UNIVERZITA PALACKÉHO V OLOMOUCI
PŘÍRODOVĚDECKÁ FAKULTA
KATEDRA MATEMATICKÉ ANALÝZY A APLIKACÍ MATEMATIKY

BAKALÁŘSKÁ PRÁCE

Newtonova metoda pro úlohu nepodmíněné optimalizace

Vedoucí bakalářské práce:
RNDr. Jitka Machalová, Ph.D.
Rok odevzdání: 2010

Vypracoval:
Richard Andrášik
MAP, III. ročník

Prohlášení

Prohlašuji, že jsem vytvořil tuto bakalářskou práci samostatně za vedení a pomoci RNDr. Jitky Machalové, Ph.D. a že jsem v seznamu použité literatury uvedl všechny zdroje použité při zpracování práce.

V Olomouci dne 7. dubna 2010

Poděkování

Rád bych na tomto místě poděkoval zejména RNDr. Jitce Machalové, Ph.D. jako mé vedoucí bakalářské práce za obětavou spolupráci i za čas, který mi věnovala při konzultacích, a svým rodičům za podporu po celou dobu mého studia.

Obsah

Použité symboly

$\mathbb{N}$	obor přirozených čísel
$\mathbb{N}_0$	$\mathbb{N}_0 = \mathbb{N} \cup \{0\}$
$\mathbb{R}$	obor reálných čísel
$\mathbb{R}^+$ $(\mathbb{R}_0^+)$	obor kladných (nezáporných) reálných čísel
$\mathbb{R}^-$ $(\mathbb{R}_0^-)$	obor záporných (nekladných) reálných čísel
$\mathbb{R}^n$	vektorový prostor dimenze n nad $\mathbb{R}$
$\mathbb{R}^{n\times m}$	prostor matic typu (n, m) nad $\mathbb{R}$
(a, b)	otevřený interval
$\langle a, b\rangle$	uzavřený interval
$\mathbf{A} = (a_{ij})_{i,j=1}^n$	čtvercová matice řádu n
$\mathbf{A}^{\mathrm{T}}$	transponovaná matice
$\mathbf{E}$	jednotková matice
$\mathbf{A} \preceq \mathbf{B}$	nerovnost mezi symetrickými maticemi
$\mathbf{A} \prec \mathbf{B}$	ostrá nerovnost mezi symetrickými maticemi
$x = (x_1, \ldots, x_n)$	uspořádaná n-tice reálných čísel – vektor
$o = (0, 0, \ldots, 0)$	nulový vektor
$\|x\|_{\mathbf{A}}$	norma vektoru x definovaná maticí $\mathbf{A}$
$\|x\|_2$	eukleidovská norma vektoru x
$\|\mathbf{A}\|_2$	norma matice $\mathbf{A}$ souhlasná s eukleid. normou
$\{x_k\}$	posloupnost bodů
$\mathcal{D}(f)$	definiční obor funkce f
$\mathcal{C}^n(\mathcal{M})$	množina všech funkcí, které mají spojité všechny derivace na množině $\mathcal{M}$ až do řádu n, kde $n \in \mathbb{N}_0$
$\nabla f(x)$	gradient funkce $f(x)$ v bodě x
$\mathbf{D}^2 f(x)$	Hessova matice funkce $f(x)$ v bodě x

Úvodní slovo

Matematika za několik tisíciletí své existence urazila dlouhou a velice náročnou cestu, při níž se její tvář mnohokráte změnila, avšak její podstata zůstala stejná. Je to nejspíše kvůli různým pohledům ve vnímání přírody, u níž se matematikové inspirují, a stejnému cíli – porozumět světu okolo nás.

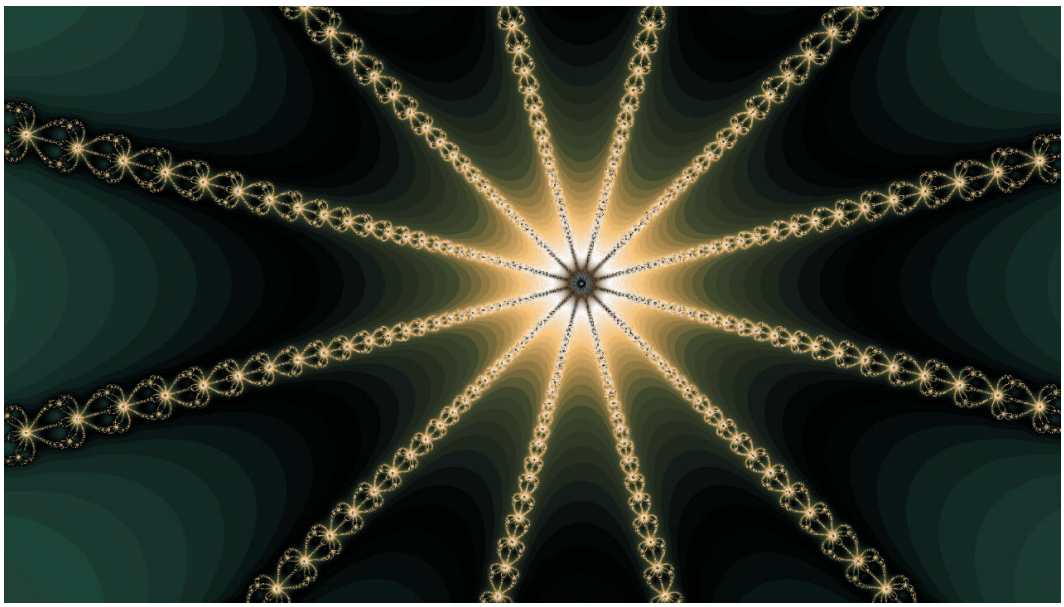
> *„Filozofie světa je obsažena v grandiózní knize, stále otevřené všem a každému – myslím tím knihu přírody. Porozumět jí však může jen ten, kdo se naučí jejímu jazyku a písmu, jímž je napsána. Napsána je jazykem matematiky a jejím písmem jsou matematické vzorce.“*
>
> *Galileo Galilei*

Zpočátku, ve starověkém Řecku a Římě, se matematikové zabývali zejména algebrou a geometrií, což vyplynulo z nutnosti a každodenního užívání matematiky obyčejnými lidmi. Například hrnčíř musel umět popsat geometrický tvar vázy, kterou vytvořil, stejně jako pastýř dokázat určit počet ovcí na louce, aby měl jistotu, že se mu žádná neztratila, ale pouze matematik zkoumal geometrické tvary, aniž by přemýšlel o váze, a čísla beze vztahu k předmětům či ovcím. Poněvadž to platí dodnes, můžeme říci, že matematik hledá vztahy a vlastnosti obecně pro pojmy, které si vytvořil ve svých představách na základě pozorování přírody, a poté se je snaží aplikovat na skutečné problémy týkající se všedního života, čemuž se říká aplikovaná matematika.

Základním stavebním kamenem aplikované matematiky je matematická analýza, jejíž základy položili Isaac Newton a Gottfried Leibnitz na konci 17. století. Nejdříve se matematické aplikace objevily ve fyzice a astronomii, poté v dalších exaktních vědách jako je třeba chemie a od počátku 20. století se rozšiřují i do ostatních vědních oborů. Vzhledem k velice rychlému rozvoji informačních technologií se v dnešní době rovněž rychle vyvíjí i numerická matematika, jež je jedním z odvětví aplikované matematiky a zabývá se řešením problémů pro určitá konkrétní data. Hlavním úkolem numerické matematiky je najít postup – algoritmus, jenž by řešil daný problém co nejpřesněji, nejrychleji a nejjednodušeji.

Numerická matematika se zabývá například interpolací a aproximací, numerickým derivováním a integrováním, řešením soustav rovnic, lineárních i nelineárních, a v neposlední řadě také matematickou optimalizací.

Cílem této práce je seznámit čtenáře se základy problematiky optimalizace, s Newtonovou metodou, jež řeší optimalizační problém, ukázat její výhody a nevýhody a prezentovat aplikaci metody na příkladech. Dále je zde zaměřena pozornost na sestavení programů pro řešení optimalizačních úloh ve výpočetním systému MATLAB a v neposlední řadě také na množinu sebeomezujících funkcí a jejich využití v optimalizaci.



Obrázek 0: Rozmanitost matematiky.

1 Přípravná kapitola

Tato kapitola se zabývá vysvětlením některých důležitých pojmů, jež jsou nezbytné k porozumění následujícímu textu. Zejména pojednává o základech diferenciálního počtu funkcí více reálných proměnných a vlastnostech symetrických matic.

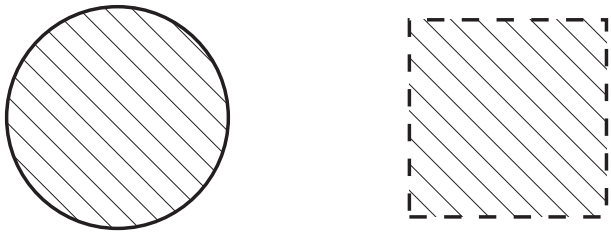
1.1 Kompaktní množiny

Definice 1.1. Řekneme, že množina $\mathcal{A} \subseteq \mathbb{R}^n$ je **kompaktní**, jestliže z každé posloupnosti bodů množiny $\mathcal{A}$ lze vybrat konvergentní podposloupnost, jejíž limita leží v $\mathcal{A}$.

Věta 1.1. *Množina $\mathcal{A} \subseteq \mathbb{R}^n$ je kompaktní právě tehdy, když je omezená a uzavřená v $\mathbb{R}^n$.*

Důkaz: viz [9], str. 13.

Příklad 1.1. Pro $n = 1$, tzn. v $\mathbb{R}$, jsou kompaktní všechny uzavřené intervaly $\langle a, b\rangle$, kde a, $b \in \mathbb{R}$. Otevřené a polouzavřené intervaly kompaktní nejsou. Položíme-li $n = 2$, pak v $\mathbb{R}^2$ je kompaktní například množina na obrázku 1 vlevo, druhá množina na tomto obrázku kompaktní není.



Obrázek 1: Kompaktní a nekompaktní množina v $\mathbb{R}^2$.

1.2 Definitnost matice

Definice 1.2. Symetrickou čtvercovou matici $\mathbf{A}$ řádu n nazýváme

- **pozitivně definitní**, jestliže $\forall x \in \mathbb{R}^n$, $x \neq o$: $x\mathbf{A}x^{\mathrm{T}} > 0$.

- **pozitivně semidefinitní**, jestliže $\forall x \in \mathbb{R}^n$: $x\mathbf{A}x^{\mathrm{T}} \geq 0$.

- **negativně definitní**, jestliže $\forall x \in \mathbb{R}^n$, $x \neq o$: $x\mathbf{A}x^{\mathrm{T}} < 0$.

- **negativně semidefinitní**, jestliže $\forall x \in \mathbb{R}^n$: $x\mathbf{A}x^{\mathrm{T}} \leq 0$.

- **indefinitní**, jestliže $\exists\, x, y \in \mathbb{R}^n$: $x\mathbf{A}x^{\mathrm{T}} > 0$, $y\mathbf{A}y^{\mathrm{T}} < 0$.

Věta 1.2. (Sylvestrovo kritérium) *Nechť je dána symetrická čtvercová matice* $\mathbf{A} = (a_{ij})_{i,j=1}^{n}$. *Označme hlavní determinanty této matice*

$$\Delta_k = \begin{vmatrix} a_{11} & \dots & a_{1k} \\ \dots & \dots & \dots \\ a_{k1} & \dots & a_{kk} \end{vmatrix}, \quad k = 1, \dots, n. \tag{1}$$

Matice $\mathbf{A}$ *je pozitivně definitní právě tehdy, když* $\Delta_k > 0$, *pro* $k = 1, \dots, n$, *a negativně definitní, když* $(-1)^k \Delta_k > 0$.

Důkaz: viz [4], str. 53.

Poznámka 1.1. Symetrická čtvercová matice $\mathbf{A}$ je pozitivně (resp. negativně) definitní, jsou-li všechna její vlastní čísla kladná (resp. záporná).
Symetrická čtvercová matice $\mathbf{A}$ je indefinitní právě tehdy, když je aspoň jedno její vlastní číslo kladné a zároveň je aspoň jedno její vlastní číslo záporné.

