Katedra
aplikované matematiky

Počítačová cvičení
Škola matematického modelování
2015

Petr Beremlijski, Rajko Ćosić, Lukáš Malý, Marie Sadowská,
Robert Skopal

evropský sociální fond v ČR
EVROPSKÁ UNIE
MINISTERSTVO ŠKOLSTVÍ, MLÁDEŽE A TĚLOVÝCHOVY
OP Vzdělávání pro konkurenceschopnost
INVESTICE DO ROZVOJE VZDĚLÁVÁNÍ

Počítačová cvičení
Škola matematického modelování

Petr Beremlijski, Rajko Ćosić, Lukáš Malý, Marie Sadowská,
Robert Skopal

Katedra
aplikované matematiky

Katedra aplikované matematiky
Fakulta elektrotechniky a informatiky
VŠB - Technická univerzita Ostrava
2015

Tato akce byla podpořena z prostředků projektu Matematika s radostí (registrační číslo CZ.1.07/1.1.00/26.0042). Jeho webová stránka je http://msr.vsb.cz.

evropský sociální fond v ČR
EVROPSKÁ UNIE
MINISTERSTVO ŠKOLSTVÍ, MLÁDEŽE A TĚLOVÝCHOVY
OP Vzdělávání pro konkurenceschopnost

INVESTICE DO ROZVOJE VZDĚLÁVÁNÍ

Cvičení 1: Matlab – nástroj pro matematické modelování

Abychom se mohli věnovat numerickému řešení matematických úloh, potřebujeme vhodné prostředí, které nám to umožní. A tak jako fyzici či chemikové mají své laboratoře, mají i numeričtí matematici svou Maticovou laboratoř[1] - Matlab. Podrobně se tomuto pracovnímu prostředí a jeho příkazům věnuje přiložený Matlabovský slabikář [4] nebo také úvod textu [3]. My si v tomto textu uvedeme pouze stručný přehled matlabovských proměnných a příkazů, které budeme potřebovat.

Prostředí

- *help, demos, intro, who, whos, clear, size, length*

Proměnné

- Skaláry
- Vektory
- Matice

Příkazy

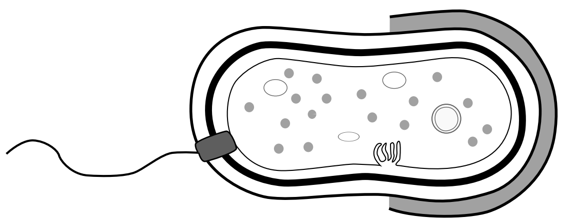
- Skalární funkce - *sin, cos, tan, cot, exp, log, abs, sqrt, round*
- Vektorové funkce a generování vektorů - *max, min, sort*
- Maticové funkce a generování matic - *det, rand, ones, zeros, eye*
- Skalární operace - $+,\ -,\ *,\ /,\hat{}$
- Maticové a vektorové operace - $+,\ -,\ *,\ '$(transponování), \ ($A\backslash v = x \Leftrightarrow Ax = v$)
  Operace „po prvcích“ - $.*,\ .\hat{},\ ./$
- 2D grafika (vykreslení grafů funkcí jedné proměnné) - *plot, hold on, hold off, figure*
- 3D grafika (vykreslení grafů funkcí dvou proměnných) - *meshgrid, mesh, contour, hold on, hold off, figure*
- Řídící příkazy - *if* (podmíněný příkaz), *for, while* (příkazy cyklu se známým počtem opakování a podmínkou na začátku)
- Relace a logické operace - $<,\ >,\ <=,\ >=,\ ==,\ \sim=,\ \&,\ |,\ \sim$
- Skripty a funkce - *function*

[1] MATrix LABoratory

Vše si nyní vyzkoušíme při řešení následujících úloh.

