

POČÍTAČOVÁ CVIČENÍ
ŠKOLA MATEMATICKÉHO MODELOVÁNÍ
2017

Petr Beremlijski, Rajko Čosić, Marie Sadowská

POČÍTAČOVÁ CVIČENÍ

ŠKOLA MATEMATICKÉHO MODELOVÁNÍ

Petr Beremlijski, Rajko Čosić, Marie Sadowská



Katedra
aplikované matematiky

Katedra aplikované matematiky
Fakulta elektrotechniky a informatiky
VŠB - Technická univerzita Ostrava
2017

Úvod

Tento text je určen pro účastníky semináře Škola matematického modelování (<http://skomam.vsb.cz>) a slouží jako pomůcka k úlohám, které řeší studenti v průběhu tohoto semináře. Tento seminář, pro který používáme zkratku ŠKOMAM, organizuje Katedra aplikované matematiky (<http://am.vsb.cz>) Fakulty elektrotechniky a informatiky Vysoké školy báňské – Technické univerzity Ostrava jednou ročně již od roku 2005. Tento rok probíhá již 13. ročník tohoto semináře.

Pro počítačové řešení úloh, kterými se budeme zabývat, jsme zvolili komerční systém Matlab. 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 [2]. 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>.

Tento seminář je pořádán s finanční podporou Fakulty elektrotechniky a informatiky (<http://www.fei.vsb.cz>), statutárního města Ostravy a projektu Math Exercises for You (<http://math4u.vsb.cz>) podpořeného programem Erasmus+. Nad akcí převzala záštitu Jednota českých matematiků a fyziků (<http://jcmf.vsb.cz>).



OSTRAVA!!!



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 laboratoř¹ - Matlab. Podrobně se tomuto pracovnímu prostředí a jeho příkazům věnuje přiložený Matlabovský slabikář [1] nebo také úvod textu [2]. 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 - *+, -, *, /, ^*
- Maticové a vektorové operace - *+, -, *, ' (transponování), \ (A \setminus v = x \Leftrightarrow Ax = v)*
Operace „po prvcích“ - *.*, .^, ./*
- 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 - *<, >, <=, >=, ==, ~=, &, |, ~*
- Skripty a funkce - *function*

¹MATrix LABoratory

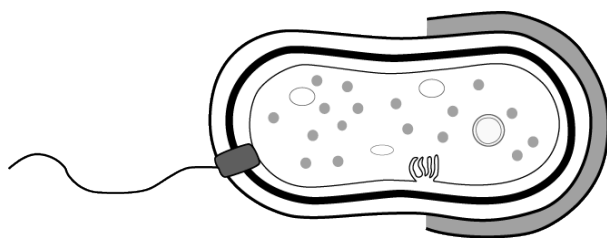
Vše si nyní vyzkoušíme při řešení následujících úloh.

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 zrněk 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 zrněk a tak dále. Tedy ať za každé další pole šachovnice dostane dvojnásobný počet zrněk rýže ve srovnání s polem předchozím. Kolik kilogramů rýže žádal, jestliže 30 000 zrněk 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:

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

▲

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. Pro odhad přesnosti metody Monte Carlo platí 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.

My si ukážeme tři způsoby, jak pomocí Monte Carla spočítat určitý integrál. První dvě metody jsou si typově podobné a spočívají v určení poměru obsahu neznámé plochy a plochy, jejíž obsah známe.

Náhodné souřadnice

Nejjednodušší způsob, jak pomocí Monte Carla odhadnout určitý integrál, je pomocí náhodného generování souřadnic v rovině. Princip si můžeme ukázat na problému určení integrálu

$$\int_0^1 \sqrt{1-x^2} dx, \quad (1)$$

který odpovídá spočítání obsahu plochy vymezené grafem funkce $f(x) = \sqrt{1-x^2}$, osou x a osou y .

