(t-h) \approx y(t) - hf(t, y(t)). \quad (6.8)$$

Zvolme „dostatečně malou“ pevnou velikost $h > 0$ a sestrojme posloupnost

$$t_0, t_{-1} := t_0 - h, t_{-2} := t_0 - 2h, \dots, t_{-N} := t_0 - Nh.$$

Označme pomocí y_n aproximaci hodnoty přesného řešení $y(t_n)$. Po přidání počáteční podmínky dostáváme z (6.8) rekurzivní vztah

$$\left. \begin{aligned} y_0 &= y(t_0), \\ y_{n-1} &= y_n - hf(t_n, y_n), \quad n = 0, -1, \dots, -N. \end{aligned} \right\} \quad (6.9)$$

Pomocí následujícího kódu lze zjistit, zda je možné, aby byl obraz Emauzští učedníci opravdu cca 300 let starý. Za konstantu N_0 dosadíme $1,42 \cdot 10^8$ a za r dosadíme 420480. Pokud není splněna nerovnost $N(n+1) \leq 5e11$, pak obraz nemůže být starý 300 let. V tom případě můžeme ověřit, zda je možné, aby vznikl před 20 lety. K tomu stačí použít stejný kód s jedinou úpravou, při které za tN dosadíme místo hodnoty 300 hodnotu 20.

```
p_r = 22;22
lambda = log(2)/(p_r);

n = 1e5;23
tN = 300;
h = tN / n;
N(1) = N_0;

for i = 1:n
    N(i+1) = N(i) - h*(-lambda*N(i) + r);
end

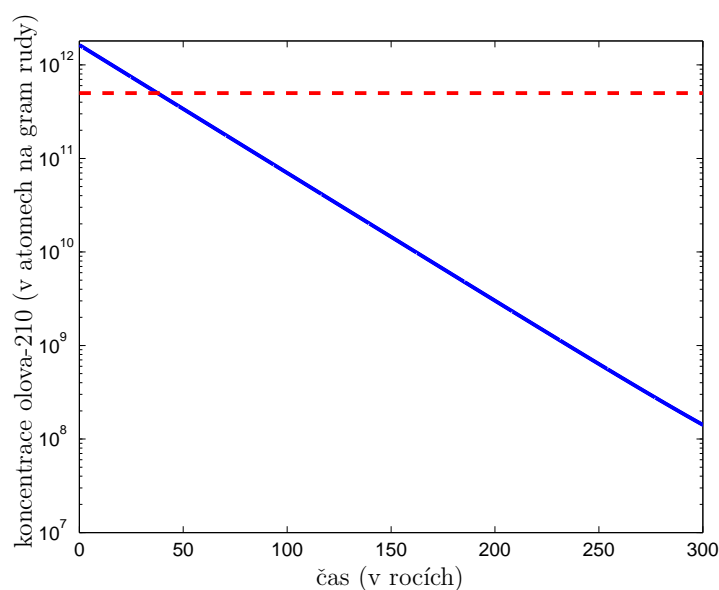
if N(n+1) <= 5e11
    disp(['Obraz je pravdepodobne stary 300 let.'])
    return
end
```

²²Konstanta p_r popisuje poločas rozpadu olova-210.

²³Poznamenejme, že n je rozumné volit dostatečně velké, protože h je nepřímo úměrné n . V kapitole 3.1 jsme si uvedli, že chyba aproximace řešení pomocí Eulerovy metody závisí přímo úměrně na kroku h .

Pro uvažovaný obraz bychom pomocí uvedeného kódu zjistili, že obraz nemůže být starý 300 let, protože v tom případě by $N(-300) \doteq 1,6375 \cdot 10^{12}$ a to není možné. Obraz je tedy dle uvedených faktů v kapitole 3.3 starý přibližně 20 let. Pro takové stáří získáme $N(-20) \doteq 2,5494 \cdot 10^8$, což je v souladu s běžnými koncentracemi atomů olova-210.

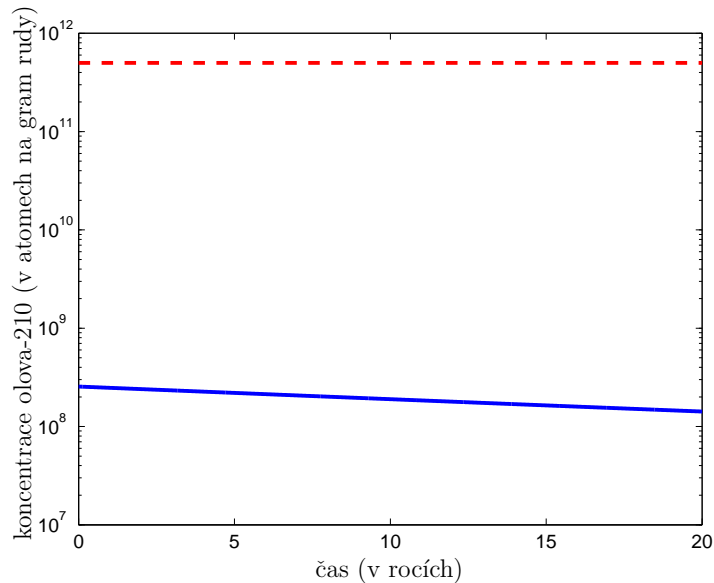
Další možností, jak vyřešit náš příklad je použití vztahu (6.9) pro nalezení hodnot funkce $N(t)$, kde $t < t_0$, vykreslení grafu této funkce a následný odhad, zda je možné, aby byl obraz starý 300 či jen 20 let. Z obr. 6.12 plyne, že obraz nemůže být starý 300 let. Zatímco z obr. 6.13 je vidět, že obraz 20 let mít může. Stupnice na svislé ose je pro lepší přehlednost logaritmická. Červená čára na obou obrázcích označuje hodnotu $5 \cdot 10^{11}$. Připomeňme, že je naprosto nemožné, aby počet atomů olova-210 v gramu rudy, ze které se oxid olovnatý vyrobil, přesáhl tuto hodnotu.



Obrázek 6.12: Pokles koncentrace radioaktivního olova-210 v gramu rudy v závislosti na čase

Pro úplnost si uveďme ještě analytické řešení naší úlohy. Po dosazení zadaných hodnot a využitím faktu, že poločas rozpadu olova-210 je 22 let, dostaneme Cauchyovu úlohu

$$\left. \begin{aligned} N'(t) &= -0,0315N(t) + 420480, \\ N(0) &= 1,42 \cdot 10^8. \end{aligned} \right\} \quad (6.10)$$



Obrázek 6.13: Pokles koncentrace radioaktivního olova-210 v gramu rudy v závislosti na čase

Derivováním si můžeme ověřit, že řešením rovnice (6.10) je funkce

$$\bar{N}(t) = \left(1,42 \cdot 10^8 - \frac{420480}{0,0315} \right) e^{-0,0315t} + \frac{420480}{0,0315}.$$

Zkusme s využitím řešení úlohy (6.10) určit, zda je možné, aby byl obraz pravý. Pokud je $t = -300$ roků, pak

$$\bar{N}(-300) \doteq 1,6383 \cdot 10^{12}.$$

Z toho je zřejmé, že obraz je podvrh. Pokud zvolíme $t = -20$ roků, pak

$$\bar{N}(-20) \doteq 2,5494 \cdot 10^8.$$

To odpovídá běžným koncentracím atomů olova-210.

V následující tabulce si ještě ukážeme, jak závisí chyba aproximace počtu atomů olova-210 $N(t)$ v čase vytvoření obrazu získaná Eulerovou metodou na volbě n , a tuto chybu porovnáme s odhadem (3.5) (tento odhad platí i pro zpětné diference). Vyčíslení odhadu (3.5) je náročnější, a proto v tabulce uvádíme jen hodnotu tohoto výrazu a podrobný výpočet přenecháme zvědavému čtenáři.

n	h	$ N_{approx}(-20) - \bar{N}(-20) $	$\frac{M}{2K} (e^{K(t_0 - (-20))} - 1) h$
10^4	$2 \cdot 10^{-3}$	4796,2	6682,0
10^5	$2 \cdot 10^{-4}$	479,6	668,2
10^6	$2 \cdot 10^{-5}$	47,96	66,82

Příklad 2 Pro numerické řešení našeho příkladu použijeme obdobné úpravy Eulerovy metody jako v předcházejícím příkladu, tj. použijeme rekurzivní vztah (6.9).