Úkol 1.1 Legenda říká, že když byly vymyšleny šachy, tak se místnímu vládci (někde v Asii) tato hra tak zalíbila, že se rozhodl odměnit jejich vynálezce a za odměnu mu nabídl cokoliv, co si bude přát. Vynálezce mu na to odpověděl, že si nepřeje nic jiného než několik zrnek rýže. A aby se to dobře počítalo, tak že chce za první políčko šachovnice dostat jedno zrnko rýže, za druhé dvě zrnka rýže, za třetí čtyři zrnka, za čtvrté osm zrnek a tak dále. Tedy ať za každé další pole šachovnice dostane dvojnásobný počet zrnek rýže ve srovnání s polem předchozím. Kolik kilogramů rýže žádal, jestliže 30 000 zrnek rýže váží 1 kilogram? ▲

Úkol 1.2 Bakterie Yersinia pestis, která způsobuje onemocnění morem, se v příznivých podmínkách dělí jednou za 100 minut. Jak dlouho by trvalo, pokud by nedocházelo k úhynu bakterií a mohly se bez omezení množit, než by jejich hmotnost překročila hmotnost Země? Předpokládejme, že v čase $t = 0$ žije jedna bakterie Yersinia pestis. Předpokládejme, že hmotnost jedné bakterie je $6 \cdot 10^{-15}$ kg a hmotnost Země je $6 \cdot 10^{24}$ kg. ▲



Obrázek 1: Bakterie

Úkol 1.3 Sestrojte grafy následujících funkcí:

- $f(x) := x^2$
- $f(x) := \sqrt{1 - x^2}$
- $f(x) := x^2 \sin\left(\frac{1}{x^2}\right)$
- $f(x) := |x|$ ▲

Úkol 1.4 Sestrojte grafy následujících funkcí:

- $f(x, y) := x^2 + y^2$
- $f(x, y) := \sqrt{x^2 + y^2}$
- $f(x, y) := (x^2 + y^2) \sin\left(\frac{1}{x^2+y^2}\right)$
- $f(x, y) := \sqrt{|xy|}$ ▲

Cvičení 2: Polynomiální regrese

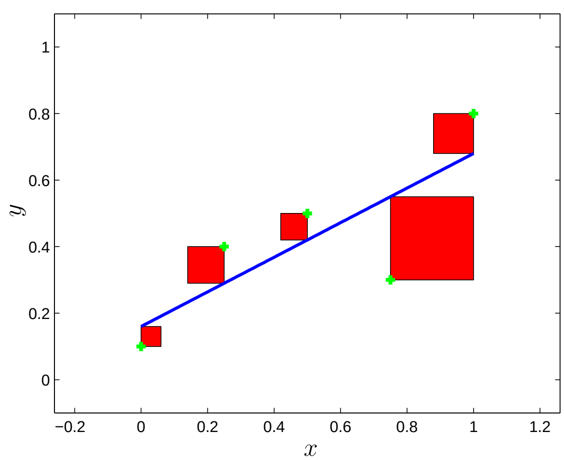
Při zkoumání různých fyzikálních jevů se setkáváme s následujícím úkolem. Na intervalu $\langle a, b\rangle \subset \mathbb{R}$ máme změřeny hodnoty zkoumané veličiny a hledáme polynom n-tého řádu $p_n(x)$, který „nejlépe“ přibližně popisuje (aproximuje) naše měření. To, že polynom $p_n(x)$ „nejlépe“ aproximuje zadané hodnoty, pro nás bude znamenat, že $p_n(x)$ je řešením úlohy