Definice 1.3. Řekneme, že dvě symetrické čtvercové matice $\mathbf{A}$, $\mathbf{B}$ jsou ve vztahu $\mathbf{A} \succ \mathbf{B}$ (resp. $\mathbf{A} \succeq \mathbf{B}$), je-li matice $\mathbf{A} - \mathbf{B}$ pozitivně definitní (resp. pozitivně semidefinitní). Relaci $\succ$ (resp. $\succeq$) nazýváme **ostrá** (resp. **neostrá**) **nerovnost mezi symetrickými maticemi**.

Definice 1.4. Mějme bod $x \in \mathbb{R}^n$, pak

$$\|x\|_{\mathbf{A}} = (x\mathbf{A}x^{\mathrm{T}})^{\frac{1}{2}} \tag{2}$$

je jeho kvadratická norma definovaná symetrickou čtvercovou maticí $\mathbf{A}$.

1.3 Gradient a Hessova matice

Definice 1.5. O funkci $f : \mathcal{D} \to \mathbb{R}$, $\mathcal{D} \subseteq \mathbb{R}^n$, řekneme, že je **m-krát spojitě diferencovatelná** na množině $\mathcal{A} \subseteq \mathcal{D}$, pokud má na $\mathcal{A}$ spojité všechny své parciální derivace až do řádu m. Zapisujeme $f \in \mathrm{C}^m(\mathcal{A})$. Jestliže má funkce f na $\mathcal{A}$ derivace všech řádů, pak $f \in \mathrm{C}^\infty(\mathcal{A})$ a říkáme, že je **hladká**.

Definice 1.6. Nechť je dána funkce $f : \mathcal{D} \to \mathbb{R}$, $\mathcal{D} \subseteq \mathbb{R}^n$. Mějme bod $x_0 \in \mathcal{D}$ a vektor $\alpha \in \mathbb{R}^n$.

1. Jesliže existuje limita

$$f'_\alpha(x_0) = \lim_{t \to 0} \frac{f(x_0 + t\alpha) - f(x_0)}{t}, \tag{3}$$

   pak ji nazýváme **směrovou derivací prvního řádu** funkce f v bodě x_0 podle vektoru α.

2. Nechť existuje derivace $f^{(m-1)}_{\alpha\ldots\alpha}$, $m \in \mathbb{N}$. Položme $f^{(0)}(x_0) = f(x_0)$. Existuje-li limita

$$f^{(m)}_{\alpha\ldots\alpha}(x_0) = \lim_{t \to 0} \frac{f^{(m-1)}_{\alpha\ldots\alpha}(x_0 + t\alpha) - f^{(m-1)}_{\alpha\ldots\alpha}(x_0)}{t}, \tag{4}$$

   nazýváme ji **směrovou derivací m-tého řádu** funkce f v bodě x_0 podle vektoru α.

Poznámka 1.2. Řekneme, že na množině $\mathcal{M} \subseteq \mathcal{D}$ existuje směrová derivace m-tého řádu, podle vektoru α, jestliže existují směrové derivace m-tého řádu podle vektoru α ve všech bodech množiny $\mathcal{M}$.

Definice 1.7. Mějme dánu funkci $f : \mathcal{D} \to \mathbb{R}$, $\mathcal{D} \subseteq \mathbb{R}^n$, jež je spojitě diferencovatelná na množině $\mathcal{A} \subseteq \mathcal{D}$, pak existuje **gradient** $\nabla f(x)$ funkce $f(x)$ v bodě x a je následujícího tvaru

$$\nabla f(x) = \left(\frac{\partial}{\partial x_1} f(x), \cdots, \frac{\partial}{\partial x_n} f(x) \right) \quad , \ \forall x \in \mathcal{A}. \tag{5}$$

Definice 1.8. Nechť je dána funkce $f : \mathcal{D} \to \mathbb{R}$, $\mathcal{D} \subseteq \mathbb{R}^n$, která je na množině $\mathcal{A} \subseteq \mathcal{D}$ dvakrát spojitě diferencovatelná, pak existuje **Hessova matice** $\mathbf{D}^2 f(x)$ funkce $f(x)$ v bodě x a má tvar

$$\mathbf{D}^2 f(x) = \left(\frac{\partial^2}{\partial x_i \partial x_j} f(x) \right)_{i,j=1}^{n} , \ \forall x \in \mathcal{A}. \tag{6}$$

Poznámka 1.3. Buď dána funkce $f : \mathcal{D} \to \mathbb{R}$, $\mathcal{D} \subseteq \mathbb{R}^n$, která je na množině $\mathcal{A} \subseteq \mathcal{D}$ dvakrát spojitě diferencovatelná. Na $\mathcal{A}$ tudíž existuje její Hessova matice. Mějme bod $x^* \in \mathcal{A}$, je-li $\mathbf{D}^2 f(x^*)$ pozitivně definitní, pak existuje okolí $\mathcal{U}$ bodu x^* takové, že pro libovolné $x \in \mathcal{U}$ je $\mathbf{D}^2 f(x)$ rovněž pozitivně definitní.

Věta 1.3. *Nechť je dána funkce $f : \mathcal{D} \to \mathbb{R}$, $\mathcal{D} \subseteq \mathbb{R}^n$, která je na množině $\mathcal{A} \subseteq \mathcal{D}$ dvakrát spojitě diferencovatelná a ryze konvexní (viz definice 1.10). Pak existují reálné konstanty m, M takové, že $0 < m < M$, pro něž platí*

$$m\mathbf{E} \preceq \mathbf{D}^2 f(x) \preceq M\mathbf{E}, \quad \forall x \in \mathcal{A}. \tag{7}$$

Důkaz: viz [1] str. 460.

Podívejme se blíže na geometrickou interpretaci gradientu.

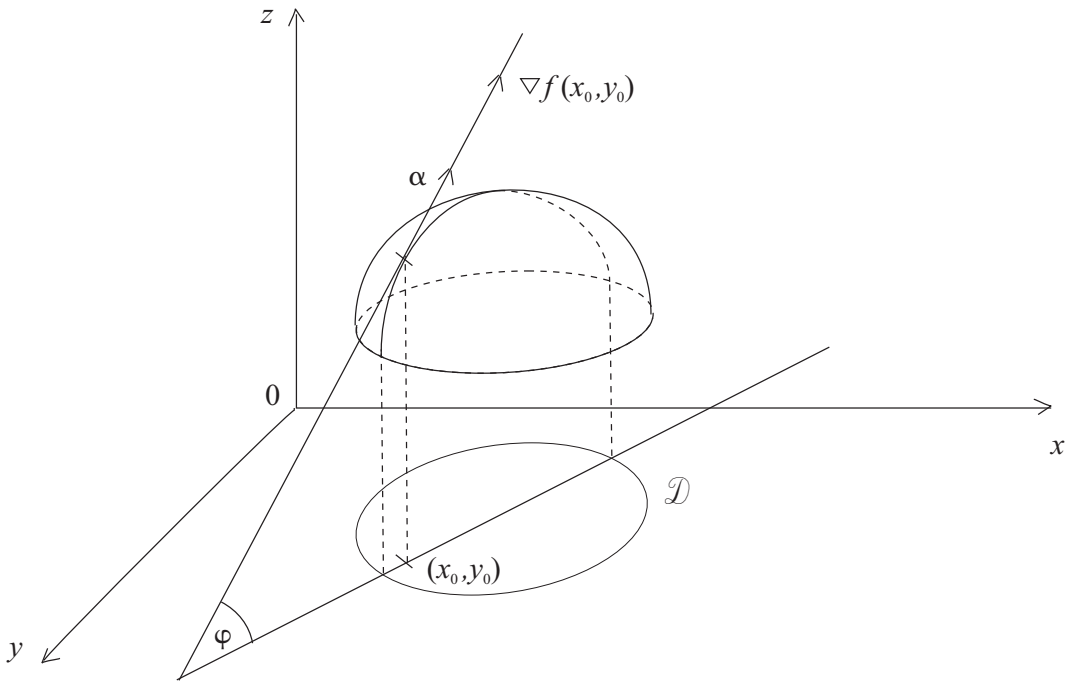
Příklad 1.2. Nechť je dána funkce $f(x_1, \ldots, x_n)$, její definiční obor $\mathcal{D} \subseteq \mathbb{R}^n$ a $x_0 \in \mathcal{D}$. Najděte takový vektor α, který určuje směr nejrychlejšího růstu (resp. spádu) funkce f v bodě x_0, $\|\alpha\|_2 = 1$.

Řešení: Poněvadž směrová derivace prvního řádu podle α je rovna směrnici tečny, budeme hledat vektor α, pro který bude směrová derivace maximální, a tedy bude maximální i směrový úhel. Pro směrovou derivaci platí

$$f'_\alpha(x_0) = \alpha \cdot \nabla f(x_0) = \|\alpha\|_2 \cdot \|\nabla f(x_0)\|_2 \cdot \cos\varphi$$

Odtud vidíme, že pro $\varphi = 0$, tedy $\cos\varphi = 1$, bude směrová derivace $f'_\alpha(x_0)$ maximální, z čehož vyplývá, že směr nejrychlejšího růstu funkce f v bodě x_0 je rovnoběžný s gradientem, a tudíž hledaný vektor α má tvar

$$\alpha = \frac{\nabla f(x_0)}{\|\nabla f(x_0)\|_2}$$



Obrázek 2: Geometrická interpretace gradientu v $\mathbb{R}^2$.

Podobně lze ukázat, že vektor

$$\beta = -\frac{\nabla f(x_0)}{\|\nabla f(x_0)\|_2}$$

určuje směr nejrychlejšího spádu funkce f v bodě x_0.

1.4 Taylorův vzorec

Věta 1.4. (Taylorova) *Nechť je dána funkce $f : \mathcal{D} \to \mathbb{R}$, $\mathcal{D} \subseteq \mathbb{R}^n$, a nechť existují všechny její směrové derivace až do řádu $m+1$ na $\mathcal{D}$. Mějme dány body x_0, $x_0 + \alpha \in \mathcal{D}$ a úsečku tyto body spojující, jež celá leží v $\mathcal{D}$. Pak existuje $\theta \in (0,1)$ tak, že platí*

$$\begin{aligned} f(x_0+\alpha) = f(x_0) + \tfrac{1}{1!} f'_\alpha(x_0) + \tfrac{1}{2!} f''_{\alpha,\alpha}(x_0) + \ldots + \tfrac{1}{m!} f^{(m)}_{\alpha\ldots\alpha}(x_0) + \\ + \tfrac{1}{(m+1)!} f^{(m+1)}_{\alpha\ldots\alpha}(x_0+\theta\alpha) \end{aligned} \tag{8}$$

Důkaz: viz [9], str. 81.

Poznámka 1.4. Lze ukázat (viz [9], str. 57 a 66), že existují-li směrové derivace prvního a druhého řádu funkce $f(x)$ v bodě x_0 podle vektoru α, pak pro ně platí

$$f'_\alpha(x_0) = \alpha \cdot \nabla f(x_0)^{\mathrm{T}}, \tag{9}$$

$$f''_{\alpha,\alpha}(x_0) = \alpha \cdot \mathbf{D}^2 f(x_0) \cdot \alpha^{\mathrm{T}}. \tag{10}$$

1.5 Konvexní funkce

Definice 1.9. Množina $\mathcal{M}$ se nazývá **konvexní**, jestliže pro libovolné body x_1, $x_2 \in \mathcal{M}$ platí

$$\lambda x_1 + (1-\lambda)x_2 \in \mathcal{M}, \quad \forall \lambda \in (0;1).$$

Definice 1.10. Řekneme, že funkce $f : \mathcal{D}(f) \to \mathbb{R}$, kde $\mathcal{D}(f) \subseteq \mathbb{R}^n$, je **konvexní** na otevřené konvexní množině $\mathcal{M} \subseteq \mathcal{D}(f)$, jestliže pro libovolné $\lambda \in (0;1)$ platí

$$f(\lambda x_1 + (1-\lambda)x_2) \leq \lambda f(x_1) + (1-\lambda) f(x_2), \quad \forall x_1,\ x_2 \in \mathcal{M}.$$

Jestliže je navíc v předchozím vztahu vždy splněna ostrá nerovnost pro $x_1 \neq x_2$, řekneme, že funkce f je **ryze konvexní** na $\mathcal{M}$.

Poznámka 1.5. Je-li funkce $f : \mathcal{D}(f) \to \mathbb{R}$, kde $\mathcal{D}(f) \subseteq \mathbb{R}^n$, konvexní (resp. ryze konvexní) na svém definičním oboru, pak jednoduše říkáme, že je funkce f konvexní (resp. ryze konvexní).