Pomocí stejného kódu jako v příkladu 1 můžeme ověřit, zda je možné, aby byl obraz Krajčářka opravdu cca 300 let starý. Za konstantu N_0 dosadíme $0,25 \cdot 10^8$ a za r dosadíme 735840. Pokud by nebyla splněna nerovnost $N(n+1) \leq 5e11$, pak obraz nemůže být starý 300 let. V tom případě můžeme ověřit, zda je možné, aby vznikl před 20 lety. K tomu stačí použít stejný kód s jedinou úpravou, při které za tN dosadíme místo hodnoty 300 hodnotu 20.

Pro uvažovaný obraz bychom pomocí uvedeného kódu zjistili, že obraz může být starý 300 let, protože $N(-300) \doteq 2,096 \cdot 10^{10}$ a to je v souladu s běžnými koncentracemi atomů olova-210.

Další možností, jak vyřešit náš příklad, je použití vztahu (6.9) pro nalezení hodnot funkce $N(t)$, kde $t < t_0$, vykreslení grafu této funkce a následný odhad, zda je možné, aby byl obraz starý 300 či jen 20 let. Z obr. 6.14 plyne, že obraz může být starý 300 let. Stupnice na svislé ose je pro lepší přehlednost logaritmická. Červená čára na obrázku označuje hodnotu $5 \cdot 10^{11}$.

Pro úplnost si uveďme ještě analytické řešení naší úlohy. Po dosazení zadaných hodnot a využitím faktu, že poločas rozpadu olova-210 je 22 let, dostaneme Cauchyovu úlohu

$$\left. \begin{aligned} N'(t) &= -0,0315N(t) + 735840, \\ N(T) &= 0,25 \cdot 10^8. \end{aligned} \right\} \quad (6.11)$$

Derivováním si můžeme ověřit, že řešením rovnice (6.11) je funkce

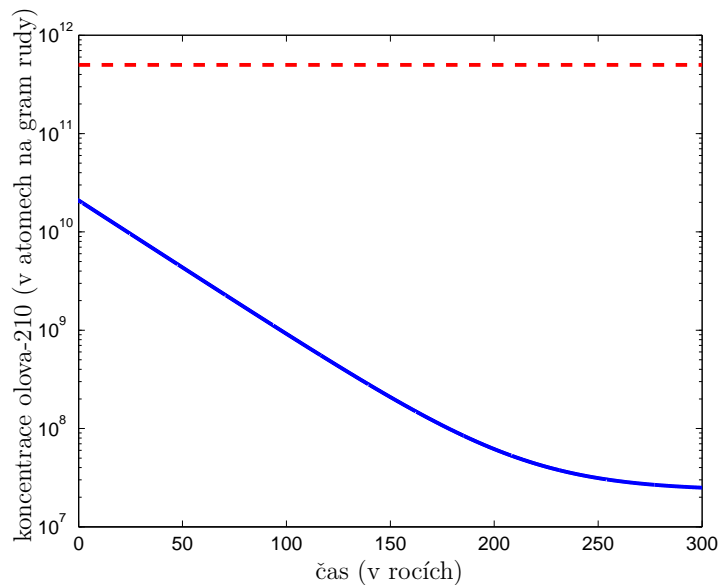
$$\bar{N}(t) = \left(0,25 \cdot 10^8 - \frac{735840}{0,0315} \right) e^{-0,0315t} + \frac{735840}{0,0315}.$$

Zkusme s využitím řešení úlohy (6.11) určit, zda je možné, aby byl obraz pravý. Pokud je $t = -300$ roků, pak

$$\bar{N}(-300) \doteq 2,097 \cdot 10^{10}.$$

Z toho je zřejmé, že obraz může být skutečně pravý.

V následující tabulce si ještě ukážeme, jak závisí chyba aproximace počtu atomů olova-210 $N(t)$ v čase vytvoření obrazu získaná Eulerovou metodou



Obrázek 6.14: Pokles koncentrace radioaktivního olova-210 v gramu rudy v závislosti na čase

na volbě n , a tuto chybu porovnáme s odhadem (3.5) (tento odhad platí i pro zpětné diference). Vyčíslení odhadu (3.5) je náročnější, a proto v tabulce uvádíme jen hodnotu tohoto výrazu a podrobný výpočet přenecháme zvědavému čtenáři.

n	h	$ N_{\text{aprox}}(-300) - \bar{N}(-300) $	$\frac{M}{2K} (e^{K(t_0 - (-300))} - 1) h$
10^4	$3 \cdot 10^{-2}$	$9,3 \cdot 10^7$	$1,3 \cdot 10^{11}$
10^5	$3 \cdot 10^{-3}$	$9,4 \cdot 10^6$	$1,3 \cdot 10^{10}$
10^6	$3 \cdot 10^{-4}$	$9,4 \cdot 10^5$	$1,3 \cdot 10^9$

Řešení ke kapitole 3.4

Příklad 1 Následující kód odhadne pomocí Eulerovy metody vývoj populace USA v letech 1790–1950. Za konstantu a dosadíme počátek zkoumaného intervalu, tedy rok 1790, a za konstantu b dosadíme konec zkoumaného intervalu, tedy rok 1950. Samotné hodnoty odhadované velikosti obyvatelstva USA jsou v proměnné y .

```

y0 = 3929000;

n = 1e5;24
h = (b-a)/n;
t = a : h : b;
y(1) = y0;

for i = 1 : n
    y(i+1) = y(i) + h*(0.03134*y(i) - 1.5887e-10*y(i)^2);
end

```

Pomocí tohoto kódu jsme našli aproximace počtů obyvatel v letech 1800, 1850, 1900 a 1950, které si můžete prohlédnout v následující tabulce.

Rok	Počet obyvatel	Odhad počtu obyvatel Eulerovou metodou
1790	3 929 000	3 929 000
1800	5 308 000	5 335 985
1850	23 192 000	23 191 645
1900	75 995 000	76 868 359
1950	150 697 000	148 675 121

Tabulka 6.1: Populace USA v letech 1790–1950 a odhady její velikosti Eulerovou metodou

Vývoj populace v letech 1790–1950 odhadnutý Eulerovou metodou si můžete prohlédnout na obr. 6.15.

Pro úplnost si uveďme ještě analytické řešení naší úlohy. Po dosazení zadaných hodnot dostaneme Cauchyovu úlohu

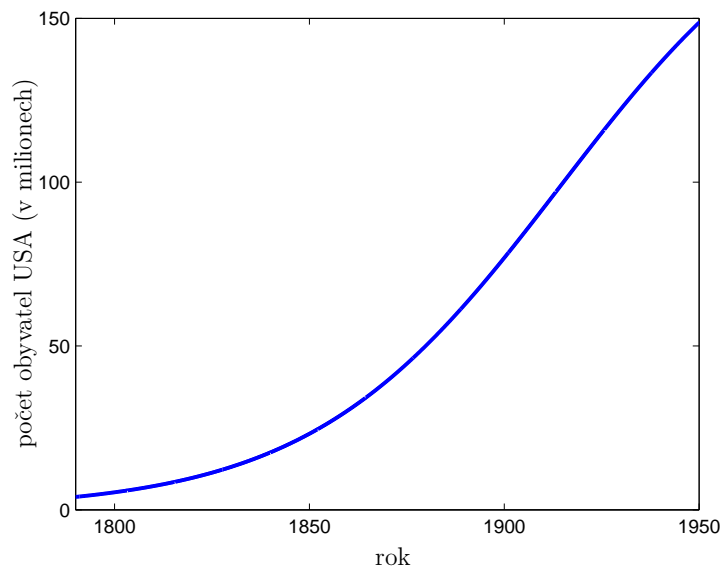
$$\left. \begin{aligned} y'(t) &= 0,03134y(t) - 1,5887 \cdot 10^{-10}y^2(t), \\ y(1790) &= 3\,929\,000. \end{aligned} \right\} \quad (6.12)$$

Můžeme si derivováním ověřit, že řešením úlohy (6.12) je funkce

$$\bar{y}(t) = \frac{123134,86}{6,2420023 \cdot 10^{-4} + 0,03071579977e^{-0,03134(t-1790)}}.$$

Po dosazení dostaneme $\bar{y}(1800) \doteq 5\,336\,024$, $\bar{y}(1850) \doteq 23\,192\,502$, $\bar{y}(1900) \doteq 76\,871\,300$ a $\bar{y}(1950) \doteq 148\,677\,189$, což jsou velmi dobré odhady skutečných hodnot.

²⁴Poznamenejme, že n je rozumné volit dostatečně velké, protože h je nepřímo úměrné n . V kapitole 3.1 jsme si uvedli, že chyba aproximace řešení pomocí Eulerovy metody závisí přímo úměrně na kroku h .