$$\min_{p_n(x)} \sum_{i=1}^{k} \left(p_n(x_i) - y_i\right)^2 , \tag{1}$$

kde $[x_i, y_i]$, $i \in \{1, \ldots, k\}$ jsou souřadnice naměřených hodnot a

$$p_n(x) = a_n x^n + a_{n-1}x^{n-1} + \cdots + a_2x^2 + a_1x + a_0,$$

kde $a_j \in \mathbb{R}$ pro každé $j \in \{0, \ldots, n\}$.[2] Všimněme si, že $p_n(x)$ řešící (1) je polynom, pro který je součet čtverců rozdílů $p_n(x_i)$ a y_i nejmenší. Proto se této metodě hledání koeficientů a_0, $a_1, \ldots, a_n$ pro určení polynomu $p_n(x)$ říká `metoda nejmenších čtverců`. Aproximaci zadaných bodů polynomem nazýváme `polynomiální regrese`. Pro lepší představu se podívejte na obr. 2, kde je znázorněna regrese zadaných bodů lineárním polynomem. Zadané body jsou vyznačeny zelenými křížky, regresní funkce je zakreslena modře a čtverce rozdílů $p_n(x_i)$ a y_i jsou zakresleny červeně.



Obrázek 2: Regrese lineárním polynomem použitím metody nejmenších čtverců

Řešení úlohy (1) je poměrně obtížné, proto k jejímu řešení použijeme programové prostředí Matlab a v něm příkaz *polyfit*.

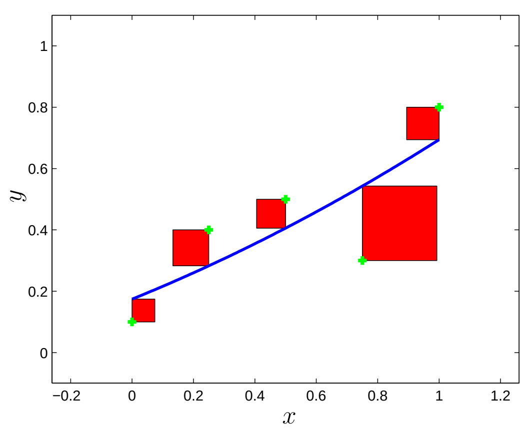
[2]Hledáním polynomu $p_n(x)$, který řeší úlohu (1), myslíme hledání hodnot koeficientů a_0, a_1, $\cdots$, a_n tohoto polynomu.

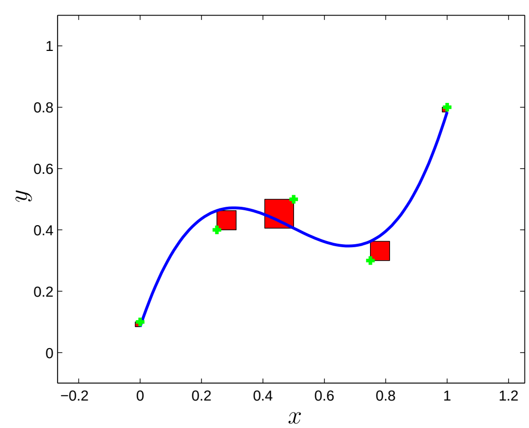
Syntaxe tohoto příkazu je následující: P = *polyfit(X,Y,N)*, kde X a Y jsou vektory obsahující první a druhé souřadnice aproximovaných hodnot a N je řád polynomu, kterým aproximujeme. Ve vektoru P získáme koeficenty polynomu $p_n(x)$, pro které platí

$$P(1) = a_n,\ P(2) = a_{n-1},\ \ldots,\ P(N-1) = a_2,\ P(N) = a_1 \text{ a } P(N+1) = a_0.$$

Pro vyčíslení hodnoty polynomu v zadaném bodě či vektoru bodů x, můžeme použít příkaz Matlabu *polyval*. Jeho syntaxe je Y = *polyval(P,x)*, kde P je vektor obsahující koeficienty daného polynomu (opět platí $P(1) = a_n$, $P(2) = a_{n-1}$, ..., $P(N-1) = a_2$, $P(N) = a_1$ a $P(N+1) = a_0$) a x obsahuje x-ové souřadnice bodu či bodů, ve kterých chceme polynom vyčíslit. Funkční hodnoty daného polynomu jsou uloženy do vektoru Y.