Poznámka 1.6. Nechť je funkce f z definice 1.10 dvakrát spojitě diferencovatelná. Lze ukázat, že f je konvexní (resp. ryze konvexní) na otevřené konvexní množině $\mathcal{M} \subseteq \mathcal{D}(f)$ právě tehdy, když je její Hessova matice pozitivně semidefinitní (resp. pozitivně definitní) pro všechna $x \in \mathcal{M}$.

Další základní poznatky o vlastnostech konvexních funkcí jsou uvedeny například v [2] str. 27 nebo v [8]. Pro naše účely postačí, budeme-li znát tvrzení předchozí poznámky.

2 Úvod do problematiky optimalizace

Optimalizace je matematická disciplína zabývající se studiem problémů a hledající nejlepší možné řešení. Historicky první numerickou optimalizační metodu uvedl A. Cauchy, jednalo se o **metodu nejrychlejšího spádu**.

Úloha, pro jejíž řešení byl vybudován rozsáhlý aparát matematické optimalizace, zní následovně: **„Najděte optimální řešení daného problému.“** Z matematického hlediska je to myšleno velice jednoduše. Intuitivně můžeme říci, že máme nějakou vhodnou funkci f, kterou je popsán daný jev či problém, a dále rovnice a nerovnice, které omezují množinu řešení. Úkolem je určit, ve kterém bodě, jenž vyhovuje daným rovnicím a nerovnicím, má funkce f své **minimum** respektive **maximum**. Budeme tedy hledat **extrémy** funkce f, což můžeme řešit pomocí klasických optimalizačních postupů nebo použitím různých numerických metod, ať už s větší či menší přesností a úspěšností.

Tato kapitola má za úkol popsat základní pojmy z problematiky matematické optimalizace, osvětlit klasickou optimalizační metodu a seznámit čtenáře s nutnými a postačujícími podmínkami optimality.

Zadání úlohy, jež se řeší pomocí metod matematické optimalizace, může znít například následovně.

Příklad 2.1. Poraďte vedoucímu zemědělského závodu, který dostal za úkol nakoupit traktory. K dispozici má kapitál v hodnotě 21 mil. a 28 lidí, kteří jsou ochotni na traktorech pracovat. Má na výběr ze dvou typů traktorů. První typ stojí 3 mil., na jeho provoz je potřeba dvou pracovníků a průměrný měsíční zisk činí 204 tisíc, traktor druhého typu by mohl zakoupit za 4 mil., k jeho provozu je potřeba tří pracovníků a průměrný měsíční zisk je 340 tisíc. Kolik traktorů kterého typu má vedoucí nakoupit, aby byl výsledný zisk co nejvyšší?

Již z tohoto příkladu, je vidět, že optimalizace má, a jistě i mít bude, v praxi široké možnosti využití. A to nejen v zemědělství, ekonomice či financích, ale i ve fyzice, medicíně a mnoha dalších oborech.

2.1 Základní pojmy a klasická optimalizační metoda

Nechť je dána funkce $f : \mathcal{D} \to \mathbb{R}$, kde $\mathcal{D} \subseteq \mathbb{R}^n$. Hledáme extrémy funkce f na množině $\mathcal{D}$. Poněvadž je maximum funkce f stejné jako mínus minimum funkce $-f$, postačí, když dále budeme mluvit jen o minimu funkce, pro maximum funkce platí analogické úvahy a tvrzení.

Víme tedy již, že hledáme minimum funkce na dané množině $\mathcal{D}$. Ideální je najít globální minimum této funkce.

Definice 2.1. Nechť je dána funkce $f : \mathcal{D} \to \mathbb{R}$, $\mathcal{D} \subseteq \mathbb{R}^n$, $x^* \in \mathcal{D}$ nazveme bodem **globálního minima**, jestliže $f(x^*) \leq f(x)$ pro všechna $x \in \mathcal{D}$.

Obvykle je velmi těžké najít globální minimum funkce, poněvadž zpravidla nevíme, jak funkce vypadá na celém svém definičním oboru. Známe pouze její průběh v určitých lokálních oblastech, a tedy pomocí většiny numerických metod jsme schopni najít pouze lokální minimum dané funkce.

Definice 2.2. Nechť je dána funkce $f : \mathcal{D} \to \mathbb{R}$, $\mathcal{D} \subseteq \mathbb{R}^n$, $x^* \in \mathcal{D}$ nazveme bodem **lokálního minima**, jestliže existuje okolí $\mathcal{U}$ bodu x^* takové, že $f(x^*) \leq f(x)$ pro všechna $x \in \mathcal{U}$. Takovéto lokální minimum nazýváme **neostré**. Pokud navíc platí $f(x^*) < f(x)$ pro všechna $x \in \mathcal{U}$, $x \neq x^*$, bod x^* nazýváme **ostrým** lokálním minimem funkce f.

Definice 2.3. Bod x^* nazveme **řešením** optimalizačního problému, pokud je globálním resp. lokálním minimem dané funkce f na $\mathcal{D}$ resp. na okolí $\mathcal{U}$ bodu x^*.

Věta 2.1. (Weierstrassova) *Nechť je funkce $f : \mathcal{D} \to \mathbb{R}$, $\mathcal{D} \subseteq \mathbb{R}^n$, spojitá na kompaktní množině $\mathcal{A} \subseteq \mathcal{D}$, pak f nabývá na množině $\mathcal{A}$ svého minima a maxima.*

Důkaz: viz [9], str. 26.

Následující věty se týkají nutných a postačujících podmínek pro minimalitu bodu x^* dané funkce a jsou základními stavebními kameny klasických postupů matematické optimalizace (viz příklad 2.2).

Věta 2.2. (Nutná podmínka prvního řádu) *Nechť funkce $f : \mathcal{D} \to \mathbb{R}$, $\mathcal{D} \subseteq \mathbb{R}^n$, je spojitě diferencovatelná na nějakém okolí $\mathcal{U}$ bodu x^*. Jestliže je $x^* \in \mathcal{D}$ lokálním minimem funkce $f(x)$, pak platí*

$$\nabla f(x^*) = \left(\frac{\partial}{\partial x_1} f(x^*), \ldots, \frac{\partial}{\partial x_n} f(x^*) \right) = o. \tag{11}$$

Důkaz: viz [7], str. 15.

Věta 2.3. (Nutná podmínka druhého řádu) *Nechť funkce $f : \mathcal{D} \to \mathbb{R}$, $\mathcal{D} \subseteq \mathbb{R}^n$, je dvakrát spojitě diferencovatelná na nějakém okolí $\mathcal{U}$ bodu $x^* \in \mathcal{D}$. Jestliže je bod x^* lokálním minimem funkce $f(x)$, pak je Hessova matice funkce $f(x)$ v bodě x^* pozitivně semidefinitní.*

Důkaz: viz [7], str. 15.

Věta 2.4. (Postačující podmínka druhého řádu) *Nechť funkce $f : \mathcal{D} \to \mathbb{R}$, kde $\mathcal{D} \subseteq \mathbb{R}^n$, je dvakrát spojitě diferencovatelná na nějakém okolí $\mathcal{U}$ bodu x^*. Jestliže je $\nabla f(x^*) = o$ a zároveň $\mathbf{D}^2 f(x^*)$ je pozitivně definitní, pak je x^* bodem ostrého lokálního minima funkce f.*

Důkaz: Podstatou důkazu je použití Taylorova vzorce pro funkci f v bodě x_0, kde x_0 je z dostatečně malého okolí bodu x^*, a tedy je matice $\mathbf{D}^2 f(x_0)$ také pozitivně definitní. Podle předpokladů věty platí $\nabla f(x^*) = o$. Z toho nám vyplyne tvrzení. Podrobněji viz [7], str. 16.

□

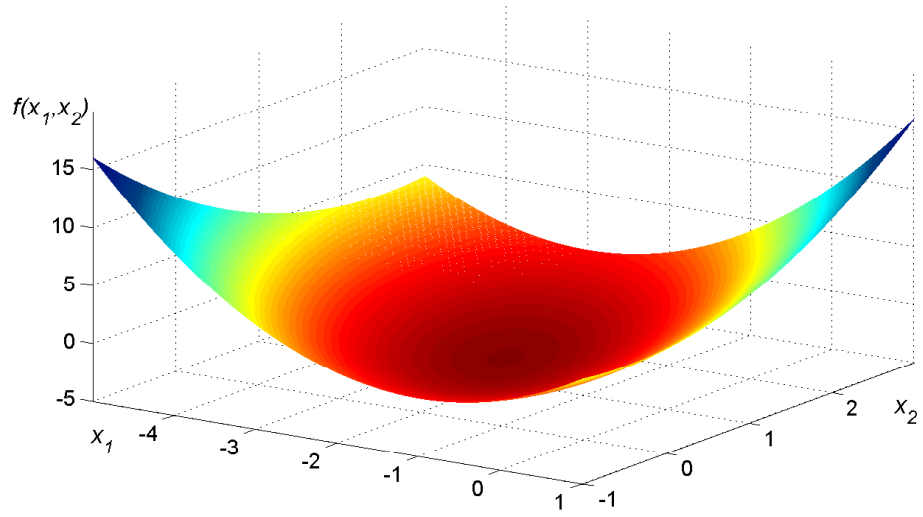
Příklad 2.2. Je dána funkce $f(x_1, x_2) = (x_1 + x_2)^2 - x_1(x_2 - 3)$. Zjistěte, ve kterém bodě má tato funkce lokální minimum.

Řešení: Definičním oborem funkce f je celé $\mathbb{R}^2$, uvažujeme tedy $\mathcal{D} = \mathbb{R}^2$. Funkce $f(x_1, x_2)$ je jistě dvakrát spojitě diferencovatelná. Nejdříve určíme $\nabla f(x_1, x_2)$ a $\mathbf{D}^2 f(x_1, x_2)$ a poté zjistíme, za jakých okolností platí předpoklady věty 2.4. Tedy

$$\nabla f(x_1, x_2) = (2x_1 + x_2 + 3,\ x_1 + 2x_2)\,,$$

$$\mathbf{D}^2 f(x_1, x_2) = \begin{pmatrix} 2 & 1 \\ 1 & 2 \end{pmatrix}.$$

Rovnost $\nabla f(x_1, x_2) = o$ je splněna pro $x_1 = -2$, $x_2 = 1$. Hlavní determinanty Hessovy matice jsou $\Delta_1 = 2$, $\Delta_2 = 3$, a tedy podle Sylvestrova kritéria je $\mathbf{D}^2 f(x_1, x_2)$ pozitivně definitní pro všechna $[x_1, x_2] \in \mathbb{R}^2$. Z toho podle věty 2.4 vyplývá, že funkce $f(x_1, x_2)$ má v bodě $[-2, 1]$ ostré lokální minimum.



Obrázek 3: Graf funkce $f(x_1, x_2) = (x_1 + x_2)^2 - x_1(x_2 - 3)$.

Mohlo by se zdát, že lze každou úlohu při hledání minima dané funkce řešit tímto způsobem, avšak mnohdy je jednodušší a rychlejší použít některou z numerických metod matematické optimalizace.

3 Numerické optimalizační metody

Podstatou úspěšného vyřešení optimalizačního problému je správné přeformulování zadané slovní úlohy do jazyka matematiky. Následuje pak vhodný výběr a použití postupu vedoucího k řešení, čímž se zabývá numerická matematika, která tak dala vzniknout numerickým optimalizačním metodám, jež našly na tomto poli své uplatnění.

Tato kapitola se zabývá matematickým popisem zadání úlohy optimalizace a má za úkol osvětlit čtenáři základní pojmy z této problematiky. Je zde rovněž objasněn rozdíl mezi podmíněnou a nepodmíněnou optimalizací a také vysvětlena podstata numerických metod matematické optimalizace.

Problém matematické optimalizace, jejž můžeme také označovat termínem **optimalizační problém**, zapisujeme ve tvaru

$$\min_x f(x), \quad x \in \mathcal{D}. \tag{12}$$

Množinu $\mathcal{D} \subseteq \mathbb{R}^n$ nazýváme **přípustnou množinou**, prvkům této množiny, $x \in \mathcal{D}$, říkáme **přípustné body** a funkci f nazýváme **účelovou funkcí**, lze rovněž použít termín **cílová funkce**.