Obrázek 6.15: Odhad počtu obyvatel USA v letech 1790–1950

Nabízí se přirozeně nápad, použít výše uvedený kód pro odhad počtu obyvatel USA i po roce 1950, např. pro rok 2014. Pokud bychom to udělali, zjistili bychom, že odhad 188 700 000 obyvatel USA, který bychom získali, je velmi špatný, a je velmi vzdálený skutečnému počtu přibližně 319 000 000 obyvatel USA v tomto roce. To je způsobeno nepřesností modelu, který nebere v úvahu obrovský nárůst legální (a pravděpodobně i ilegální) imigrace do USA. Vždyť zatímco na přelomu 40. a 50. let byla každoroční legální imigrace do USA kolem 200 000 lidí, tak v 80. letech to již bylo mezi 500 000 až 600 000 lidmi, od 90. let je již legální imigrace kolem 1 000 000 ročně a od roku 2005 již legální imigrace neklesla nikdy pod 1 000 000 obyvatel.

V následující tabulce si ještě ukážeme, jak závisí chyba aproximace počtu obyvatel USA v roce 1950 získaná Eulerovou metodou na volbě n , a tuto chybu porovnáme s odhadem (3.5). Vycíslení odhadu (3.5) je náročnější, a proto v tabulce uvádíme jen hodnotu tohoto výrazu a podrobný výpočet přenecháme zvědavému čtenáři.

n	h	$ y_{approx}(1950) - \bar{y}(1950) $	$\frac{M}{2K} (e^{K(1950-t_0)} - 1) h$
10^4	$1,6 \cdot 10^{-2}$	$2,1 \cdot 10^4$	$6,1 \cdot 10^5$
10^5	$1,6 \cdot 10^{-3}$	$2,1 \cdot 10^3$	$6,1 \cdot 10^4$
10^6	$1,6 \cdot 10^{-4}$	$2,1 \cdot 10^2$	$6,1 \cdot 10^3$

Řešení ke kapitole 3.5

Příklad 1 Následující kód nalezne přibližně pomocí Eulerovy metody čas v minutách, který je potřebný pro ochlazení kávy ze 100°C na teplotu menší než 50°C. Za T_{k0} volíme 100°C, za konstantu k dosadíme 0,04 a za T_v volíme 20°C. Výsledný čas je v proměnné `cas`.

```
tN = 60;25
n = 1e5;26
h = tN / n;
t = 0 : h : tN;
i = 1;
T_k(1) = T_k0;
while T_k(i) >= 50
    T_k(i+1) = T_k(i) + h*(k*(T_v - T_k(i)));
    i = i + 1;
end
cas = t(i);
```

Pro dostatečně velké n , zjistíme, že se káva ochladí pod 50°C za 24,522 minut.

Jinou možností, jak vyřešit náš příklad, je použití vztahu (3.4) pro nalezení hodnot funkce $T_k(t)$, vykreslení grafu této funkce a následný odhad času, který je potřeba, než se káva ochladí na požadovanou teplotu (viz obr. 6.16).

Uveďme si ještě analytické řešení naší úlohy. Po dosazení uvedených hodnot dostaneme Cauchyovu úlohu

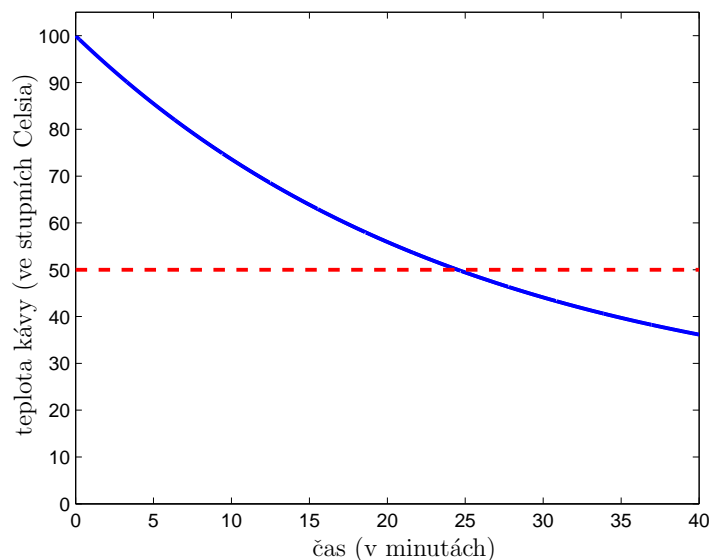
$$\left. \begin{aligned} T'_k(t) &= 0,04(20 - T_k(t)), \\ T_k(0) &= 100. \end{aligned} \right\} \quad (6.13)$$

Derivováním lze snadno ověřit, že řešením rovnice (6.13) je funkce $\overline{T}_k(t) = 80e^{-0,04t} + 20$. Z toho plyne

$$50 = 80e^{-0,04t} + 20.$$

²⁵Čas tN jsme „uhádli“ tak, aby příliš nepřevyšoval potřebný čas pro ochlazení kávy. Toto „hádání“ se dá vyřešit např. prvotním spuštěním tohoto programu s dostatečně velkým tN .

²⁶Poznamenejme, že n je rozumné volit dostatečně velké, protože h je nepřímo úměrné n . V kapitole 3.1 jsme si uvedli, že chyba aproximace řešení pomocí Eulerovy metody závisí přímo úměrně na kroku h .



Obrázek 6.16: Pokles teploty kávy v závislosti na čase

To znamená, že káva se ochladí pod 50°C za přibližně 24,521 minut.

V následující tabulce si ještě ukážeme, jak závisí chyba aproximace teploty kávy v čase $T = 24,522$ minut získaná Eulerovou metodou na volbě n , a tuto chybu porovnáme s odhadem (3.5). Vyčíslení odhadu (3.5) je náročnější, a proto v tabulce uvádíme jen hodnotu tohoto výrazu a podrobný výpočet přenecháme zvědavému čtenáři.

n	h	$ T_{k\text{ approx}}(T) - \overline{T}_k(T) $	$\frac{M}{2K} (e^{K(T-t_0)} - 1) h$
10^4	$6 \cdot 10^{-2}$	$3,5 \cdot 10^{-2}$	$1,6 \cdot 10^{-1}$
10^5	$6 \cdot 10^{-3}$	$3,5 \cdot 10^{-3}$	$1,6 \cdot 10^{-2}$
10^6	$6 \cdot 10^{-4}$	$3,5 \cdot 10^{-4}$	$1,6 \cdot 10^{-3}$

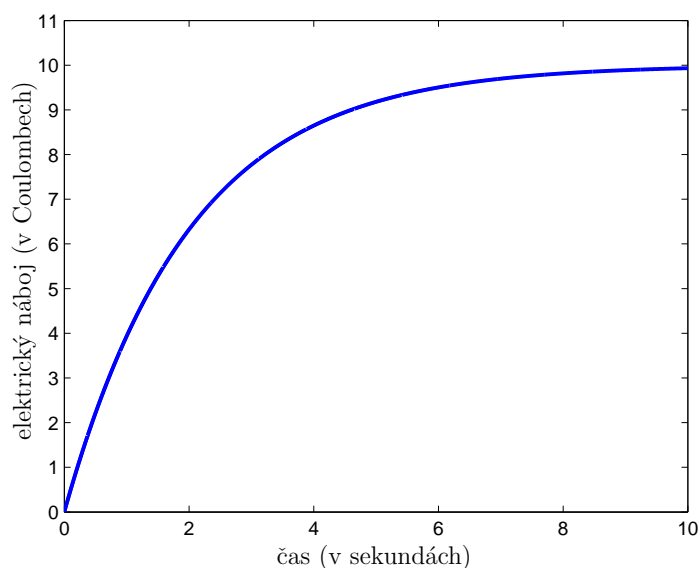
Řešení ke kapitole 3.6

Příklad 1 Následující kód slouží k získání aproximace průběhu elektrického náboje v elektrickém obvodu na obr. 3.5, který sledujeme v časovém intervalu $\langle 0, 10 \rangle$ s. Za konstantu **a** dosadíme počátek zkoumaného intervalu, tedy 0, a za konstantu **b** dosadíme konec zkoumaného intervalu, tedy 10. Samotné hodnoty odhadovaného elektrického náboje jsou v proměnné **q**.

```
q0 = 0;
E = 10;
R = 2;
C = 1;
n = 1e5;27
h = (b-a) / n;
t = a : h : b;
q(1) = q0;

for i = 1 : n
    q(i+1) = q(i) + h * (1/R * (E - 1/C * q(i)));
end
```

Průběh elektrického náboje odhadnutý Eulerovou metodou si můžete prohlédnout na obr. 6.17.