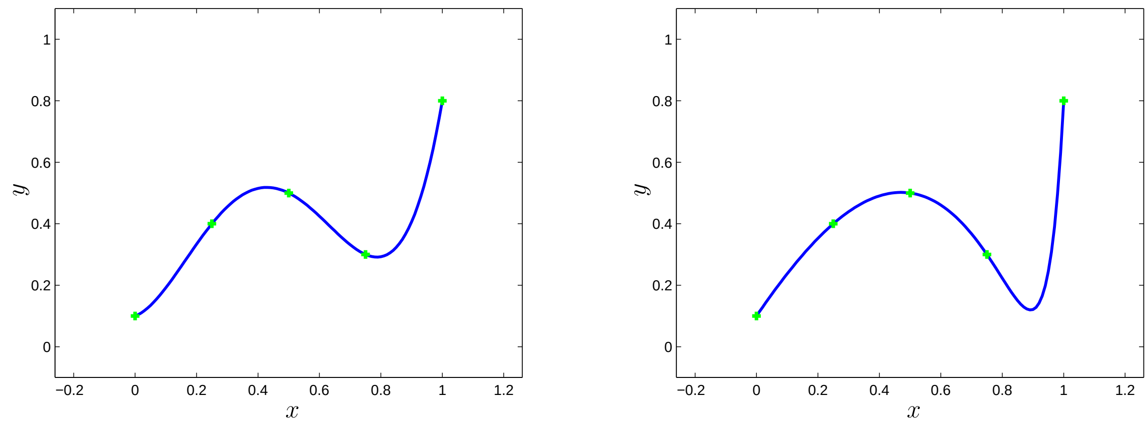
Na následujících obrázcích si můžete prohlédnout regresi zadaných bodů $[x_i, y_i]$, kde $i \in \{1, \ldots, 5\}$, pomocí polynomů vyšších řádů. Body jsou opět vyznačeny zelenými křížky, výsledné regresní funkce jsou zakresleny modře a čtverce rozdílů $p_n(x_i)$ a y_i jsou zakresleny červeně. Na regresi kvadratickým a kubickým polynomem se podívejte na obr. 3, regrese polynomem čtvrtého a dvacátého řádu je na obr. 4.





Obrázek 3: Regrese kvadratickým (vlevo) a kubickým (vpravo) polynomem použitím metody nejmenších čtverců

Z uvedených příkladů se dá usuzovat, že polynomy vyšších řádů aproximují zadané body lépe. Takový závěr ale už neplatí v celém intervalu, kde regresi používáme.



Obrázek 4: Regrese polynomem čtvrtého (vlevo) a dvacátého (vpravo) řádu použitím metody nejmenších čtverců

Úkol 2.1 Vygenerujte si pomocí příkazu *rand* náhodných 11 hodnot v intervalu $\langle 0, 1\rangle$. Vektor takto získaných hodnot považujte za y-ové souřadnice bodů, jejichž x-ové souřadnice jsou 0, 0,1, 0,2, 0,3, 0,4, 0,5, 0,6, 0,7, 0,8, 0,9, 1. Tyto body postupně aproximujte polynomy 1., 2., . . . , 15. řádu. ▲

Úkol 2.2 Získejte hodnoty součinitele aerodynamického odporu vzduchu C_d pro hodnoty 0,7, 0,8, 0,9, 1, 1,1, 1,2, 1,3, 1,4, kterých nabývá první parametr popisující karoserii Tatry 87. K získání těchto hodnot použijte přiložený program *tatra.exe*. Poté aproximujte tyto hodnoty postupně polynomy 1., 2., . . . , n. řádu.[3] Tím získáme polynomy aproximující závislost na prvním parametru karoserie Tatry 87. ▲

[3] Číslo n zvolte tak, aby polynom n-tého řádu „dobře“ aproximoval zadané body.