3.1 Nepodmíněná optimalizace

Vzhledem k tomu, že je aparát matematické optimalizace v dnešní době již velmi rozsáhlý, rozdělujeme jej na několik specializovaných disciplín podle typu účelové funkce a přípustné množiny. Jedná se například o lineární či kvadratické programování, které řeší problém nalezení minima lineární resp. kvadratické funkce.

Optimalizaci lze také rozlišit podle typu přípustné množiny a to na **podmíněnou** resp. **nepodmíněnou**, jež se zabývají nalezením minima funkce $f : \mathcal{D} \to \mathbb{R}$, přičemž o první zmíněné hovoříme v případě, kdy

$$\mathcal{D} = \{x \in \mathcal{D}(f) \mid c_i(x) \geq 0, \forall i = 1, \ldots, m\} \subset \mathcal{D}(f),$$

kde $m \in \mathbb{N}$, $c_i : \mathcal{D}(c_i) \to \mathbb{R}$, $\mathcal{D}(c_i) \subseteq \mathbb{R}^n$, $\forall i = 1, \ldots, m$.

Je-li $\mathcal{D} = \mathcal{D}(f)$, jedná se o nepodmíněnou optimalizaci. Poněvadž je tato práce zaměřena na Newtonovu metodu pro úlohu nepodmíněné optimalizace, budeme dále uvažovat účelovou funkci $f : \mathcal{D}(f) \to \mathbb{R}$.

O podmíněné optimalizaci se bude hovořit pouze v závěru páté kapitoly ve spojitosti s využitím sebeomezujících funkcí.

3.2 Optimalizační algoritmus

Jedním z přístupů k řešení úloh matematické optimalizace je použití numerických metod. Pod pojmem numerická optimalizační metoda si můžeme představit určitý postup, přesněji řečeno **algoritmus**, jenž má za úkol najít minimum dané účelové funkce. Tyto metody jsou iterační, což znamená, že neřeší úlohu hned zpočátku jako celek, ale postupně krok za krokem směřují k řešení.

Optimalizační algoritmy mají společnou základní myšlenku postupu, kterou je možno popsat následovně:

Algoritmus 3.1.

- Zvolíme **počáteční odhad** hodnoty řešení, bod $x_0 \in \mathcal{D}(f)$.
- Sestavíme posloupnost $\{x_k\}$, $x_k \in \mathcal{D}(f)$, $\forall k \in \mathbb{N}_0$, zpřesňujících se odhadů hledaného řešení, přičemž prvním členem posloupnosti $\{x_k\}$ je bod x_0.

Numerické optimalizační metody rozdělujeme podle typu posloupnosti $\{x_k\}$ na **finitní metody**, pokud se jedná o konečnou posloupnost $\{x_k\}_{k=0}^{m}$, $m \in \mathbb{N}$, a **infinitní metody**, při níž konstruujeme nekonečnou posloupnost $\{x_k\}_{k=0}^{\infty}$.

Newtonova optimalizační metoda je infinitní metodou, budeme tedy dále v textu předpokládat, že hledáme posloupnost $\{x_k\}_{k=0}^{\infty}$ zpřesňujících se odhadů hledaného řešení.

Poznámka 3.1. Na posloupnost $\{x_k\}_{k=0}^{\infty}$ klademe požadavek konvergence číselné posloupnosti $\{f(x_k)\}_{k=0}^{\infty}$ k číslu $f(x^*)$, kde $x^* \in \mathcal{D}(f)$ je bod lokálního minima funkce $f(x)$.

Definice 3.1. Nechť $f(x^*) = \inf_{x \in \mathcal{D}(f)} f(x)$. Posloupnost $\{x_k\}_{k=0}^{\infty}$, pro kterou platí

$$f(x_k) \to f(x^*) \quad \text{pro } k \to \infty \ , \tag{13}$$

se nazývá **minimalizující posloupnost** úlohy (12).

Definice 3.2. Minimalizující posloupnost $\{x_k\}_{k=0}^{\infty}$ definujeme rekurentně vztahem

$$x_{k+1} = x_k + t_k d_k \ , \tag{14}$$

kde $d_k \in \mathbb{R}^n$ nazýváme **směr** k-tého kroku a číslo $t_k \in \mathbb{R}$, $t_k > 0$, **délka** k-tého kroku, neboť je-li $\|d_k\|_2 = 1$, pak je $t_k = x_{k+1} - x_k$.

Algoritmus 3.2. Mějme dánu účelovou funkci f. Obecně můžeme algoritmus numerických metod, souhrnně označovaných jako **metody spádu**, zformulovat takto:

- Zvolíme počáteční odhad $x_0 \in \mathcal{D}(f)$
- **Opakujeme**
  1. Určíme směr k-tého kroku d_k.
  2. Zvolíme délku k-tého kroku $t_k > 0$.
  3. Položíme $x_{k+1} := x_k + t_k d_k$.

  dokud není splněna zastavovací podmínka.

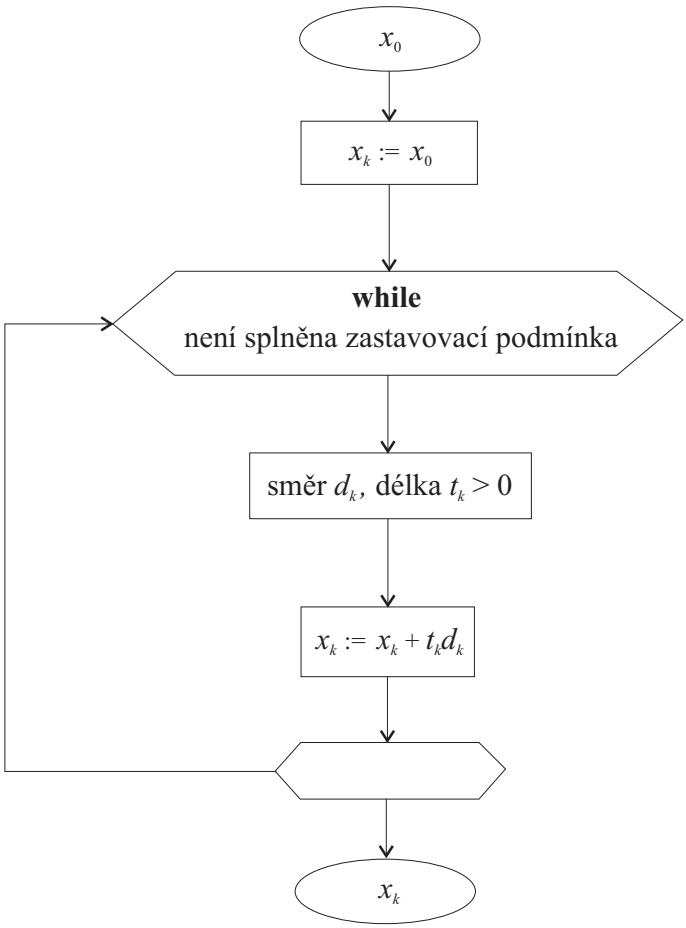
Je potřeba dodat, že zastavovací podmínka je kritérium, podle kterého dokážeme posoudit „kvalitu“ odhadu minima účelové funkce f v k-tém kroku.

Metody, pomocí kterých určují optimalizační algoritmy minimalizující posloupnost, se již různí. Rozdíl mezi jednotlivými numerickými metodami je ve výpočtu směru k-tého kroku d_k a určení délky k-tého kroku t_k. Můžeme je rozdělit do tří skupin podle počtu derivací, jež potřebují k výpočtu k-tého kroku.

- metody nultého řádu – nevyžadují výpočty derivací
  (např.: Metoda simplexů, Hooke-Jeaveseova metoda)

- metody prvního řádu – vyžadují výpočet gradientu
  (např.: Metoda nejrychlejšího spádu, Metoda sdružených gradientů, kvazi-Newtonovské metody)
- metody druhého řádu – vyžadují výpočet gradientu a Hessovy matice
  (např.: Newtonova metoda)

Graficky lze algoritmus znázornit pomocí vývojového diagramu, jenž pro obecnou spádovou metodu vypadá následovně:



3.3 Rychlost konvergence algoritmu

Optimalizační algoritmus, který použijeme při řešení daného problému, vybereme podle jeho efektivity a přesnosti. Efektivitu algoritmu posuzujeme podle

rychlosti, s jakou konverguje posloupnost $\{x_k\}_{k=0}^{\infty}$ k hledanému lokálnímu minimu x^* účelové funkce f.

Definice 3.3. Mějme dánu posloupnost $\{x_k\}_{k=0}^{\infty}$, jež konverguje k bodu x^*. Řekneme, že tato posloupnost konverguje s **rychlostí řádu r** právě tehdy, když existují čísla $k_0 \in \mathbb{N}_0$, $q \in \mathbb{R}_0^+$ taková, že

$$\|x_{k+1} - x^*\|_2 \leq q\|x_k - x^*\|_2^r, \quad \forall k \in \mathbb{N}_0,\ k \geq k_0. \tag{15}$$

Poznámka 3.2. Jestliže $r = 1$, pak hovoříme o **lineární rychlosti konvergence**. Pokud $r = 2$, jedná se o **kvadratickou rychlost konvergence**.

Rychlost konvergence algoritmu můžeme také chápat jako rychlost konvergence číselné posloupnosti $\{f(x_k)\}_{k=1}^{\infty}$ k číslu $f(x^*)$.

Definice 3.4. Mějme dánu účelovou funkci $f(x)$, která má lokální minimum v bodě x^*. Nechť $\{x_k\}_{k=0}^{\infty}$ je posloupnost, jež konverguje k bodu x^*. Řekneme, že tato posloupnost konverguje s **rychlostí řádu r**, jestliže existují čísla $k_0 \in \mathbb{N}_0$, $q \in \mathbb{R}_0^+$ taková, že

$$|f(x_{k+1}) - f(x^*)| \leq q|f(x_k) - f(x^*)|^r, \quad \forall k \in \mathbb{N}_0,\ k \geq k_0. \tag{16}$$

Již z názvu této vlastnosti vyplývá, že čím bude rychlost konvergence vyšší, tím méně iteračních kroků je potřeba k dosažení požadované přesnosti výsledku. Je rovněž zřejmé, že pro různě „obtížné“ účelové funkce bude algoritmus konvergovat různě rychle. Tedy numerickou optimalizační metodu vybíráme i podle typu účelové funkce, na níž ji chceme aplikovat.

4 Newtonova optimalizační metoda

Numerickou metodou optimalizace, jíž se budeme dále zabývat, je Newtonova metoda. Jedná se o infinitní numerickou metodu druhého řádu, jejíž hlavní myšlenka – hledání směru v k-tém kroku – vychází z Taylorovy věty.

V celé této kapitole budeme uvažovat účelovou funkci $f : \mathcal{D}(f) \to \mathbb{R}$, kde $\mathcal{D}(f) \subseteq \mathbb{R}^n$, která je dvakrát spojitě diferencovatelná na svém definičním oboru, což znamená, že zde existuje její gradient a Hessova matice. Dále předpokládejme, že $x^* \in \mathcal{D}(f)$ je bodem lokálního minima účelové funkce, tedy platí $\nabla f(x^*) = o$ a $\mathbf{D}^2 f(x^*)$ je pozitivně definitní. Mějme dáno okolí $\mathcal{U}$ bodu x^*, na kterém je Hessova matice pozitivně definitní, a počáteční odhad $x_0 \in \mathcal{U}$.

4.1 Myšlenka

Jestliže jsou splněny předpoklady věty 1.4 (kapitola 1.4), lze podle ní aproximovat účelovou funkci f v bodě x_0, což znamená

$$f(x_0 + \alpha) = f(x_0) + f'_\alpha(x_0) + \frac{1}{2} f''_{\alpha,\alpha}(x_0) + \frac{1}{3!} f^{(3)}_{\alpha,\alpha,\alpha}(x_0 + \theta\alpha).$$

Definice 4.1. Definujme nyní funkci $\hat{f}$, a to následovně

$$\hat{f}(x_0 + \alpha) = f(x_0) + f'_\alpha(x_0) + \frac{1}{2} f''_{\alpha,\alpha}(x_0), \tag{17}$$

neboli (dosadíme $\alpha = y - x_0$)

$$\hat{f}(y) = f(x_0) + (y - x_0)\nabla f(x_0)^{\mathrm{T}} + \frac{1}{2}(y - x_0)\mathbf{D}^2 f(x_0)(y - x_0)^{\mathrm{T}}. \tag{18}$$

Funkci $\hat{f}$ nazveme **Taylorova aproximace druhého řádu** funkce f v bodě x_0.