Obrázek 6.17: Odhad průběhu elektrického náboje v elektrickém obvodu

Uvedme si ještě analytické řešení naší úlohy. Po dosazení zadaných hod-

²⁷Poznamenejme, že n je rozumné volit dostatečně velké, protože h je nepřímo úměrné n . V kapitole 3.1 jsme si uvedli, že chyba aproximace řešení pomocí Eulerovy metody závisí přímo úměrně na kroku h .

not dostaneme Cauchyovu úlohu

$$\left. \begin{aligned} 2q'(t) + q(t) &= 10, \\ q(0) &= 0. \end{aligned} \right\} \quad (6.14)$$

Pomocí derivování můžeme ověřit, že řešením úlohy (6.14) je funkce

$$\bar{q}(t) = -10e^{-\frac{1}{2}t} + 10.$$

Graf této funkce odpovídá průběhu na obr. 6.17.

V následující tabulce si ještě ukážeme, jak závisí chyba aproximace elektrického náboje v čase $t = 10$ sekund získaná Eulerovou metodou na volbě n , a tuto chybu porovnáme s odhadem (3.5). Vyčíslení odhadu (3.5) je náročnější, a proto v tabulce uvádíme jen hodnotu tohoto výrazu a podrobný výpočet přenecháme zvědavému čtenáři.

n	h	$ q_{approx}(10) - \bar{q}(10) $	$\frac{M}{2K} (e^{K(10-t_0)} - 1) h$
10^4	10^{-3}	$8,4 \cdot 10^{-5}$	$4,0 \cdot 10^{-3}$
10^5	10^{-4}	$8,4 \cdot 10^{-6}$	$4,0 \cdot 10^{-4}$
10^6	10^{-5}	$8,4 \cdot 10^{-7}$	$4,0 \cdot 10^{-5}$

Příklad 2 Pomocí stejného kódu jako v příkladu 1 získáme aproximaci průběhu elektrického náboje v elektrickém obvodu na obr. 3.5, který sledujeme v časovém intervalu $(0, 10)$ s. Za konstanty a a b opět dosadíme počátek a konec zkoumaného intervalu, tedy 0 a 10. Samotné hodnoty odhadovaného elektrického náboje jsou zase v proměnné q . Jedinou změnu provedeme v nastavení konstanty E a počáteční podmínky q_0 .

$$q_0 = 10;$$

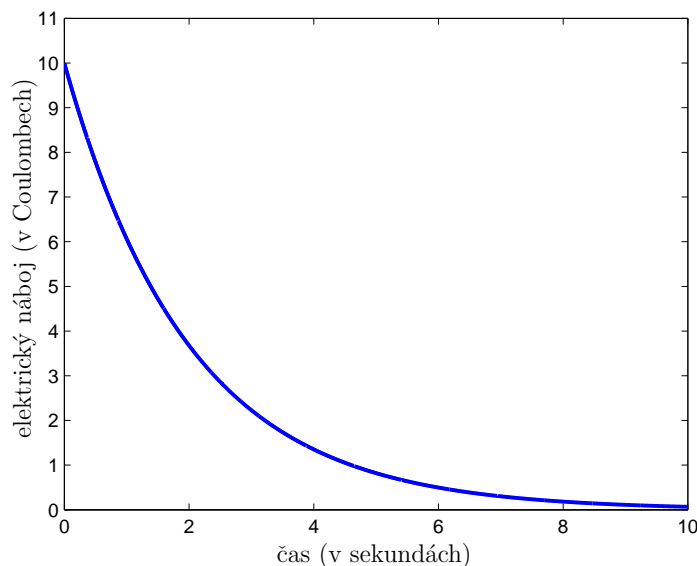
$$E = 0;$$

⋮

Průběh elektrického náboje odhadnutý Eulerovou metodou si můžete prohlédnout na obr. 6.18.

Uveďme si opět ještě analytické řešení naší úlohy. Po dosazení zadaných hodnot dostaneme Cauchyovu úlohu

$$\left. \begin{aligned} 2q'(t) + q(t) &= 0, \\ q(0) &= 10. \end{aligned} \right\} \quad (6.15)$$



Obrázek 6.18: Odhad průběhu elektrického náboje v elektrickém obvodu

Pomocí derivování můžeme ověřit, že řešením úlohy (6.15) je funkce

$$\bar{q}(t) = 10e^{-\frac{1}{2}t}.$$

Graf této funkce odpovídá průběhu na obr. 6.18.

V následující tabulce si ještě ukážeme, jak závisí chyba aproximace elektrického náboje v čase $t = 10$ sekund získaná Eulerovou metodou na volbě n , a tuto chybu porovnáme s odhadem (3.5). Vyčíslení odhadu (3.5) je náročnější, a proto v tabulce uvádíme jen hodnotu tohoto výrazu a podrobný výpočet přenecháme zvědavému čtenáři.

n	h	$ q_{approx}(10) - \bar{q}(10) $	$\frac{M}{2K} (e^{K(10-t_0)} - 1) h$
10^4	10^{-3}	$8,4 \cdot 10^{-5}$	$3,7 \cdot 10^{-3}$
10^5	10^{-4}	$8,4 \cdot 10^{-6}$	$3,7 \cdot 10^{-4}$
10^6	10^{-5}	$8,4 \cdot 10^{-7}$	$3,7 \cdot 10^{-5}$

Řešení ke kapitole 3.7

Příklad 1 Následující kód slouží k získání aproximace průběhu výchylek kyvadla v časovém intervalu $\langle 0, 10 \rangle$ s. Za konstantu a dosadíme počátek zkoumaného intervalu, tedy 0, a za konstantu b dosadíme konec zkoumaného

intervalu, tedy 10. Samotné hodnoty odhadované výchyly jsou v proměnné y .

```
y0 = 0.1;
z0 = 0;
g = 10;
l = 1;
n = 1e5;28
h = (b-a) / n;
t = a : h : b;
y(1) = y0;
z(1) = z0;

for i = 1 : n
    y(i+1) = y(i) + h * z(i);
    z(i+1) = z(i) + h * ((-g/l)*y(i));
end
```

Průběh výchylek odhadnutý Eulerovou metodou si můžete prohlédnout na obr. 6.19.

Uveďme si ještě analytické řešení naší úlohy. Po dosazení zadaných hodnot dostaneme Cauchyovu úlohu

$$\left. \begin{aligned} y''(t) &= -10y(t), \\ y(0) &= 0,1, \quad y'(0) = 0. \end{aligned} \right\} \quad (6.16)$$

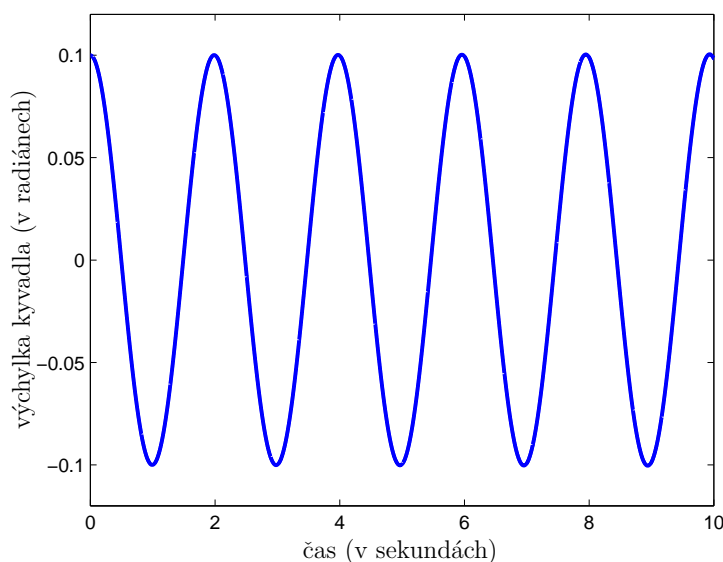
Pomocí derivování můžeme ověřit, že řešením úlohy (3.18) je funkce

$$\bar{y}(t) = 0,1 \cos(\sqrt{10}t).$$

Graf této funkce odpovídá průběhu na obr. 6.19.

V následující tabulce si ještě ukážeme, jak závisí chyba aproximace výchyly kyvadla v čase $t = 10$ sekund získaná Eulerovou metodou na volbě n , a tuto chybu porovnáme s odhadem (3.5). Vyčíslení odhadu (3.5) je náročnější, a proto v tabulce uvádíme jen hodnotu tohoto výrazu a podrobný výpočet přenecháme zvědavému čtenáři.