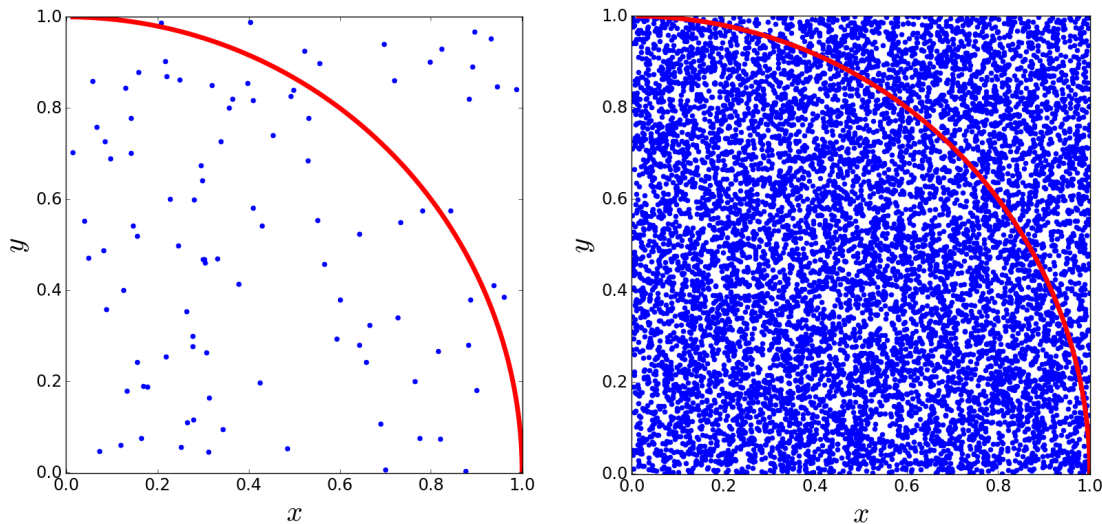
Grafem integrované funkce je na intervalu $\langle 0, 1 \rangle$ čtvrtina kružnice (viz obr. 2). Označme písmenem S plochu mezi tímto grafem, osou x a osou y . Nejprve vepíšeme plochu S do obrazce, jehož obsah známe, třeba do jednotkového čtverce. Metoda Monte Carlo funguje tak, že v jednotkovém čtverci náhodně generuje body. Pravděpodobnost, že vygenerovaný bod leží v S , je rovna poměru obsahu plochy S k obsahu plochy jednotkového čtverce, do kterého jsme ji uzavřeli.

Úkol 2.1 Odhadněte číslo π pomocí metody Monte Carlo. Využijeme k tomu integrál

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

jehož odhad nám dá přibližnou hodnotu $\pi/4$. ▲

Tento přístup je intuitivní a kromě simulace v počítači si jej lze vyzkoušet i v praxi,



Obrázek 2: Odhad obsahu plochy pod grafem funkce $f(x) = \sqrt{1-x^2}$ (zakreslena červeně) na intervalu $\langle 0, 1 \rangle$ pomocí generování souřadnic v rovině. Pokrytí plochy jednotkového čtverce pomocí 100 (vlevo) a 10 000 (vpravo) náhodně generovaných bodů (zakresleny modře).

například na písčné pláži. Generování bodů není nic jiného, než házení kamínků (náhodnou silou a náhodným směrem) do jednotkového čtverce, ve kterém máme nakreslený graf funkce, jejíž integrál chceme odhadnout. Poměr počtu kamínků, které spadly pod graf funkce, k celkovému počtu kamínků ve čtverci nám dá odhad daného integrálu.

Náhodná procházka

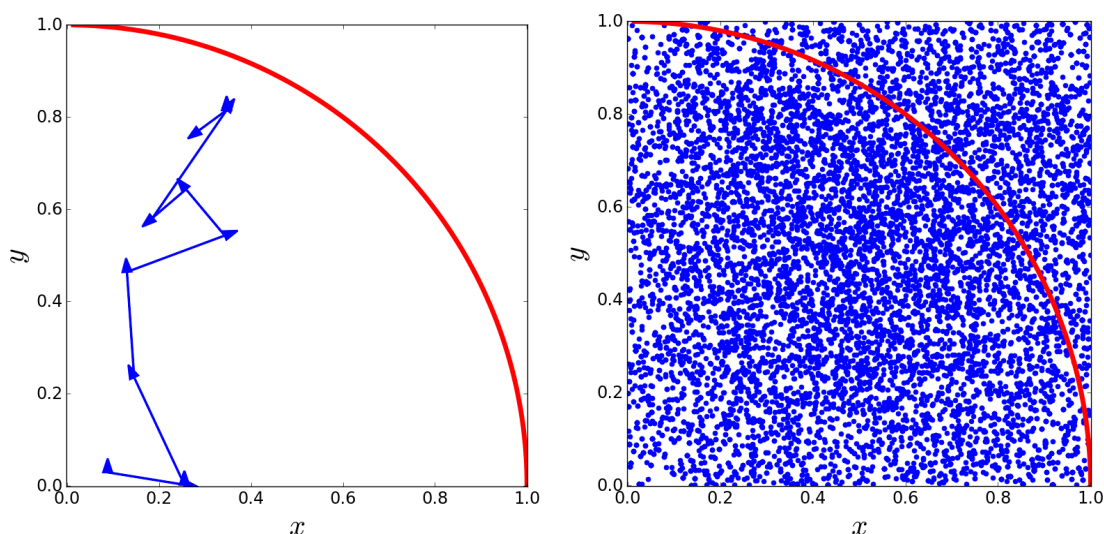
Zůstaňme na chvíli u představy simulace pomocí házení kamínků do písku a představme si, že se chce do simulace zapojit i malé dítě, které ještě nedohodí dost daleko. Pro jednoduchost uvažujme, že dítě může házet kamenem jen ve čtverci o délce strany δ . Pokud bychom pokračovali podle předchozích pravidel, nebyly by pozice ve čtverci generovány zcela náhodně. Je tedy třeba nějak upravit algoritmus.