Podívejme se nyní blíže na funkci $\hat{f}$. Vzhledem k pozitivní definitnosti matice $\mathbf{D}^2 f(x_0)$ se jedná o konvexní kvadratickou funkci proměnné y. Hledejme její minimum, které je odhadem minima účelové funkce f, poněvadž je-li x_0 blízko x^*, je $\min \hat{f}(x) \approx f(x^*)$.

Zderivujeme vztah (18) a položíme jej roven nulovému vektoru, tedy dostáváme

$$\nabla \hat{f}(y) = \nabla f(x_0) + (y - x_0)\mathbf{D}^2 f(x_0) = o.$$

Odtud pak

$$y - x_0 = -\nabla f(x_0)\left[\mathbf{D}^2 f(x_0)\right]^{-1},$$

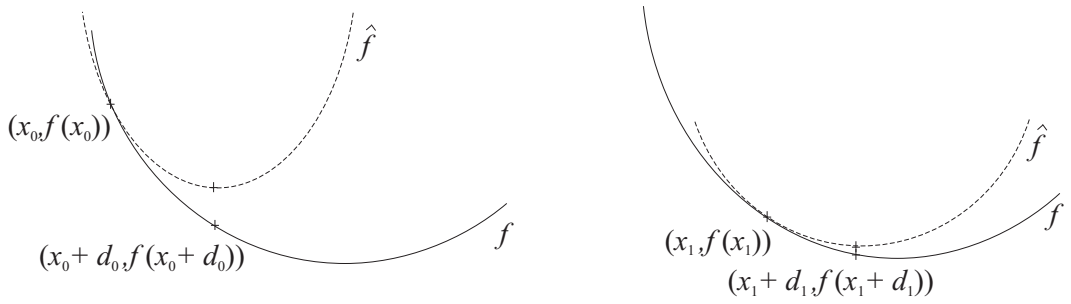
jednoduchou úpravou získáme

$$y = x_0 - \nabla f(x_0)\left[\mathbf{D}^2 f(x_0)\right]^{-1}. \tag{19}$$

Zjistili jsme, že vektor y tvaru (19) je minimum Taylorovy aproximace $\hat{f}$, a tedy hledaným odhadem minima účelové funkce f.

Označme $d_0 = -\nabla f(x_0)[\mathbf{D}^2 f(x_0)]^{-1}$. Vektor d_0 je veličina, kterou je potřeba přidat k bodu x_0, abychom minimalizovali aproximaci druhého řádu účelové funkce f (viz obrázek 4).

Položíme bod $x_1 := y$ a budeme v konstrukci pokračovat dále pro x_1 namísto x_0 (viz obrázek 4). Tímto způsobem sestavíme minimalizující posloupnost $\{x_k\}$.



Obrázek 4: Účelová funkce f a její aproximace druhého řádu $\hat{f}$, první iterační krok – vlevo, druhý iterační krok – vpravo.

Definice 4.2. Nechť je dána účelová funkce f a počáteční odhad x_0. Definujeme **Newtonovu posloupnost** vztahem

$$x_{k+1} = x_k - \nabla f(x_k)\left[\mathbf{D}^2 f(x_k)\right]^{-1}, \quad \forall k \in \mathbb{N}_0. \tag{20}$$

Pro $x_k \in \mathbb{R}^n$, $k \in \mathbb{N}_0$, vektor

$$d_k = -\nabla f(x_k)\left[\mathbf{D}^2 f(x_k)\right]^{-1} \tag{21}$$

nazveme **Newtonův krok** funkce f v bodě x_k.

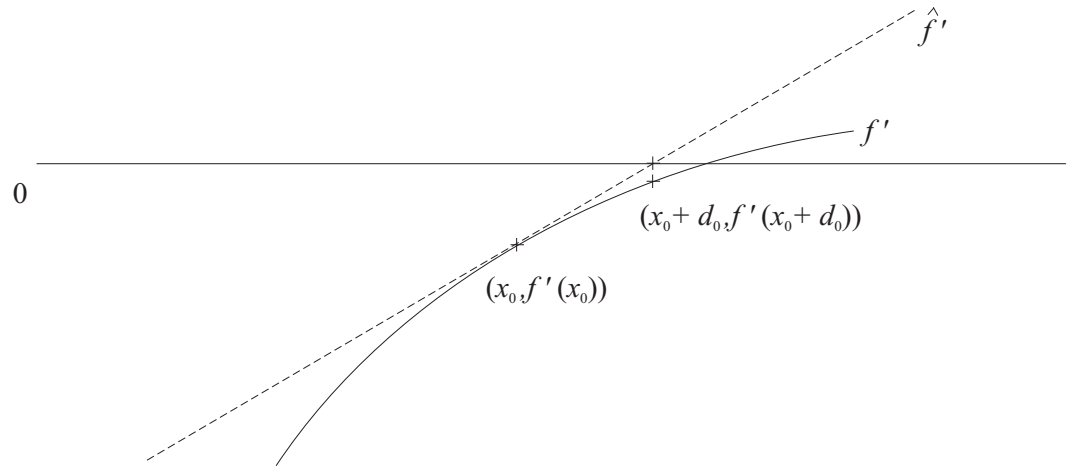
Poznámka 4.1. Newtonův krok je směr spádu účelové funkce, neboť vzhledem k pozitivní definitnosti Hessovy matice je

$$d_k \nabla f(x_k)^{\mathrm{T}} = -\nabla f(x_k)\left[\mathbf{D}^2 f(x_k)\right]^{-1} \nabla f(x_k)^{\mathrm{T}} < 0,$$

z čehož vyplývá, že $x_k \to x^*$, pro $k \to \infty$.

V kapitole 4.4 si ukážeme, že je Newtonova posloupnost $\{x_k\}$ skutečně minimalizující posloupností úlohy (12) konvergující k bodu x^* a také, že matice $\mathbf{D}^2 f(x_k)$ je pozitivně definitní $\forall k \in \mathbb{N}_0$ (viz věta 4.3).

Příklad 4.1. Je zadána účelová funkce $f : \mathbb{R} \to \mathbb{R}$, která je na okolí $\mathcal{U}$ bodu x^* ryze konvexní. Najděte její minimum při volbě počátečního odhadu $x_0 \in \mathcal{U}$.



Obrázek 5: Derivace funkce f a její lineární aproximace.

Řešení: Minimalizační problém je charakterizován rovnicí $f'(x^*) = 0$, kde f' je, vzhledem ke kladné druhé derivaci funkce f, rostoucí. Sestavíme funkci $\hat{f}'$, což je lineární aproximace funkce f' v bodě x_0. Je-li x_0 blízko bodu x^*, pak je nulový bod funkce $\hat{f}'$ velmi dobrou aproximací řešení minimalizačního problému (viz obrázek 5).

Definice 4.3. Nechť $x_k \in \mathbb{R}^n$, veličina

$$\lambda(x_k) = \left[\nabla f(x_k)\left[\mathbf{D}^2 f(x_k)\right]^{-1}\nabla f(x_k)^{\mathrm{T}}\right]^{\frac{1}{2}} \tag{22}$$

se nazývá **Newtonův úbytek** funkce $f(x)$ v bodě x_k.

Pokusíme se nahlédnout, jakou roli $\lambda(x_k)$ v Newtonově metodě hraje. Podívejme se nejprve na jiné způsoby vyjádření Newtonova úbytku. Dosadíme-li Newtonův krok d_k do vztahu (22), pak dostáváme

$$\lambda^2(x_k) = -d_k \nabla f(x_k)^{\mathrm{T}} \tag{23}$$

Využijeme-li symetričnosti Hessovy matice a rovnost (22) upravíme

$$\lambda^2(x_k) = \nabla f(x_k)\left[\mathbf{D}^2 f(x_k)\right]^{-1}\mathbf{D}^2 f(x_k)\left[\mathbf{D}^2 f(x_k)\right]^{-1}\nabla f(x_k)^{\mathrm{T}},$$

$$\lambda^2(x_k) = d_k \mathbf{D}^2 f(x_k) d_k^{\mathrm{T}} = \|d_k\|^2_{\mathbf{D}^2 f(x_k)}. \tag{24}$$

Při analýze vlastností algoritmu musíme vědět, jakým způsobem posoudíme „kvalitu“ odhadu x_k a rychlost konvergence posloupnosti $\{x_k\}_{k=0}^{\infty}$. Poněvadž neznáme x^*, nemůžeme ani určit $\|x^* - x_k\|_2$. K tomuto účelu využijeme rozdíl $f(x_k) - f(x^*)$.

$$f(x_k) - f(x^*) \approx f(x_k) - \min_y \hat{f}(y) = f(x_k) - \hat{f}(x_k + d_k) \tag{25}$$

Pro $k \in \mathbb{N}_0$ můžeme Taylorovu aproximaci druhého řádu funkce f v bodě x_k psát ve tvaru

$$\hat{f}(x_k + d_k) = f(x_k) + d_k \nabla f(x_k)^{\mathrm{T}} + \frac{1}{2} d_k \mathbf{D}^2 f(x_k) d_k^{\mathrm{T}},$$

čehož dále využijeme. Za použití vztahů (23) a (24) dostáváme

$$\hat{f}(x_k + d_k) = f(x_k) - \lambda^2(x_k) + \frac{1}{2}\lambda^2(x_k),$$

a po úpravě platí

$$f(x_k) - \hat{f}(x_k + d_k) = \frac{1}{2}\lambda^2(x_k). \tag{26}$$

Čili $\frac{1}{2}\lambda^2(x_k)$ je odhadem $f(x_k) - f(x^*)$. Tento výraz slouží ke zkoumání rychlosti konvergence algoritmu a hraje úlohu zastavovacího kritéria algoritmu. Zvolenou povolenou odchylku od přesného řešení úlohy, tj. $\varepsilon \in \mathbb{R}^+$, v k-tém kroku porovnáme s $\frac{1}{2}\lambda^2(x_k)$, je-li $\frac{1}{2}\lambda^2(x_k) > \varepsilon$, pokračujeme další iterací. V opačném případě algoritmus zastavíme.

Zajímavou vlastností Newtonovy metody je její afinní invariance.

Věta 4.1. *Newtonův krok d_k a Newtonův úbytek $\lambda(x_k)$ jsou invariantní vzhledem ke změnám souřadnic.*

Důkaz: viz [1], str. 486.

Předchozí věta nám může například pomoci při optimalizaci některých funkcí zadaných implicitně.

Příklad 4.2. Navrhněte postup, jak minimalizovat funkci $y(x)$ danou implicitně vztahem $y = (x + y)\ln(x + y)$.

Řešení: Nejprve zavedeme lineární substituci

$$\begin{aligned} u &= x + y, \\ v &= y. \end{aligned} \tag{27}$$

Pomocí těchto vztahů, jež dosadíme do implicitního zadání funkce y, vyjádříme v jako funkci proměnné u. Získáme tak předpis

$$v = u \ln u.$$

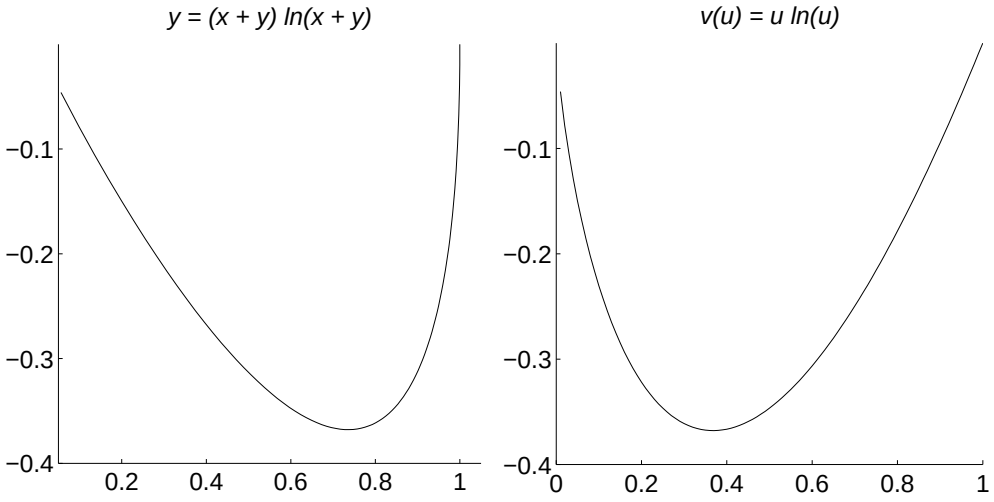
Funkci $v(u)$ budeme minimalizovat Newtonovou metodou (viz kapitola 4.3). Najdeme lokální minimum funkce $v(u)$ a dopočítáme funkční hodnotu v tomto bodě, tj.

$$\begin{aligned} u^* &= 0{,}3679 \\ v^* &= -0{,}3679 \end{aligned}$$

Nakonec využijeme větu 4.1, dvojici u^*, v^* dosadíme do vztahů (27) a vypočteme x^*, y^*. Celkem tedy máme

$$x^* = 0{,}7358$$
$$y^* = -0{,}3679,$$

což je hledané řešení. Funkce $y(x)$ má lokální minimum v bodě $x^* = 0{,}7358$ a nabývá zde hodnoty $y^* = -0{,}3679$.