²⁸Poznamenejme, že n je rozumné volit dostatečně velké, protože h je nepřímo úměrné n . V kapitole 3.1 jsme si uvedli, že chyba aproximace řešení pomocí Eulerovy metody závisí přímo úměrně na kroku h .



Obrázek 6.19: Odhad průběhu výchylek kyvadla

n	h	$ y_{aprox}(10) - \bar{y}(10) $	$\frac{M}{2K} (e^{K(10-t_0)} - 1) h$
10^4	10^{-3}	$5,0 \cdot 10^{-3}$	3,7
10^5	10^{-4}	$4,5 \cdot 10^{-4}$	$3,5 \cdot 10^{-1}$
10^6	10^{-5}	$4,9 \cdot 10^{-5}$	$3,5 \cdot 10^{-2}$

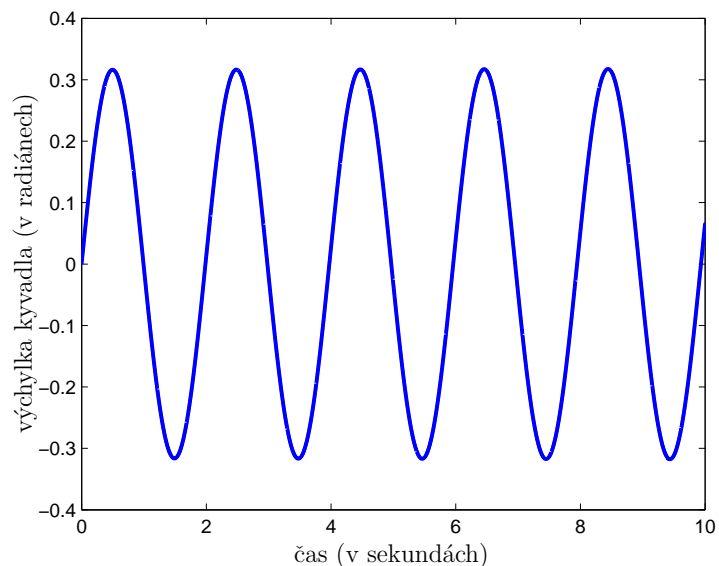
Příklad 2 Pomocí stejného kódu jako v příkladu 1 získáme aproximaci průběhu výchylek kyvadla v časovém intervalu $\langle 0, 10 \rangle$ s. Za konstanty a a b opět dosadíme počátek a konec zkoumaného intervalu, tedy 0 a 10. Samotné hodnoty odhadované výchylky jsou zase v proměnné y . Jedinou změnu provedeme v nastavení počátečních podmínek y_0 a z_0 .

```
y0 = 0;
z0 = 1;
:
```

Průběh výchylek odhadnutý Eulerovou metodou si můžete prohlédnout na obr. 6.20.

Uveďme si opět ještě analytické řešení naší úlohy. Po dosazení zadaných hodnot dostaneme Cauchyovu úlohu

$$\left. \begin{aligned} y''(t) &= -10y(t), \\ y(0) = 0, \quad y'(0) &= 1. \end{aligned} \right\} \quad (6.17)$$



Obrázek 6.20: Odhad průběhu výchylek kyvadla

Pomocí derivování můžeme ověřit, že řešením úlohy (3.19) je funkce

$$\bar{y}(t) = \frac{\sqrt{10}}{10} \sin(\sqrt{10}t).$$

Graf této funkce odpovídá průběhu na obr. 6.20.

V následující tabulce si ještě ukážeme, jak závisí chyba aproximace výchylky kyvadla v čase $t = 10$ sekund získaná Eulerovou metodou na volbě n , a tuto chybu porovnáme s odhadem (3.5). Vycíslení odhadu (3.5) je náročnější, a proto v tabulce uvádíme jen hodnotu tohoto výrazu a podrobný výpočet přenecháme zvědavému čtenáři.

n	h	$ y_{approx}(10) - \bar{y}(10) $	$\frac{M}{2K} (e^{K(10-t_0)} - 1) h$
10^4	10^{-3}	$3,3 \cdot 10^{-3}$	11,6
10^5	10^{-4}	$3,3 \cdot 10^{-4}$	1,11
10^6	10^{-5}	$3,2 \cdot 10^{-5}$	$1,10 \cdot 10^{-1}$

6.4 Úlohy kapitoly 4

Řešení ke kapitole 4.1

Příklad 1 Uvedeme si nejprve kód řešení:

```
function u = struna1(c,k,l,n)
% u = struna1(c,k,l,n)
%
% Funkce vyresi metodou siti tuto Dirichletovu ulohu
%      -k*u''(x) = c na (0,l)
%      u(0) = u(l) = 0
%
% c ... konstantni hustota zatezujici sily
% k ... tuhost struny
% l ... delka struny
% n+1 ... pocet uzlu na strune (pravidelna sit)
% u ... vysledny pruhyb struny

% vzdalenost mezi sousednimi uzly
h = l / n;

% matice tuhosti K
K = diag(2*ones(n-1,1)) + diag(-ones(n-2,1),1) + diag(-ones(n-2,1),-1);
K = k/h^2 * K;

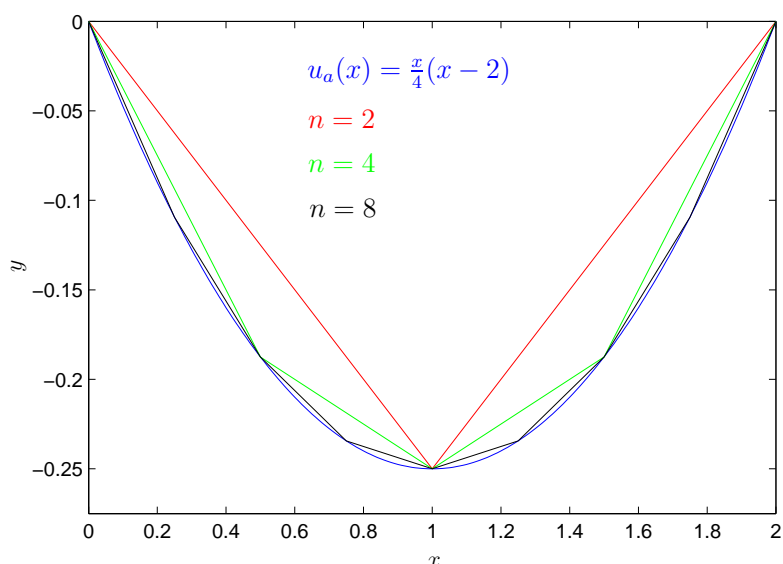
% vektor prave strany f
f = c * ones(n-1,1);

% reseni soustavy - pruhyb struny v uzlech
u = K \ f;
u = [0; u; 0];

% vykresleni reseni
figure
x = 0 : h : l;
plot(x,u)

% srovnani s analytickym resenim
u_a = c/2/k * x .* (l-x);
err_abs = max(abs(u - u_a')) % maximalni absolutni chyba
```

Na obr. 6.21 vidíme průhyb struny pro $c = -1$, $k = 2$, $\ell = 2$ a různá n . Můžeme si všimnout, že průhyby struny se v uzlových bodech vždy přesně



Obrázek 6.21: Analytické řešení u_a pro $c = -1$, $k = 2$ a $\ell = 2$ a vypočtené průhyby struny pro různá n

shodují s analytickým řešením.²⁹ Je to proto, že metoda sítí je přesná pro $f := c = konst.$ – nahrazení pomocí diferencí na str. 54 je totiž v tomto případě přesné.

Příklad 2 Nejprve si vyjádříme analytické řešení úlohy pro zatěžující sílu s hustotou $f(x) := (x - 1/3)^2 - 1$.^{30,31}

$$u_a(x) = -\frac{(x - \frac{1}{3})^4}{12k} + \frac{x^2}{2k} + ax + b,$$

$$a = \frac{1}{2k\ell} \left[\frac{(\ell - \frac{1}{3})^4}{6} - \ell^2 - \frac{1}{486} \right], \quad b = \frac{1}{972k}.$$

²⁹Po doběhnutí programu s jakýmkoli vstupním n dostáváme v proměnné `err_abs` vždy (počítačovou) nulu.

³⁰To dostaneme stejným způsobem, jaký jsme použili na str. 55 pro konstantní f .