Jedna z možností je, že se první kámen hodí náhodně do jednotkového čtverce a poté se bude konstruovat náhodná procházka podle následujících pravidel:

- každý hod začíná v místě dopadu předchozího hodu,
- hází se náhodně ve čtverci o délce strany δ , jehož střed je v místě, odkud se zrovna hází,
- pokud by nějaký hod mířil ven z jednotkového čtverce, kámen se položí na předchozí kámen a pokračuje se ze stejného místa (jinými slovy – hod se započítá, ale pozice zůstane stejná).

Tento způsob nám zaručí, že po čase budou kameny ve čtverci rozmístěny opravdu náhodně. Pak už můžeme opět pomocí poměru počtu kamínků pod grafem k celkovému počtu kamínků ve čtverci odhadnout hledaný obsah. Schéma náhodné procházky je na obr. 3.

Tento způsob je mnohem složitější a méně přesný než předchozí, proto se využívá jen v případech, kdy není možné použít jiný přístup. Nejčastěji se využívá v případech, kdy nejsme schopni náhodně vybírat čísla s požadovanou hustotou pravděpodobnosti. V tomto textu se s podobným případem nesetkáme – pokaždé volíme čísla s uniformní (rovnoměrnou) hustotou pravděpodobnosti.



Obrázek 3: Odhad obsahu plochy pod grafem funkce $f(x) = \sqrt{1-x^2}$ (zakreslena červeně) na intervalu $\langle 0, 1 \rangle$ pomocí generování souřadnic náhodnou procházkou. Konstrukce náhodné procházky s $\delta = 0,25$ pomocí 15 (vlevo) a 10 000 (vpravo) bodů (zakresleny modře).

Úkol 2.2 Odhadněte číslo π pomocí metody Monte Carlo s generováním bodů pomocí náhodné procházky. Opět se nám bude hodit integrál