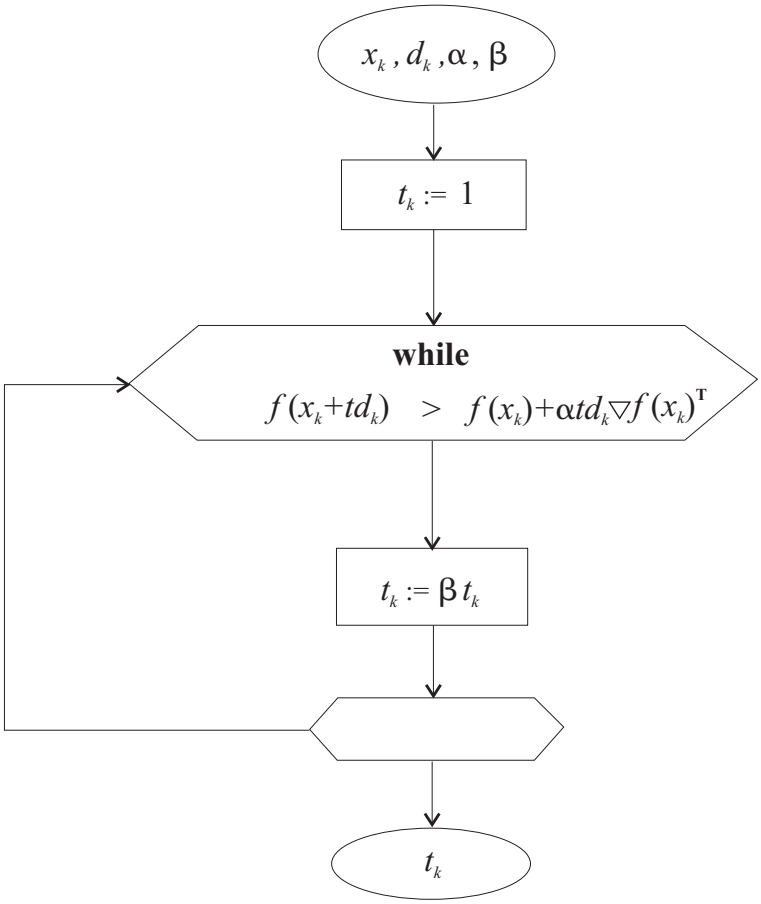
Obrázek 6: Grafy funkcí $y = (x + y)\ln(x + y)$ a $v = u \ln u$.

4.2 Zpětné vyhledávání

Zpětné vyhledávání je jednou z metod, pomocí níž určují spádové metody nepodmíněné optimalizace délku k-tého kroku t_k. Nejedná se ovšem o metodu exaktní, avšak výhodou je její jednoduchost a efektivnost. Její hlavní myšlenkou je minimalizovat účelovou funkci ve směru d_k, tj. řešit úlohu

$$\min_t f(x_k + td_k), \quad t \in \mathbb{R},\ t \geq 0. \tag{28}$$

Algoritmus zpětného vyhledávání znázorněný pomocí vývojového diagramu má tuto strukturu:

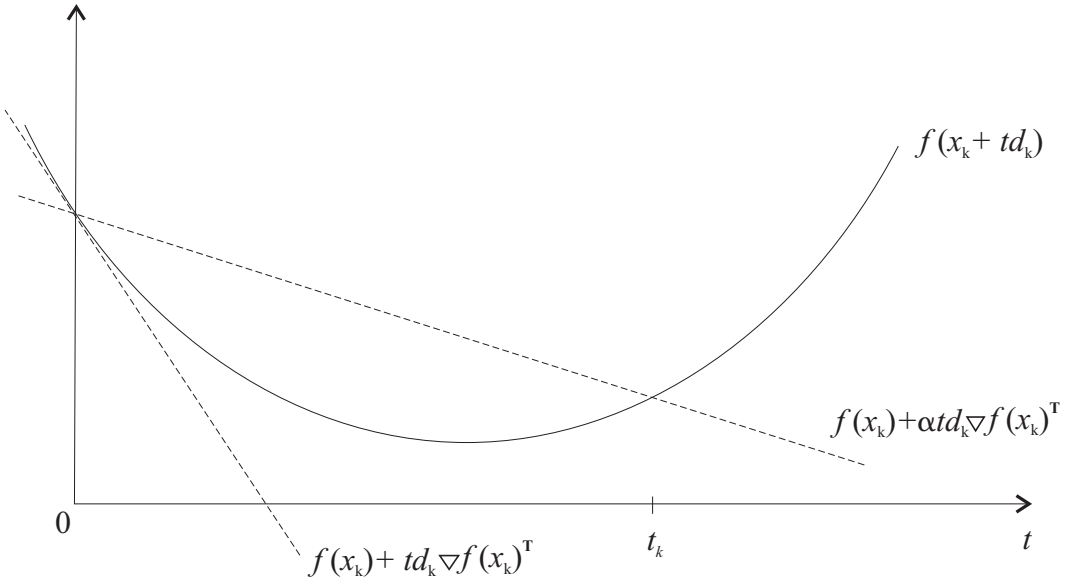


Algoritmus 4.1. Nechť je $\{x_k\}_{k=0}^{\infty}$ minimalizující posloupnost. Algoritmus zpětného vyhledávání délky kroku vypadá následovně:

- Máme dán bod $x_k \in \mathbb{R}^n$ a směr d_k. Dále zvolíme parametry $\alpha \in (0;\ 0{,}5)$ a $\beta \in (0;\ 1)$, na kterých závisí přesnost této metody.
- Položíme $t_k := 1$.
- Dokud je splněno $f(x_k + t_k d_k) > f(x_k) + \alpha t_k d_k \nabla f(x_k)^{\mathrm{T}}$, položíme $t_k := \beta t_k$.

Parametr α zpravidla volíme v rozmezí 0,01 až 0,3; podle toho, jak přesný odhad chceme zjistit, přičemž čím je α menší, tím lepší odhad získáme. Naproti tomu parametr β nám říká, jak hrubý odhad bude, a obvykle jej volíme v rozmezí 0,1 (pro hrubý odhad) až 0,8 (pro jemnější odhad). Doporučené hodnoty parametrů jsou převzaty z [1], str. 466.

Tato metoda se označuje jako zpětné vyhledávání, neboť na počátku zvolíme $t_k = 1$ a poté zpětně zjišťujeme zda $f(x_k + t_k d_k)$ je postačující odhad (28). Není-li tomu tak, zmenšíme krok a opět otestujeme. Tento postup opakujeme tak dlouho, dokud odhad nezískáme.



Obrázek 7: Zpětné vyhledávání kroku t_k. Podmínka metody je splněna na intervalu, kde je graf funkce f pod horní čárkovanou přímkou.

Pro kontrolu či urychlení výpočtu můžeme použít krátký M-soubor, který dále využijeme i pro Newtonovu metodu. Na počátku jsou zvoleny parametry α, β a poté probíhá cyklus, ve kterém se hledá krok t_k. Funkce určí krok t_k, jenž bude splňovat podmínky popsané výše, tj. $t_k \approx \min_t f(x_k + td_k)$, $t \in \mathbb{R}$, $t \geq 0$.

M-soubor 4.1. (Zpětné vyhledávání)

```
function[tk] = backtracking(f,gradf,x,xk,dk)


%backtracking  Zpětné vyhledávání
%   backtracking(F,GRADF,X,XK,DK) hledá optimální délku kroku
%   v bodě XK ve směru DK, kde F je optimalizovaná funkce,
%   GRADF její gradient a X je vektor proměnných.
```

```
%   Tato funkce je potřebná ke správné funkčnosti M-souborů
%   newtonmin a newtonmax, v nichž je použita. Pro samostatné
%   použití by ji bylo možno adekvátně přizpůsobit.

tk = 1; alfa = 0.02; beta = 0.8;

while( subs(f,x,xk + tk .* dk) > subs(f,x,xk) + ...
        alfa .* tk .* dk * transpose(double(subs(gradf,x,xk))) )
    tk = beta * tk;
end
```

Tento M-soubor by bylo rovněž možno přizpůsobit pro samostatné použití. Vstupními parametry by byly symbolická funkce `f`, symbolický vektor proměnných `x`, vektory `xk` a `dk`, a také parametry `alfa` a `beta`.

4.3 Algoritmus

Uvedeme si nyní postup, podle kterého řešíme optimalizační problém, tj. úlohu (12), Newtonovou metodou popsanou výše. Chceme minimalizovat účelovou funkci f, jež splňuje předpoklady této kapitoly, s počátečním odhadem x_0.

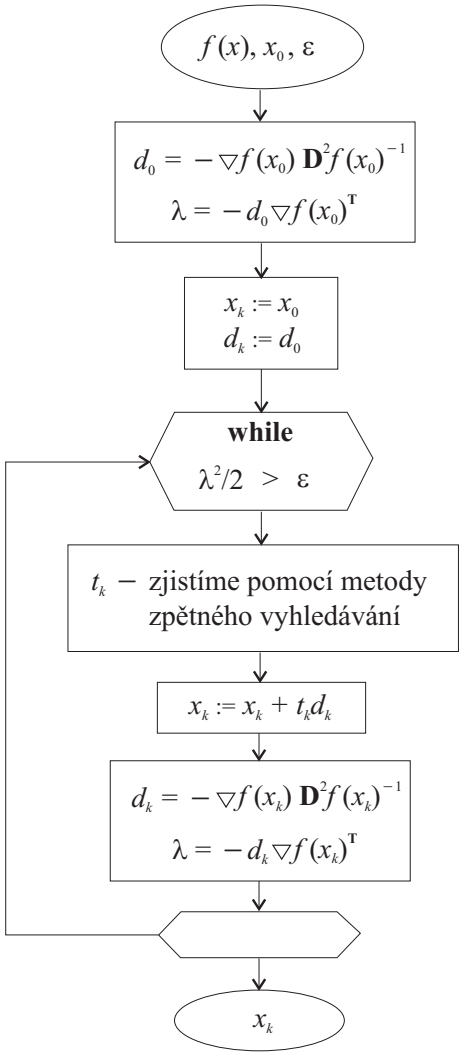
Algoritmus 4.2. Nechť je dána tolerance přesnosti $\varepsilon > 0$ a počáteční odhad $x_0 \in \mathcal{D}(f)$, v němž je Hessova matice účelové funkce pozitivně definitní.
Opakujeme

1. Vypočteme Newtonův krok a úbytek

   $$d_k = -\nabla f(x_k)[\mathbf{D}^2 f(x_k)]^{-1}, \quad \lambda^2(x_k) = -d_k \nabla f(x_k)^{\mathrm{T}}.$$

2. Otestujeme zastavovací kritérium, tj. **skončíme**, jestliže $\frac{1}{2}\lambda^2(x_k) \leq \varepsilon$.
3. Zvolíme délku kroku t_k pomocí algoritmu zpětného vyhledávání.
4. Položíme $x_{k+1} := x_k + t_k d_k$.

Pro lepší názornost se podívejme na vývojový diagram Newtonovy optimalizační metody:



Velmi užitečným k urychlení výpočtů je následující M-soubor sloužící k nalezení minima dané účelové funkce Newtonovou optimalizační metodou. Nejdříve jsou ověřeny vstupní parametry, poté je napočítán gradient účelové funkce a její Hessova matice a následně ověřena pozitivní definitnost této Hessovy matice. Dále funkce postupuje podle algoritmu popsaného výše v této kapitole. Funkce `f` se musí zadat jako symbolická, stejně tak i vektor proměnných `x`.