³¹Všimněme si, že $f(x) < 0$ pro $x \in (0, 4/3)$ a $f(x) > 0$ pro $x \in (4/3, 2)$, tzn. že strunu zatlačujeme směrem dolů v intervalu $(0, 4/3)$ a zvedáme nahoru v intervalu $(4/3, 2)$.

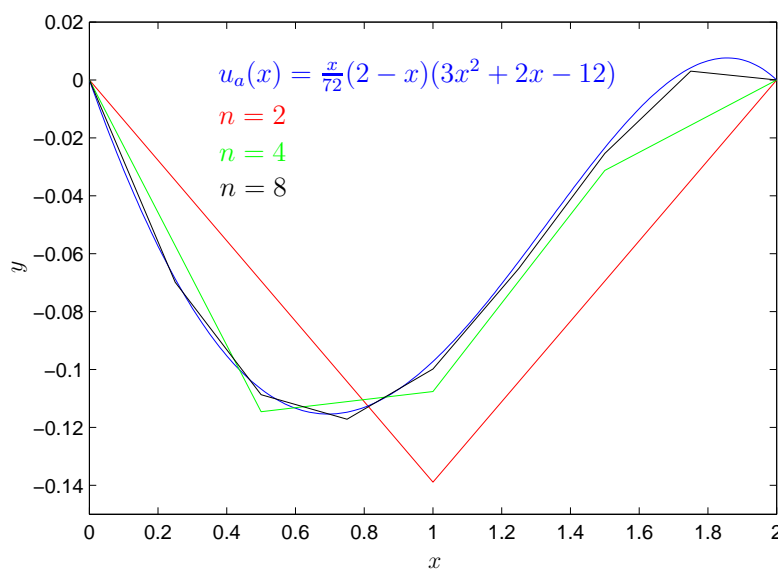
Kód použijeme stejný jako v příkladě 1, jen upravíme předpis pro vektor pravé strany:

```
% vektor prave strany f
x = h : h : l-h;
f = (x - 1/3).^2 - 1;
f = f';
```

a upravíme i část kódu pro srovnání s analytickým řešením:³²

```
% srovnani s analytickym resenim
a = ( (1-1/3)^4/6 - 1^2 - 1/486 ) / 2 / k / 1;
b = 1 / 972 / k;
u_a = -(x-1/3).^4/12/k + x.^2/2/k + a*x + b;
err_abs = max(abs(u - u_a')) % maximalni absolutni chyba
```

Na obr. 6.22 vidíme průhyb struny pro $k = 2$ a $\ell = 2$ a různá n . V následující



Obrázek 6.22: Analytické řešení u_a pro $k = 2$ a $\ell = 2$ a vypočtené průhyby struny pro různá n

tabulce pak uvádíme výsledky pro $k = 2$ a $\ell = 2$:

³²Takto upravenou funkci `struna1` můžeme přejmenovat na `struna2`, přičemž vstupními argumenty funkce `struna2` budou pouze proměnné k , l a n .

n	err_abs
2	$4,17 \cdot 10^{-2}$
4	$1,04 \cdot 10^{-2}$
8	$2,6 \cdot 10^{-3}$
100	$1,67 \cdot 10^{-5}$
300	$1,85 \cdot 10^{-6}$
500	$6,67 \cdot 10^{-7}$
1000	$1,67 \cdot 10^{-7}$

Příklad 3 K řešení použijeme následující kód:

```

l = 2;
d = 0.2;
n = 300;

k = 2; % vhodna inicializace

while 1
    k = k - 0.1;
    u = struna2(k,l,n);
    if min(u) <= -d
        break
    end
end

k = k + 0.1;

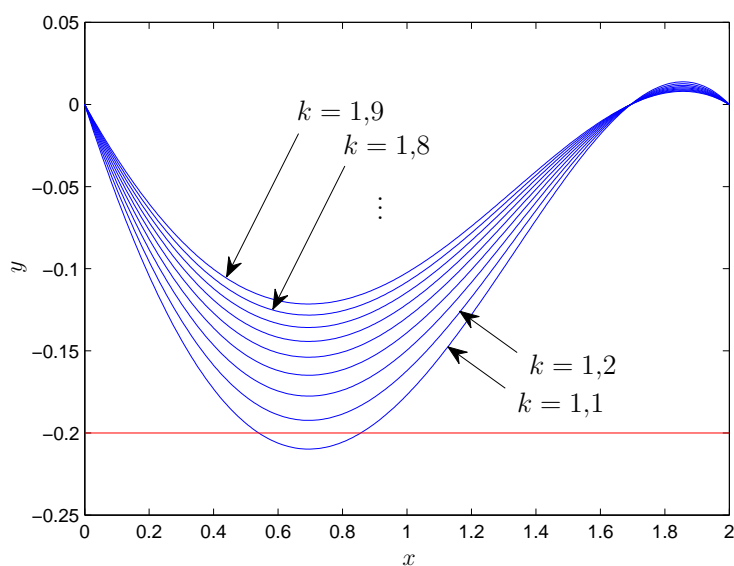
```

Funkce `struna2`, která se volá uvnitř cyklu, je kód řešení příkladu 2 (tedy upravený kód řešení příkladu 1). Inicializace tuhosti k se odvíjí od řešení příkladu 2, kde jsme mimo jiné zjistili, že pro $k = 2$ se struna nachází celá nad překážkou. Po doběhnutí programu zjistíme, že struna musí mít (s přesností 10^{-1}) tuhost nejméně $k = 1,2$, aby se nedotkla zadané překážky. Na obr. 6.23 jsou znázorněny průhyby stuny pro postupně klesající tuhost.

6.5 Úlohy kapitoly 5

Řešení ke kapitole 5.2

Příklad 1 Rovnovážnou polohu volně zavěšené kuličky dané hmotnosti najdeme jako řešení minimalizační úlohy (5.7). K tomu použijeme Newtonovu



Obrázek 6.23: Jednotlivé průhyby struny pro klesající tuhost

metodu pro minimalizaci funkce více proměnných. Hledanou rovnovážnou polohu najdeme pomocí následujícího kódu. Hmotnost kuličky je v proměnné m , za kterou postupně dosazujeme hodnoty 1, 2, 3, 4, 5 a 10.

```
function [y,it] = newton(df,ddf,x0,epsilon)

%[y,it] = newton(df,ddf,x0,epsilon)
%
% Zakladni Newtonova metoda pro minimalizaci fce f(x)
%
% y ... minimum
% it ... pocet iteraci
%
% df ... gradient minimalizovane fce f(x)
% ddf ... Hessian minimalizovane fce f(x)
% x0 ... pocatecni iterace
% epsilon ... ukoncuji podminka

global m

m = 1;
```

```

it = 0;
x = x0;
grad = feval(df,x);
Hess = feval(ddf,x);

while norm(grad) >= epsilon33
    it = it + 1;
    x = x - Hess \ grad;
    grad = feval(df,x);
    Hess = feval(ddf,x);
end

y = x;

```

Newtonova metoda potrebuje v každém kroce iteračního procesu vyčíslit gradient a Hessián. Gradient funkce $f(x_1, x_2) = x_1^2 + x_2^2 + mx_2$ vypočteme pomocí následujícího kódu.

```

function y = df(x)

global m

y = [2*x(1); 2*x(2) + m];

```

Hessián této funkce vypočteme pomocí kódu:

```

function y = ddf(x)

global m

y = [2 0; 0 2];

```

Pokud ukončující podmínku `epsilon` zvolíme jako 10^{-4} , získáme pro požadované hmotnosti rovnovážné polohy kuličky uvedené v tabulce 6.2.

Příklad 2 Pomocí stejného kódu jako v příkladě 1 získáme rovnovážnou polohu kuličky dané hmotnosti zavěšené nad překážkou. Protože polohu kuličky získáme jako řešení minimalizační úlohy s omezením (5.9), použijeme nyní metodu kvadratické penalty a na rozdíl od příkladu 1 budeme minimali-

³³Funkce `norm` vypočítá velikost vektoru `grad`.

m	souřadnice x_1	souřadnice x_2
1	0	-0,5
2	0	-1
3	0	-1,5
4	0	-2
5	0	-2,5
10	0	-5

Tabulka 6.2: Rovnovážné polohy volně zavěšené kuličky o hmotnosti m

zovat funkci $f(x_1, x_2) = x_1^2 + x_2^2 + mx_2 + \frac{1}{2}\rho\|\alpha(x)\|^2$, kde $\alpha_1(x_1, x_2) = \max\{-x_1 - x_2 - 1, 0\}$ a $\alpha_2(x_1, x_2) = \max\{x_1 - x_2 - 3, 0\}$. Penalizační parametr ρ zvolíme jako 10^5 .