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

jehož odhad nám dá přibližnou hodnotu $\pi/4$. ▲

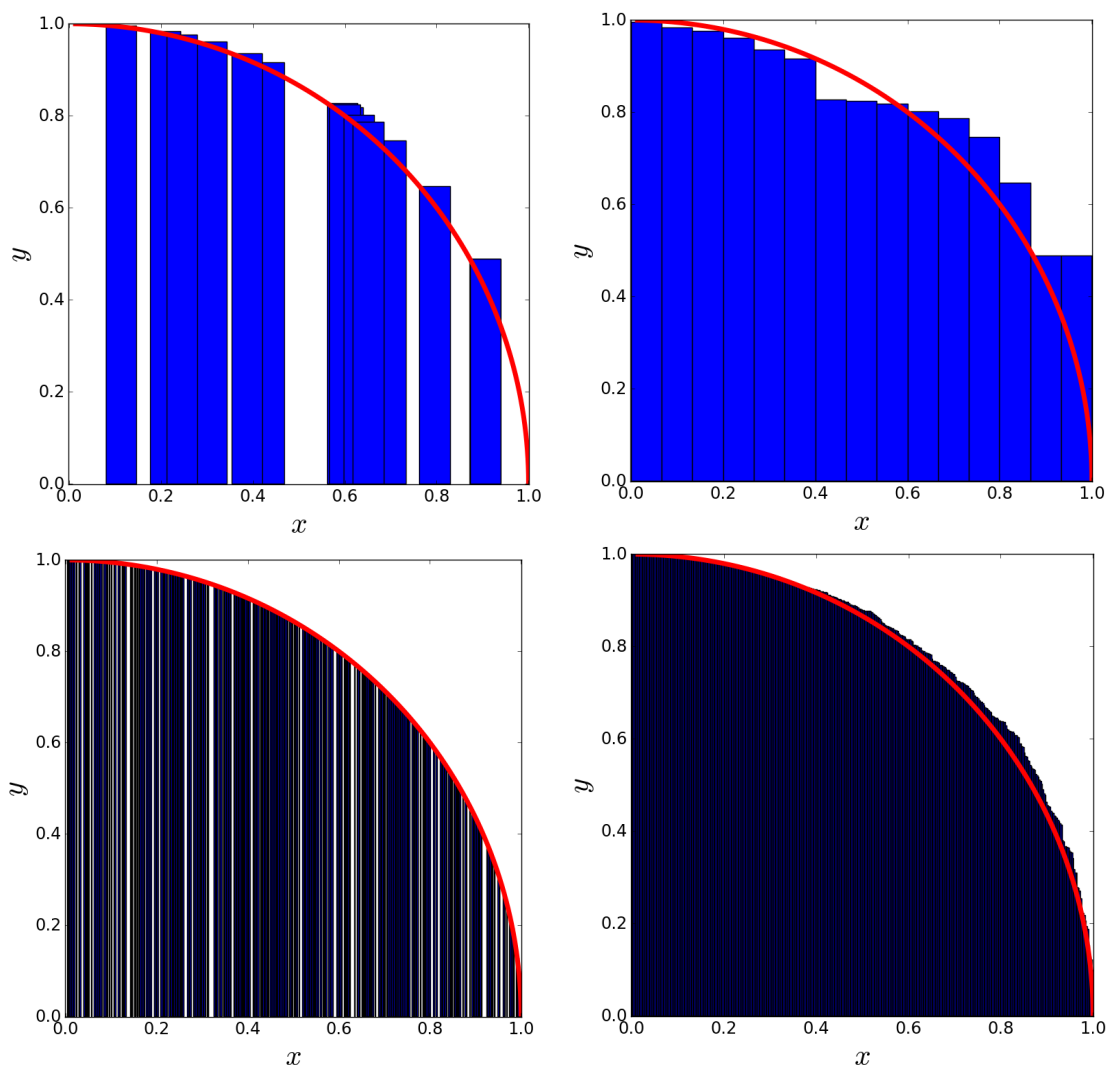
Crude Monte Carlo

Tato metoda je odvozena z definice určitého integrálu a využívá vzorec

$$\int_0^1 f(x) \, dx = \lim_{n \rightarrow \infty} \frac{1}{n} \sum_{i=1}^n f(x_i), \quad (2)$$

kde body x_i , ve kterých se funkce vyčísluje, jsou generovány náhodně na intervalu $\langle 0, 1 \rangle$.

Na rozdíl od předchozích metod, které odhadovaly plochu pod grafem pomocí poměru bodů pod grafem k celkovému počtu bodů, tento přístup počítá odhad pomocí průměru funkčních hodnot v náhodných bodech. Nepotřebujeme tedy generovat body v rovině, ale pouze v jednom rozměru. Jak tato metoda funguje, můžete vidět na obr. 4.



Obrázek 4: Odhad obsahu plochy pod grafem funkce $f(x) = \sqrt{1-x^2}$ na intervalu $\langle 0, 1 \rangle$ pomocí metody Crude Monte Carlo. Horní řada odpovídá simulaci s $n = 10$ vyčísleními, spodní odpovídá simulaci s $n = 500$ vyčísleními. V levé části jsou vykresleny jednotlivé příspěvky tak, jak byly generovány v průběhu simulace. Vpravo jsou seřazeny tak, aby se nepřekrývaly pro snazší porovnání s odhadovanou plochou.

Úkol 2.3 Odhadněte číslo π pomocí metody Crude Monte Carlo. Využijeme integrál

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

jehož odhad nám dá přibližnou hodnotu $\pi/4$. ▲

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

Definice 3.1 Bod $x^* \in \mathbb{R}$ nazýváme pevným bodem funkce $f : \mathbb{R} \mapsto \mathbb{R}$, pokud

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

Věta 3.1 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, že²

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

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 .