M-soubor 4.2. (Newtonova metoda – minimalizace)

```
function [xk,f_xk,k] = newtonmin(f,x,x0,epsilon,maxit)

%newtonmin  Newtonova optimalizační metoda
```

```
%   newtonmin(F,X,X0,EPSILON,MAXIT) hledá minimum symbolické
%   funkce F(X) pomocí Newtonovy optimalizační metody
%   vstup: F       - symbolická funkce
%          X       - symbolický vektor proměnných
%          X0      - počáteční odhad
%          EPSILON - tolerance přesnosti, kladné reálné číslo
%          MAXIT   - maximální počet iterací, jenž bude
%                    po spuštění Newtonovy metody proveden
%   výstup: xk   - odhad lokálního minima funkce F(X)
%           f_xk - funkční hodnota v bodě xk
%           k    - počet provedených iterací

%   Funkci lze použít i při hledání maxima funkce, zadá se -F
%   namísto F, hledaná hodnota maxima je pak číslo opačné
%   k číslu f_xk.

% doplnění vstupních argumentů
if (nargin < 4)
    epsilon = 10.^-4;
end
if (nargin < 5)
    maxit = 100;
end

% kontrola vstupních dat
if (epsilon <= 0)
    error('Chyba na vstupu, ...
           EPSILON musí být kladné reálné číslo!');
end
dim = length(x0);
```

```
if (dim ~= length(x))
    error('Chyba na vstupu, ...
           vektory X, X0 nemají stejnou délku!');
end

% výpočet gradientu funkce f
for (i = 1:dim)
    gradf(i) = diff(f,x(i));
end
% výpočet Hessovy matice funkce f
for (i = 1:dim)
    for (j = i:dim)
        Hessf(i,j) = diff(gradf(i),x(j));
        Hessf(j,i) = Hessf(i,j);
    end
end

k = 0; xk = x0; z = 0;

gradf_xk = double(subs(gradf,x,xk));
Hessf_xk = double(subs(Hessf,x,xk));

% kontrola pozitivní definitnosti Hessovy matice
v = eig(Hessf_xk);
for (i = 1:dim)
    if (v(i) <= 0)
        error('V bodě počátečního odhadu není funkce ...
               konvexní, zvolte prosím jiný bod X0!');
    end
end
```

```
dk = - (gradf_xk * inv(Hessf_xk));
lambda = (-dk) * transp(gradf_xk);
lambda0 = lambda;

% iterační cyklus - jádro funkce
while(lambda/2 >= epsilon & k <= maxit)
    if (z == 0)
        tk = backtracking(f,gradf,x,xk,dk);
        if (tk == 1)
            z = 1;
        end
    end
    xk = xk + tk.*dk;
    k = k + 1;
    gradf_xk = double(subs(gradf,x,xk));
    Hessf_xk = double(subs(Hessf,x,xk));
    dk = - (gradf_xk * inv(Hessf_xk));
    lambda = (-dk) * transp(gradf_xk);

    %kontrola, zda metoda nediverguje
    if (isreal(lambda))
        if (lambda > lambda0)
            error('Metoda diverguje, ...
                    zkuste jiný počáteční odhad!');
        end
    else
        lambda = 0;
        xk = xk1;
        display('Metoda diverguje, ...
```

```
                zkuste jiný počáteční odhad!');
    end
end

% výpočet funkční hodnoty v hledaném bodě
f_xk = subs(f,x,xk);
```

Jednoduchou úpravou tohoto M-souboru můžeme sestavit M-soubor sloužící k nalezení maxima dané účelové funkce pomocí Newtonovy optimalizační metody.

M-soubor 4.3. (Newtonova metoda – maximalizace)

```
function [xk,f_xk,k] = newtonmax(f,x,x0,epsilon,maxit)

%newtonmax  Newtonova optimalizační metoda
%   newtonmax(F,X,X0,EPSILON,MAXIT) hledá maximum symbolické
%   funkce F(X) pomocí Newtonovy optimalizační metody
%   vstup: F       - symbolická funkce
%          X       - symbolický vektor proměnných
%          X0      - počáteční odhad
%          EPSILON - tolerance přesnosti, kladné reálné číslo
%          MAXIT   - maximální počet iterací, jenž bude
%                    po spuštění Newtonovy metody proveden
%   výstup: xk   - odhad lokálního maxima funkce F(X)
%           f_xk - funkční hodnota v bodě xk
%           k    - počet provedených iterací

% doplnění vstupních argumentů
if (nargin < 4)
    epsilon = 10.^-4;
end
if (nargin < 5)
```

```
    maxit = 100;
end


f = -f;
[xk,f_xk,k] = newtonmin(f,x,x0,epsilon,maxit);
f_xk = -f_xk;
```

Použití těchto M-souborů může vypadat například následovně. Aplikujeme funkci `newtonmin` na příklad 2.2, v němž jsme použili klasickou optimalizační metodu. Jedná se o funkci $f(x_1, x_2) = (x_1 + x_2)^2 - x_1(x_2 - 3)$. Zvolme počáteční odhad například $[5, -3]$ a toleranci $\varepsilon = 0{,}0001$. Maximální počet iterací nemusíme nastavovat, neboť je jeho hodnota přednastavena na 100 iterací, což nám v tomto případě vyhovuje. Do Matlabu postupně napíšeme:

```
x = sym('[x1,x2]');
fce = (x(1) + x(2))^2 - x(1)*(x(2) - 3);
[reseni,funkcni_hodnota,pocet_kroku] = ...
             newtonmin(fce,x,[5,-3],0.0001);
```

Matlab nám vrátí:

```
reseni =
    -2.0000    1.0000
funkcni_hodnota =       pocet_kroku =
    -3                        1
```

Vidíme, že již po prvním iteračním kroku jsme dospěli ke stejnému výsledku jako při použití klasické optimalizace, což svědčí o rychlé konvergenci Newtonovy metody.

4.3.1 Praktické možnosti algoritmu

Uvedením M-filu `newtonmin` do praxe posléze dojdeme k závěru, že program funguje velmi rychle pro optimalizační problémy řešené v dimenzích menších

než deset (tj. pro funkce $f : \mathcal{D}(f) \to \mathbb{R}$, kde $\mathcal{D}(f) \subseteq \mathbb{R}^k$, $k \in \mathbb{N}$, $k \leq 10$). Chceme-li tento M-file využít i ve vyšších dimenzích, pak srovnatelně rychle pracuje i při aplikaci na funkce z určité třídy. Jedná se o funkce, jež mají takzvaně **řídkou** Hessovu matici, což znamená, že matice obsahuje velké množství nulových prvků. Do této třídy patří například funkce $f(x) = x_1^4 x_2^2 + \sum_{i=3}^{100} x_i^2$, jejíž Hessova matice je tvaru

$$\mathbf{D}^2 f(x) = \begin{pmatrix} 12x_1^2x_2^2 & 8x_1^3x_2 & 0 \cdots 0 \\ 8x_1^3x_2 & 2x_1^4 & 0 \cdots 0 \\ 0 & 0 & 2 \cdots 0 \\ \vdots & \vdots & \vdots \ddots \vdots \\ 0 & 0 & 0 \cdots 2 \end{pmatrix} \in \mathbb{R}^{100\times 100}.$$

Naproti tomu existují funkce, jejichž Hessovy matice mají všechny své prvky nenulové. Pro tyto funkce M-file `newtonmin` již běží pomaleji.

Nabízí se několik možností, jak tento nedostatek opravit. Jednou z nich je využití **diagonální aproximace** Hessovy matice. Namísto výpočtu úplné Hessovy matice vypočteme pouze ty její prvky, které leží na hlavní diagonále, ostatní položíme rovny nule. Je zřejmé, že je tento proces mnohem rychlejší než výpočet všech prvků Hessovy matice.

S takto vzniklou diagonální maticí dále při optimalizaci pracujeme stejně jako s Hessovou maticí. Nový M-file se od původního bude lišit pouze ve výpočtu Hessovy matice účelové funkce.

M-soubor 4.4. (Newtonova metoda – aproximace Hessovy matice)

```
function [xk,f_xk,k] = newtaproxmin(f,x,x0,epsilon,maxit)
...
% výpočet Hessovy matice funkce f
for (i = 1:dim)
    Hessf(i,i) = diff(gradf(i),x(i));
end
...
```

Výhodou tohoto postupu je jeho vysoká rychlost při aplikaci na všechny funkce. Na druhé straně ovšem takto upravená Newtonova metoda dosahuje menší přesnosti jednotlivých iterací, a tedy je potřeba více kroků k dosažení stejné přesnosti odhadu.

Druhou možností, jak se vypořádat s pomalým chodem M-filu `newtonmin` při práci ve vysokých dimenzích, je změna charakteru vstupních parametrů této funkce. Jde o to, že dosazování do symbolické Hessovy matice, vzniklé ze symbolické účelové funkce, je v Matlabu časově velmi náročné. Program se díky této skutečnosti výrazně zpomalí při práci s maticemi řádů dvacet či vyšších, neboť musí v každém kroku dosadit do čtyř set či více výrazů.

Změnu provedeme v několika krocích. Derivaci symbolických výrazů funkcí `diff` nahradíme numerickou derivací, čímž se zbavíme náročného dosazování.

1. **Definice účelové funkce**

   Sestavíme M-file `mojefce`, který nám poslouží k definici účelové funkce. Chceme-li například minimalizovat funkci $f(x) = \sin x$, pak tento M-file bude ve tvaru:

   ```
   function [y] = mojefce(x)
   y = sin(x);
   ```

2. **Výpočet gradientu a Hessovy matice**

   Naprogramujeme funkci `dfmojefce`, jež nám numericky vypočte gradient a Hessovu matici účelové funkce v bodě x.

   ```
   function [dfx,ddfx] = dfmojefce(x)

   % dfx  - gradient dané účelové funkce
   % ddfx - Hessova matice dané účelové funkce

   h = 10^-4;
   ```

```
% výpočet gradientu a diagonálních prvků Hessovy matice
for(i = 1:length(x))
    y1 = x; y3 = x;
    y1(i) = y1(i) - h;
    y3(i) = y3(i) + h;
    dfx(i) = (mojefce(y3) - mojefce(y1)) ./ (2.*h);
    ddfx(i,i) = (mojefce(y3) - 2.*mojefce(x) + ...
                 mojefce(y1)) ./ (h.^2);
end

% výpočet smíšených druhých derivací (tedy zbývajících
% prvků Hessovy matice)
for(i = 1:length(x))
    for(j = 1:length(x))
        if(i > j)
            y1 = x; y2 = x; y3 = x;
            y1(i) = y1(i) + h;
            y2(i) = y1(j) + h;
            y2(j) = y3(i) + h;
            y3(j) = y3(j) + h;
            ddfx(i,j) = (mojefce(x) - mojefce(y1) + ...
                   mojefce(y2) - mojefce(y3)) ./ (h.^2);
            ddfx(j,i) = ddfx(i,j);
        end
    end
end
```

3. **Zpětné vyhledávání**

   Musíme si adekvátně přizpůsobit M-soubor `backtracking`.

```
function[tk] = novezpetne(gradf_xk,xk,dk)

%novezpetne  Zpětné vyhledávání
%   novezpetne(GRADF_XK,XK,DK) hledá optimální délku
%   kroku v bodě XK ve směru DK, kde GRADF_XK je
%   gradient účelové funkce v bodě XK.

tk = 1; alfa = 0.02; beta = 0.8;

while( mojefce(xk + tk .* dk) > mojefce(xk)  + ...
        alfa .* tk .* dk * transpose(gradf_xk))
        tk = beta * tk;
end
```

4. **Optimalizační M-file**

   Nakonec si přizpůsobíme i funkci `newtonmin`.