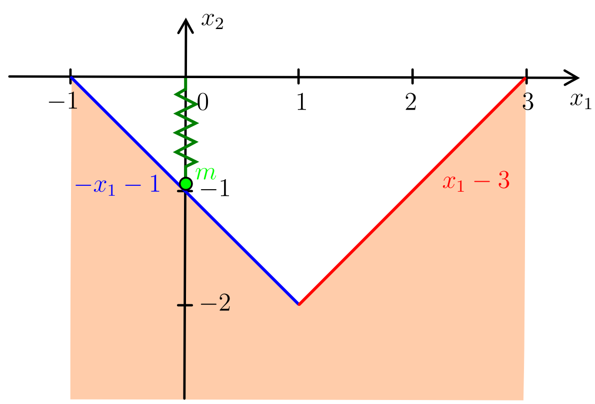
V tomto cvičení se seznámíme s tzv. optimalizačními úlohami, ukážeme si jejich praktickou aplikaci a také, jak je lze numericky řešit. Mnohem více mohou čtenáři nalézt v textu [2].

Optimalizační úlohy vznikají často při řešení praktických úloh. Matematicky jsou tyto úlohy obvykle formulovány jako problémy hledání extrémů `cenové funkce`, což je funkce $f : \mathbb{R}^n \to \mathbb{R}$. Extrémy cenové funkce často hledáme na `přípustné množině` $\Omega \subset D_f$. 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 $\overline{x} \in \Omega$ tak, aby platilo

$$f(\overline{x}) \leq f(x), \qquad x \in \Omega.^4 \tag{2}$$

Pokud je $\Omega = D_f$, mluvíme o `optimalizaci bez omezení`. Pokud pro Ω platí $\Omega \subset D_f$ a $\Omega \neq D_f$, 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 $\overline{x} \in \Omega$ tak, aby platila nerovnost (2), 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).



Obrázek 5: Příklad rovnovážného stavu kuličky na pružině (úloha s omezením)

Odkazy na obecně formulované optimalizační úlohy najdeme již ve starověku. Patří mezi ně i Dídonina[5] úloha. Dídó údajně dostala po přistání na africkém pobřeží od místního

[4] Tuto úlohu můžeme zadat také takto: „Řešte úlohu $\min_{x\in\Omega} f(x)$."

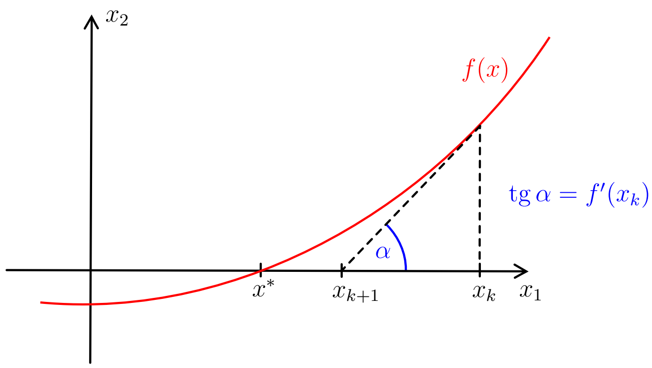
[5] Dle legendy byla Dídó fénická princezna, sestra krále Pygmalióna a zakladatelka starověkého Kartága.

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 (2) 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 (2), odvodíme si iterační metodu pro hledání řešení rovnice. Tato metoda se také nazývá Newtonova. 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. 6 následující vztah



Obrázek 6: Newtonova metoda

$$f'(x_k) = \frac{f(x_k)}{x_k - x_{k+1}}. \tag{3}$$

Rovnici (3) snadno upravíme do tvaru

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}. \tag{4}$$

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 $\lvert x_{k+1} - x_k\rvert \geq \varepsilon \quad (\lvert f(x_{k+1})\rvert \geq \varepsilon)$ $k = k + 1$ $x_{k+1} = x_k - f(x_k)/f'(x_k)$ end 3. x_{k+1} aproximuje řešení rovnice $f(x) = 0$