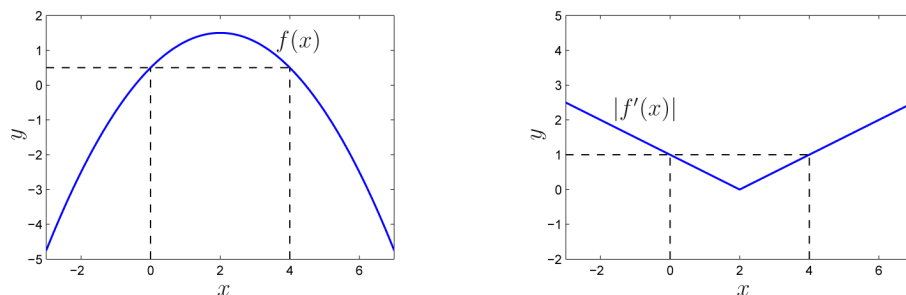
Lze ukázat, že čím více je hodnota nejmenšího horního odhadu množiny $\{|f'(x)| : x \in I\}$ blíží 0, tím rychleji posloupnost (x_n) konverguje k pevnému bodu f .

Pro dané x_0 uvažujme posloupnost (x_n) danou předpisem

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

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. 5), že posloupnost (3) konverguje k $\sqrt{2}$ pro libovolné $x_0 \in (0, 4)$.

²V 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. $\lim_{h \rightarrow 0+} \frac{f(x+h)-f(x)}{h}$.

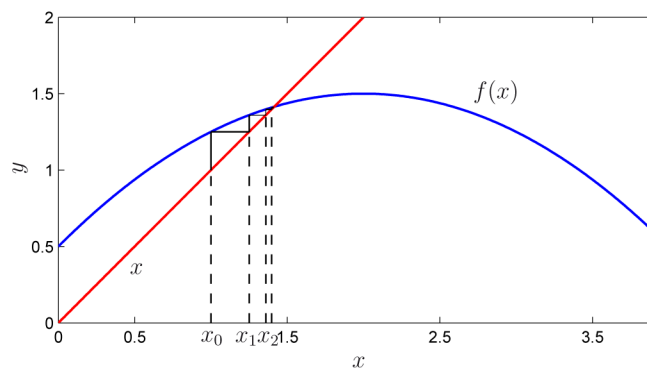


Obrázek 5: Grafy funkcí $f(x) = x - \frac{1}{4}(x^2 - 2)$ a $|f'(x)| = \left|1 - \frac{1}{2}x\right|$

Algoritmus 3.1 (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. 6 je ilustrována konvergence posloupnosti (3) k pevnému bodu $\sqrt{2}$ funkce f pro $x_0 = 1$.



Obrázek 6: Ilustrace konvergence metody prosté iterace k pevnému bodu $\sqrt{2} \doteq 1,41421356$ funkce $f(x) = x - \frac{1}{4}(x^2 - 2)$

Nalezněme nyní pomocí metody prosté iterace přibližné řešení rovnice

$$10 e^{-\frac{x}{10}} - x = 0. \quad (4)$$

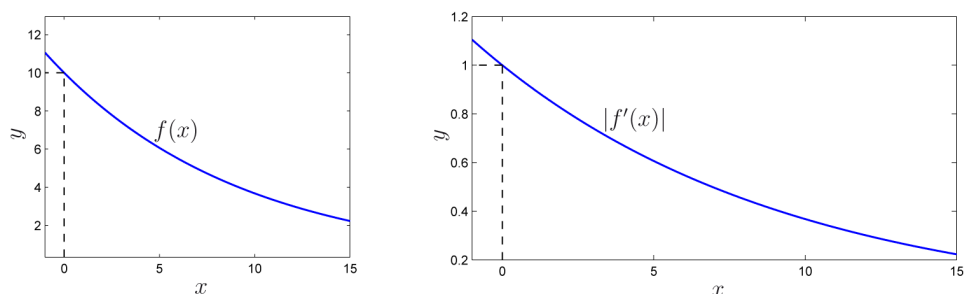
Řešení rovnice (4) převedeme na řešení rovnice

$$10 e^{-\frac{x}{10}} = x,$$

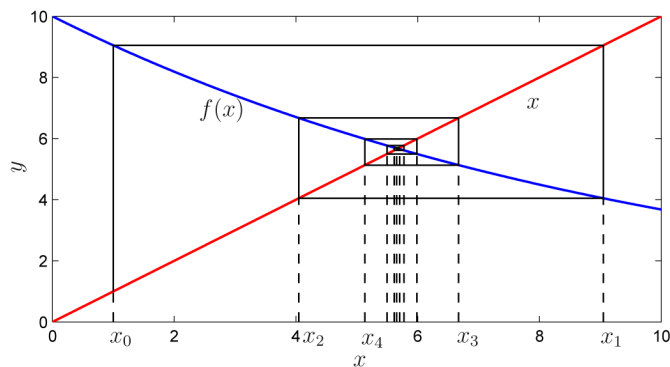
tj. budeme hledat pevný bod funkce $f(x) := 10 e^{-\frac{x}{10}}$. Lze si rozmyslet (viz také obr. 7), že posloupnost (x_n) daná předpisem