Pro Newtonovu metodu musíme opět v každém kroce iteračního procesu vyčíslit gradient a Hessián. Gradient funkce $f(x_1, x_2) = x_1^2 + x_2^2 + mx_2 + \frac{1}{2}\rho\|\alpha(x)\|^2$ vypočteme pomocí kódu:

```
function y = df(x)

global m
rho = 1e5;

f = x(1)^2 + x(2)^2 + m*x(2);
h(1,1) = -x(1) - x(2) - 1;
h(2,1) = x(1) - x(2) - 3;

alpha = max(zeros(2,1), h);

gf = [2*x(1); 2*x(2) + m];

if alpha(1,1) == 0 && alpha(2,1) == 034
    ga = zeros(2,1);
elseif alpha(1,1) > 0 && alpha(2,1) == 0
    ga = alpha(1,1)*[-1;-1];
elseif alpha(1,1) == 0 && alpha(2,1) > 0
```

³⁴V této části kódu vypočteme gradient funkce $\frac{1}{2}\|\alpha(x)\|^2$ a zapíšeme jej do proměnné ga . Pokud si čtenář chce náš výpočet ověřit, doporučujeme, aby si přepsal tuto funkci bez použití funkce `max`. Tzn. je rozumné napsat si funkční předpis této funkce pro tyto případy: $-x_1 - x_2 - 1 \leq 0 \wedge x_1 - x_2 - 3 \leq 0$, $-x_1 - x_2 - 1 > 0 \wedge x_1 - x_2 - 3 \leq 0$, $-x_1 - x_2 - 1 \leq 0 \wedge x_1 - x_2 - 3 > 0$ a $-x_1 - x_2 - 1 > 0 \wedge x_1 - x_2 - 3 > 0$ a pro tyto případy pak spočítat první parciální derivace této funkce.

```

        ga = alpha(2,1)*[1;-1];
    else
        ga = alpha(1,1)*[-1;-1] + alpha(2,1)*[1;-1];
    end;

y = gf + rho*ga;

```

Hessián této funkce vypočteme pomocí kódu:

```

function y = ddf(x)

global m
rho = 1e5;

f = x(1)^2 + x(2)^2 + m*x(2);
h(1,1) = -x(1) - x(2) - 1;
h(2,1) = x(1) - x(2) - 3;

alpha = max(zeros(2,1), h);

Hf = [2 0; 0 2];

if alpha(1,1) == 0 && alpha(2,1) == 035

    Ha = zeros(2,2);
elseif alpha(1,1) > 0 && alpha(2,1) == 0
    Ha = [1 1; 1 1];
elseif alpha(1,1) == 0 && alpha(2,1) > 0
    Ha = [1 -1; -1 1];
else
    Ha = [2 0; 0 2];
end;

y = Hf + rho*Ha;

```

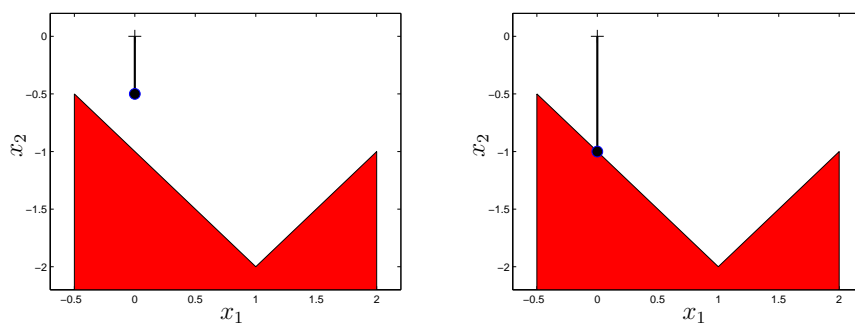
³⁵V této části kódu vypočteme Hessián funkce $\frac{1}{2}\|\alpha(x)\|^2$ a zapíšeme jej do proměnné Ha. Pokud si čtenář chce náš výpočet ověřit, doporučujeme, aby si podobně jako v předchozím kódu přepsal tuto funkci bez použití funkce max. Tzn. opět je rozumné napsat si funkční předpis této funkce pro čtyři případy uvedené v předchozí rutině a pro tyto případy pak spočítat druhé parciální derivace této funkce.

Pokud ukončující podmínku ϵ zvolíme jako 10^{-4} , získáme pro požadované hmotnosti rovnovážné polohy kuličky uvedené v tabulce 6.3.

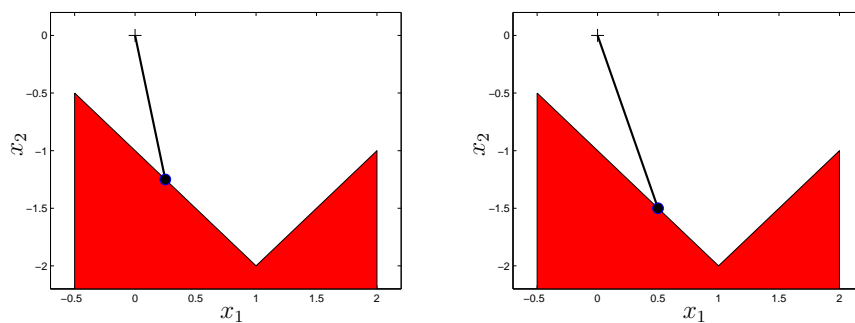
m	souřadnice x_1	souřadnice x_2
1	0	-0,5
2	0	-1
3	0,25	-1,25
4	0,5	-1,5
5	0,75	-1,75
10	1	-2

Tabulka 6.3: Rovnovážné polohy kuličky o hmotnosti m zavěšené nad překážkou

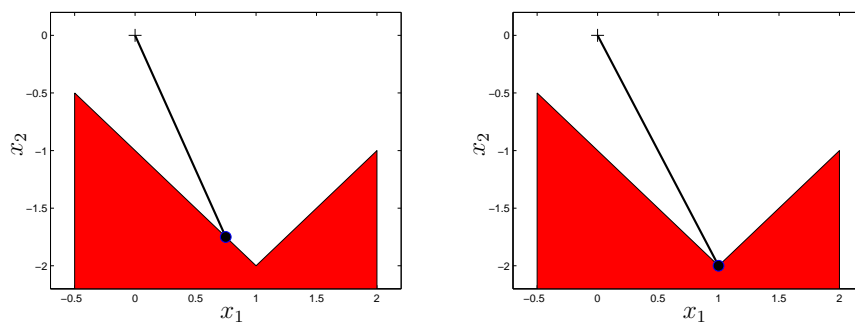
Na následujících obrázcích jsou zakresleny rovnovážné polohy kuličky zavěšené nad překážkou pro všechny uvažované hmotnosti.



Obrázek 6.24: Rovnovážné polohy kuličky o hmotnosti $m=1$ (vlevo) a $m=2$ (vpravo) zavěšené nad překážkou



Obrázek 6.25: Rovnovážné polohy kuličky o hmotnosti $m=3$ (vlevo) a $m=4$ (vpravo) zavěšené nad překážkou



Obrázek 6.26: Rovnovážné polohy kuličky o hmotnosti $m=5$ (vlevo) a $m=10$ (vpravo) zavěšené nad překážkou

Apendix

Apendix A: Inverzní funkce

Definice A.1 Bud' $f : \mathbb{R} \mapsto \mathbb{R}$. Funkci f^{-1} , pro niž platí současně:

- i) definiční obor Df^{-1} funkce f^{-1} je roven oboru hodnot funkce f ,
- ii) pro každé $x \in Df^{-1}$ platí, že $f^{-1}(x) = y \Leftrightarrow f(y) = x$,

nazveme funkcí inverzní k funkci f .

Poznámka A.1 Lze ukázat, že f^{-1} existuje právě tehdy, je-li f prostá. Graf f^{-1} je přitom osově souměrný s grafem f dle přímky $y = x$.