Nyní využijeme předpisu (4) 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 $f'(x) = 0$,[6] budeme numericky hledat řešení rovnice $f'(x) = 0$. Použitím předpisu (4) obdržíme

$$x_{k+1} = x_k - \frac{f'(x_k)}{f''(x_k)}. \tag{5}$$

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 $\lvert x_{k+1} - x_k\rvert \geq \varepsilon \quad (\lvert f'(x_{k+1})\rvert \geq \varepsilon)$ $k = k + 1$ $x_{k+1} = x_k - f'(x_k)/f''(x_k)$ end 3. x_{k+1} aproximuje minimum funkce $f(x)$

Zobecněním předpisu (5) pro funkce více proměnných získáme metodu pro minimalizaci funkce více proměnných. Více se o optimalizačních metodách tohoto typu dozvíte v textu [1].

[6] Této podmínce říkáme nutná podmínka existence lokálního extrému.

Úkol 3.1 Pomocí Newtonovy metody najděte minimum funkce $f(x) := x^4 + 8x^3 + 24x^2 + 32x + 16$. ▲

Úkol 3.2 Pomocí Newtonovy metody najděte minimum polynomu aproximujícího funkci popisující závislost součinitele aerodynamického odporu vzduchu C_d na prvním parametru karoserie Tatry 87 a tím najděte hodnotu tohoto parametru, pro který je C_d minimální. ▲

Reference

[1] P. Beremlijski, M. Sadowská, M. Theuer: *Počítačová cvičení - Škola matematického modelování 2013*. Text vytvořený pro seminář „Škola matematického modelování 2013“, Vysoká škola báňská - Technická univerzita Ostrava (2013). (http://skomam.vsb.cz/archiv/2013/files/cviceni/cviceni_2013.pdf)

[2] 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)

[3] 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)

[4] K. Sigmon: *Matlab Primer*. University of Florida (1993).

Definice A.1 Buď $f: \mathbb{R} \to \mathbb{R}$ a $x \in \mathbb{R}$. Existuje-li

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

značíme ji $f'(x)$ a nazýváme `derivací funkce` f `v bodě` x.

Poznámka A.1 Většinou – a nejinak je to v tomto textu – se pod pojmem derivace rozumí konečná (tzv. `vlastní`) derivace.

Definice A.2 `Derivací funkce` f budeme dále rozumět funkci f' definovanou předpisem

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

Věta A.1 Buď $f, g: \mathbb{R} \to \mathbb{R}$ 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 $g'(x)$,
- $\left(\dfrac{f}{g}\right)'(x) = \dfrac{f'(x)g(x) - f(x)g'(x)}{g^2(x)}$, existují-li (vlastní) derivace $f'(x)$ a $g'(x)$ a je-li $g(x) \neq 0$.

Pozorování A.1

- $(c)' = 0,\ c \in \mathbb{R}$ (konst.), $x \in \mathbb{R}$,
- $(x^r)' = rx^{r-1},\ r \in \mathbb{R},\ x \in (0, +\infty)$,[7]
- $(\sin x)' = \cos x,\ x \in \mathbb{R}$,
- $(\cos x)' = -\sin x,\ x \in \mathbb{R}$,
- $(\operatorname{tg} x)' = \dfrac{1}{\cos^2 x},\ x \in \mathbb{R} \setminus \left\{\dfrac{\pi}{2} + k\pi :\ k \in \mathbb{Z}\right\}$,
- $(\operatorname{cotg} x)' = -\dfrac{1}{\sin^2 x},\ x \in \mathbb{R} \setminus \{k\pi :\ k \in \mathbb{Z}\}$,
- $(\mathrm{e}^x)' = \mathrm{e}^x,\ x \in \mathbb{R}$,
- $(\ln x)' = \dfrac{1}{x},\ x \in (0, +\infty)$.

[7] Připomeňme si, že $x^0 := 1$.

Definice A.3 Zadefinujme 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 <u>`druhou derivaci funkce`</u> f <u>`v bodě`</u> x_0 jako

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

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

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

přičemž $f^{(0)} := f$.