$$x_{n+1} := 10 e^{-\frac{x_n}{10}}, \quad n = 0, 1, 2, 3, \dots,$$

konverguje k pevnému bodu funkce f pro libovolné $x_0 > 0$. Na obr. 8 je ilustrována konvergence této posloupnosti k pevnému bodu funkce f pro $x_0 = 1$.



Obrázek 7: Grafy funkcí $f(x) = 10 e^{-\frac{x}{10}}$ a $|f'(x)| = | -e^{-\frac{x}{10}} |$



Obrázek 8: 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

$$g(x) = 0. \quad (5)$$

Předpokládejme, že g je spojitá a má spojitou a nenulovou derivaci v nějakém okolí³ $\mathcal{U}(x^*)$, kde x^* řeší rovnici (5). Budeme vycházet z metody prosté iterace. Protože x^* řeší rovnici (5), tak x^* řeší také rovnici

$$x = f(x) := x - \frac{g(x)}{h(x)}, \quad (6)$$

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

$$x_{n+1} := x_n - \frac{g(x_n)}{g'(x_n)}, \quad n = 0, 1, 2, 3, \dots, \quad (7)$$

konverguje k řešení x^* rovnice (5), pokud x_0 je dostatečně blízko x^* , funkce f je daná (6) a jsou splněny předpoklady věty ze str. 8.

Algoritmus 3.2 (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

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

³Okolím $\mathcal{U}(x^*)$ budeme rozumět nějaký otevřený omezený interval se středem v x^* .

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

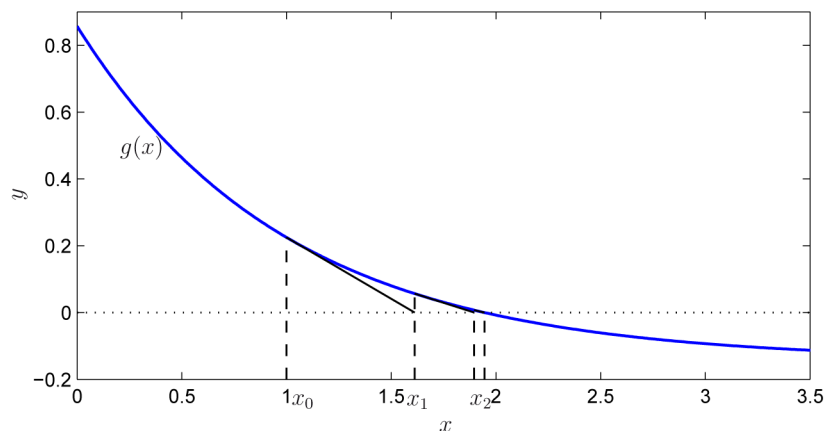
$$g'(x_n)(x - x_n) + g(x_n) = 0. \quad (8)$$

Je-li $g'(x_n) \neq 0$, pak lze řešení rovnice (8) psát ve tvaru

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

Porovnáme-li pak poslední rovnost s iterační formulí (7), zjistíme, že každé x_{n+1} vytvoříme jako průsečík tečny sestrojené 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. 9 ilustrujeme přibližné řešení rovnice $g(x) := e^{-x} - 1/7 = 0$ pomocí Newtonovy metody pro $x_0 = 1$.



Obrázek 9: Ilustrace konvergence Newtonovy metody k řešení $x^* \doteq 1,94591015$ rovnice $g(x) := e^{-x} - 1/7 = 0$

Úkol 3.1 Pomocí Newtonovy metody aproximujte řešení rovnic

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

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

Úkol 3.2 Pomocí Newtonovy metody najděte přibližné hodnoty čísel $\sqrt{2}$ a π .

Reference

- [1] K. Sigmon: *Matlab Primer*. University of Florida (1993).
- [2] 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>)

Definice A.1 Bud' $f : \mathbb{R} \mapsto \mathbb{R}$ a $x \in \mathbb{R}$. Existuje-li

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

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.

Věta A.1 Bud' $f, g : \mathbb{R} \mapsto \mathbb{R}$ 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 $g(x) \neq 0$.

Pozorování A.1

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