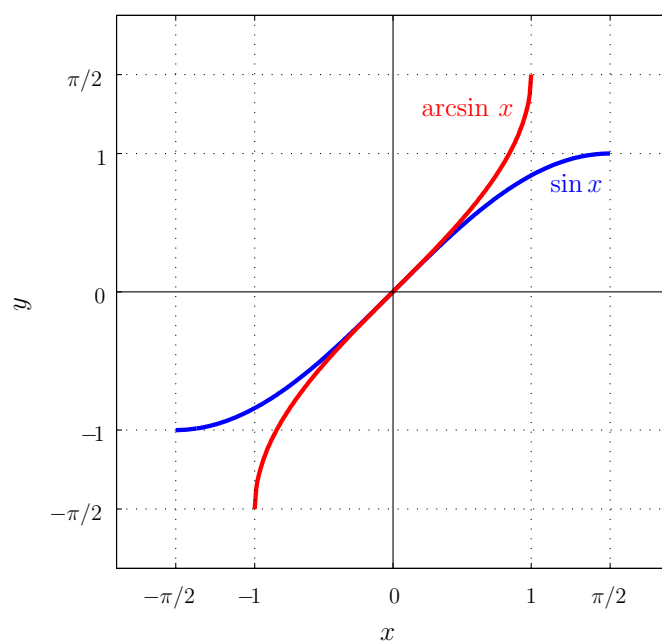
Apendix B: Funkce arkussinus

Definice B.1 Funkci inverzní k funkci sinus zúžené na interval $\langle -\frac{\pi}{2}, \frac{\pi}{2} \rangle$ nazveme arkussinus a označujeme jako arcsin, tj.

$$\arcsin := \left(\sin|_{\langle -\frac{\pi}{2}, \frac{\pi}{2} \rangle} \right)^{-1}.$$

Pozorování B.1 Funkce arkussinus má tyto vlastnosti (viz také níže uvedený obrázek):

- definiční obor je roven $\langle -1, 1 \rangle$,
- oborem hodnot je interval $\langle -\frac{\pi}{2}, \frac{\pi}{2} \rangle$,
- funkce arcsin je lichá, tj. $\arcsin x = \arcsin(-x)$ pro každé $x \in \langle -1, 1 \rangle$.



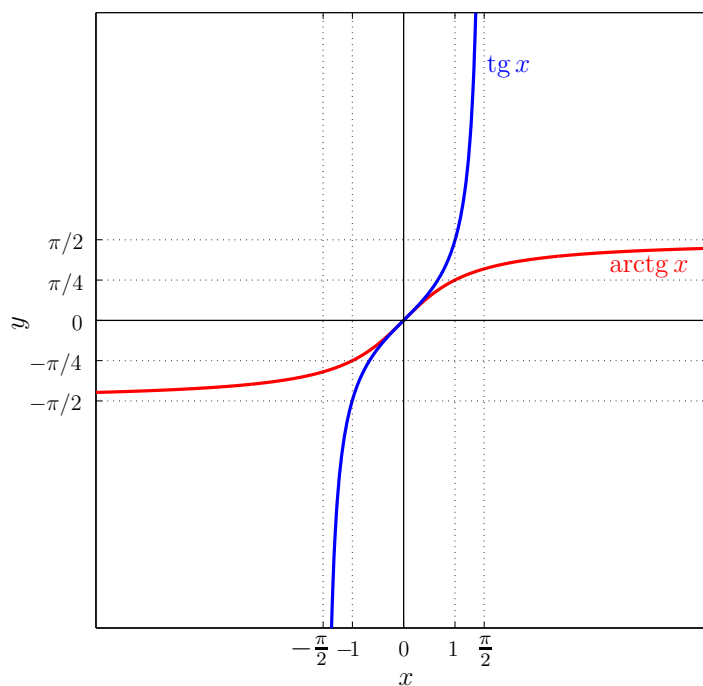
Apendix C: Funkce arkustangens

Definice C.1 Funkci inverzní k funkci tangens zúžené na interval $(-\frac{\pi}{2}, \frac{\pi}{2})$ nazveme arkustangens a označujeme jako arctg , tj.

$$\operatorname{arctg} := \left(\operatorname{tg}|_{(-\frac{\pi}{2}, \frac{\pi}{2})} \right)^{-1}.$$

Pozorování C.1 Funkce arkustangens má tyto vlastnosti:

- definiční obor je roven \mathbb{R} ,
- oborem hodnot je interval $(-\frac{\pi}{2}, \frac{\pi}{2})$,
- funkce arctg je lichá, tj. $\operatorname{arctg} x = \operatorname{arctg}(-x)$ pro každé $x \in \mathbb{R}$.



Apendix D: Soustavy lineárních rovnic a jejich řešení

Definice D.1 Soustavou n lineárních rovnic o n neznámých x_1, \dots, x_n nazýváme systém rovnic ve tvaru:

$$\left. \begin{array}{cccc} a_{11}x_1 & + & \cdots & + & a_{1n}x_n & = & b_1 \\ \vdots & & \ddots & & \vdots & & \vdots \\ a_{n1}x_1 & + & \cdots & + & a_{nn}x_n & = & b_n \end{array} \right\} \quad (7.1)$$

Čísla a_{ij} , $i = 1, \dots, n$, $j = 1, \dots, n$, nazýváme koeficienty soustavy a čísla b_i , $i = 1, \dots, n$, nazýváme pravé strany.

Soustavu (7.1) budeme úsporně zapisovat pomocí matice A a vektoru b , kde

$$A = \begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{n1} & \cdots & a_{nn} \end{bmatrix}, \quad b = \begin{bmatrix} b_1 \\ \vdots \\ b_n \end{bmatrix}.$$

Matici A nazýváme maticí soustavy (7.1) a vektor b nazýváme pravou stranou soustavy (7.1). Má-li vektor x za složky neznámé x_1, \dots, x_n , můžeme soustavu (7.1) zapsat v maticové podobě

$$Ax = b.$$

Řešení x soustavy (7.1) můžeme najít pomocí inverzní matice k matici A (pojem inverzní matice jsme definovali v kapitole 2.1). Řešení x můžeme nalézt tak, že soustavu $Ax = b$ vynásobíme zleva maticí A^{-1} a po úpravě získáme

$$x = A^{-1}b.$$

Protože výpočet inverzní matice je numericky značně náročný, používají se pro řešení velkých soustav jiné metody. Běžně používaná je Gaussova eliminační metoda, která spočívá v nahrazení původní soustavy soustavou, která má stejné řešení jako soustava původní, ale toto řešení z ní snadno vyčteme. Pro nalezení nové jednodušší soustavy používáme tzv. ekvivalentní úpravy.

Gaussovu eliminační metodu má řada matematických softwarů již implementováno, např. v programu Matlab můžeme řešení soustavy $Ax = b$ zapsat příkazem $\mathbf{x} = \mathbf{A} \setminus \mathbf{b}$.

Literatura

- [1] J. Bouchala: *Matematická analýza 1*. Skripta Vysoké školy báňské - Technické univerzity Ostrava (2005). <http://homel.vsb.cz/~bou10/MA1/ma1.html>
- [2] M. Braun: *Differential Equations and Their Applications*. Springer Verlag (1993).
- [3] M. Barnsley: *Fractals Everywhere*. Academic Press, Boston (1993).
- [4] Z. Dostál, P. Beremlijski: *Metody optimalizace*. Text vytvořený při realizaci projektu „Matematika pro inženýry 21. století“, Vysoká škola báňská - Technická univerzita Ostrava (2012). <http://mi21.vsb.cz/modul/metody-optimalizace>
- [5] Z. Dostál, V. Vondrák: *Lineární algebra*. Text vytvořený při realizaci projektu „Matematika pro inženýry 21. století“, Vysoká škola báňská - Technická univerzita Ostrava (2011). <http://mi21.vsb.cz/modul/linearni-algebra>
- [6] B.G. Katzung: *Základní a klinická farmakologie*. H & H (1994).
- [7] T. Kozubek, T. Brzobohatý, V. Hapla, M. Jarošová, A. Markopoulos: *Lineární algebra s Matlabem*. Text vytvořený při realizaci projektu „Matematika pro inženýry 21. století“, Vysoká škola báňská - Technická univerzita Ostrava (2012). <http://mi21.vsb.cz/modul/linearni-algebra-s-matlabem>
- [8] B. Krajc, P. Beremlijski: *Obyčejné diferenciální rovnice*. Text vytvořený při realizaci projektu „Matematika pro inženýry 21. století“, Vysoká škola báňská - Technická univerzita Ostrava (2012). <http://mi21.vsb.cz/modul/obycejne-diferencialni-rovnice>
- [9] M. Tennenbaum, H. Pollard: *Ordinary Differential Equations: An Elementary Textbook for Students of Mathematics, Engineering, and the Sciences*. Dover Publications (1985).

- [10] P. Vodstrčil, J. Bouchala: *Drobná překvapení spojená s numerickou integrací*. Pokroky matematiky, fyziky a astronomie, svazek 55, číslo 4 (2010). <http://dml.cz/dmlcz/141970>