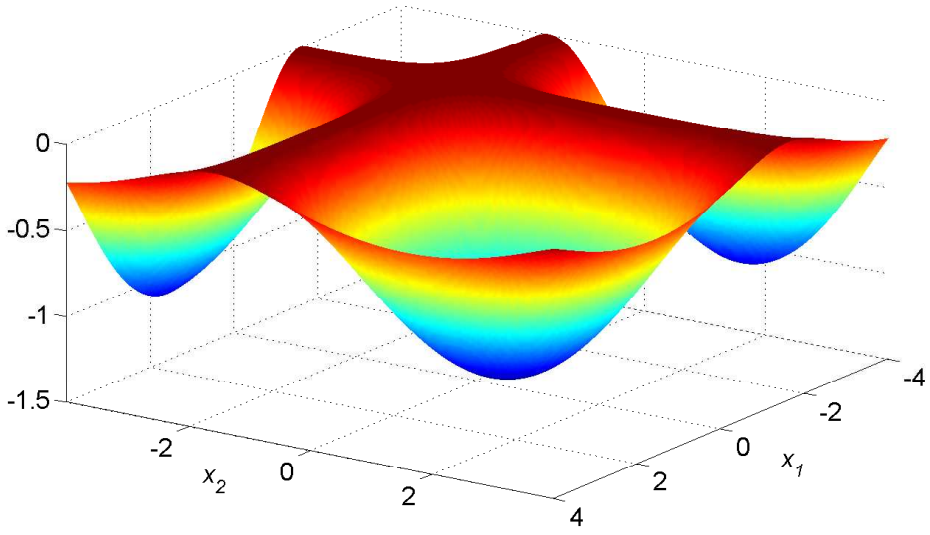
```
function [xk,f_xk,k] = numernewtonmin(x0,epsilon,maxit)
...
k = 0; xk = x0; z = 0;
[gradf_xk,Hessf_xk] = dfmojefce(xk);
...
% iterační cyklus - jádro funkce
while(lambda/2 >= epsilon & k <= maxit)
    if (z == 0)
        tk = novezpetne(gradf_xk,xk,dk);
...
    [gradf_xk,Hessf_xk] = dfmojefce(xk);
    dk = - (gradf_xk * inv(Hessf_xk));
    lambda = (-dk) * transpose(gradf_xk);
end
```

```
...
% výpočet funkční hodnoty v hledaném bodě
f_xk = mojefce(xk);
```

Nevýhodou této metody je náročná příprava před jejím prvním spuštěním, po němž již pouze musíme zadávat účelovou funkci do M-filu `mojefce`. Rovněž konvergence metody je v tomto případě nejistá a mnohdy bude potřeba vyzkoušet větší množství počátečních odhadů.

Je samozřejmé, že veškeré úvahy a úpravy M-souborů, které jsme provedli za účelem urychlit běh programu pro minimalizaci účelových funkcí, lze aplikovat i při řešení úlohy, jež se týká hledání maxima účelové funkce.

Příklad 4.3. Nechť $f(x) = -\prod_{k=1}^{n} [\ln(2 + \sin x_k)]$ je funkce definovaná na $\mathbb{R}^n$, přičemž $x = (x_1, x_2, \ldots, x_n) \in \mathbb{R}^n$. Pomocí M-souborů `newtonmin`, `newtaproxmin` a `numernewtonmin` najděte minimum funkce $f(x)$, pro a) $n = 2$, b) $n = 15$, c) $n = 25$. Odhadněte výhody a nevýhody jednotlivých M-souborů.



Obrázek 8: Graf funkce $f(x) = -\prod_{k=1}^{2} [\ln(2 + \sin x_k)]$.

Řešení: Je zřejmé, že funkce f má pro libovolné $n \in \mathbb{N}$ lokální minimum v bodě $x^* = [\frac{\pi}{2}, \ldots, \frac{\pi}{2}] \in \mathbb{R}^n$. Uvidíme, jak si s tímto problémem poradí jednotlivé M-soubory.

V Matlabu spustíme postupně všechny tři M-soubory. Za počáteční odhady volíme například vektor x_0 samých jedniček, pokud by metoda divergovala, zvolíme lepší počáteční odhad. Tolerance ε nechť má hodnotu 10^{-10}. Výsledky zapíšeme přehledně do tabulky. V jednotlivých řádcích je nejprve znázorněn případ pro $n = 2$, poté pro $n = 15$ a $n = 25$.

Tabulka 1

Název funkce	$\|x_k - x^*\|_2$	Funkční hodnota v bodě x_k	n
newtonmin	$1{,}0260 \cdot 10^{-12}$	$-1{,}206948960812$	2
	$6{,}2031 \cdot 10^{-6}$	$-4{,}098894955135$	15
	$1{,}1301 \cdot 10^{-7}$	$-10{,}498114718439$	25
newtaproxmin	0	$-1{,}206948960812$	2
	0	$-4{,}098894955159$	15
	0	$-10{,}498114718439$	25
numernewtonmin	$1{,}2080 \cdot 10^{-11}$	$-1{,}206948960812$	2
	$9{,}6562 \cdot 10^{-9}$	$-4{,}098894955159$	15
	$4{,}3734 \cdot 10^{-9}$	$-10{,}498114718439$	25

Tabulka 2

Název funkce	Počet iteračních kroků	Čas (s)	n
newtonmin	3	0,187	2
	3	18,078	15
	3	511,594	25
newtaproxmin	2	0,156	2
	2	1,094	15
	2	2,625	25
numernewtonmin	3	0,015	2
	4	0,125	15
	4	0,250	25

Zajímavým poznatkem v tomto případě jistě je, že použití aproximací nemusí vždy vést k méně přesným výsledkům.

Vyzkoušíme-li si tyto tři M-fily i na jiných příkladech a ohodnotíme-li kvalitu některých jejich vlastností, můžeme dojít k závěru, jež je popsán v následující tabulce.

Tabulka 3

Název funkce	Rychlost algoritmu	Přesnost	Rychlost konvergence
newtonmin	nízká	vysoká	vysoká
newtaproxmin	střední	střední	vysoká
numernewtonmin	vysoká	střední	vysoká

Shrnutím poznatků této kapitoly docházíme k závěru, že pro „malé“ optimalizační problémy (tj. problémy v prostorech s malou dimenzí) doporučíme použití funkce `newtonmin`, ovšem pro „větší“ problémy využijeme M-file `numernewtonmin` nebo `newtaproxmin`.

Ve speciálním případě, je-li funkce podobného tvaru jako v našem příkladě (tj. $f(x) = \prod_{k=1}^{n} g(x_k)$, kde $g : \mathcal{D}(g) \to \mathbb{R}$ je funkce jedné reálné proměnné, $n \in \mathbb{N}$), je velmi výhodné použít právě funkci s aproximací Hessovy matice `newtaproxmin`.

Pro ilustraci si nyní uvedeme dva příklady, na nichž si ukážeme použití Newtonovy optimalizační metody.

Příklad 4.4. Z okna ve výšce $h_0 = 12$ metrů byl vyhozen fotbalový míč rychlostí $v = 8$ metrů za sekundu. Určete úhel α, pod kterým je potřeba míč vyhodit, aby doletěl co nejdále. Odpor vzduchu zanedbejte.

Řešení: Označme $\mathcal{K}$ křivku, po které se míč pohybuje. Jedná se o parabolu, která protíná osu x ve dvou bodech. Jedním z nich je bod $[d, 0]$, $d > 0$, což je místo dopadu míče. Budeme se snažit najít vztah mezi délkou hodu a velikostí úhlu.

Nejdříve zjistíme parametrické vyjádření křivky $\mathcal{K}$ v závislosti na čase t. Vzhledem k tomu, že platí zákon superpozice (skládání) pohybů, můžeme říci, že pohyb vykonávaný míčem je složen z vrhu svislého vzhůru, ve směru osy y, a pohybu rovnoměrného přímočarého, ve směru osy x. Pro dráhu rovnoměrného přímočarého pohybu platí

$$x = v_x t.$$

Dráhu pohybu ve směru osy y vypočteme ze vztahu

$$y = h_0 + v_y t - \frac{1}{2} g t^2,$$

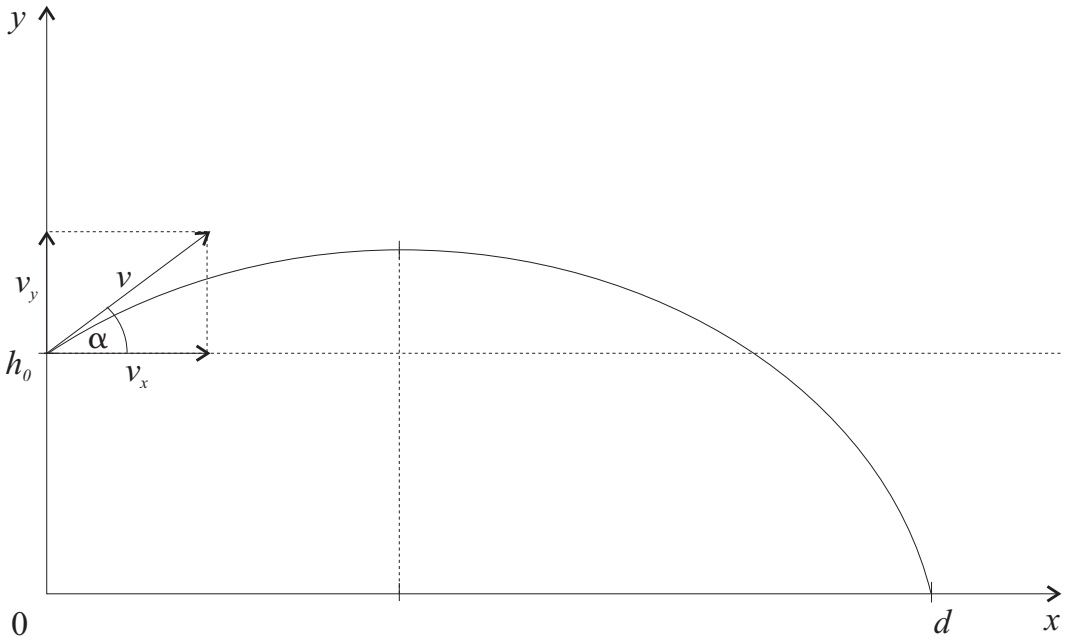
kde $g = 9{,}81\ \mathrm{m/s^2}$ je gravitační zrychlení. Veličiny v_x, v_y jsou složky rychlosti, přičemž je zřejmé (viz obrázek 9), že platí

$$\begin{aligned} v_x &= v\cos\alpha \\ v_y &= v\sin\alpha, \end{aligned}$$

z čehož dostáváme parametrické rovnice paraboly $\mathcal{K}$ ve tvaru

$$x = vt\cos\alpha \tag{29}$$

$$y = h_0 + vt\sin\alpha - \frac{1}{2}gt^2. \tag{30}$$



Obrázek 9: Míč letící z bodu $[0, h_0]$ po křivce do bodu $[d, 0]$.

Ze vztahu (29) vyjádříme t, tj.

$$t = \frac{x}{v\cos\alpha}.$$

Jelikož v okamžiku dopadu míče na zem je jeho y-ová souřadnice nulová, položíme $y = 0$. Dosazením za t do rovnice (30) získáme

$$h_0 + \frac{x\sin\alpha}{\cos\alpha} - \frac{1}{2}g\Big(\frac{x}{v\cos\alpha}\Big)^2 = 0. \tag{31}$$

Nyní chceme vyjádřit proměnnou x. Po úpravě vztahu (31) dostáváme kvadratickou funkci

$$gx^2 - v^2 \sin(2\alpha) \cdot x - 2h_0 v^2 \cos^2 \alpha = 0,$$

kterou známým způsobem vyřešíme

$$x(\alpha) = \frac{v^2 \sin 2\alpha \pm 2v \cos \alpha \sqrt{v^2 \sin^2 \alpha + 2gh_0}}{2g}.$$

Uvažujeme však pouze kladný kořen, tj.

$$x(\alpha) = \frac{v^2 \sin 2\alpha + 2v \cos \alpha \sqrt{v^2 \sin^2 \alpha + 2gh_0}}{2g}, \tag{32}$$

neboť délka je vždy nezáporná.

Funkce $x(\alpha)$ je naší hledanou funkcí, kterou chceme maximalizovat. Budeme tedy zjišťovat, pro které α je funkční hodnota $x(\alpha)$ maximální.

Podívejme se nejdříve, jak bude tato situace vypadat pro výšku $h_0 = 0$. Člen $2gh_0$ ve vztahu (32) pod odmocninou se vynuluje a funkce $x(\alpha)$ se po úpravě zredukuje.

$$x(\alpha) = \frac{v^2 \sin 2\alpha}{g} \tag{33}$$

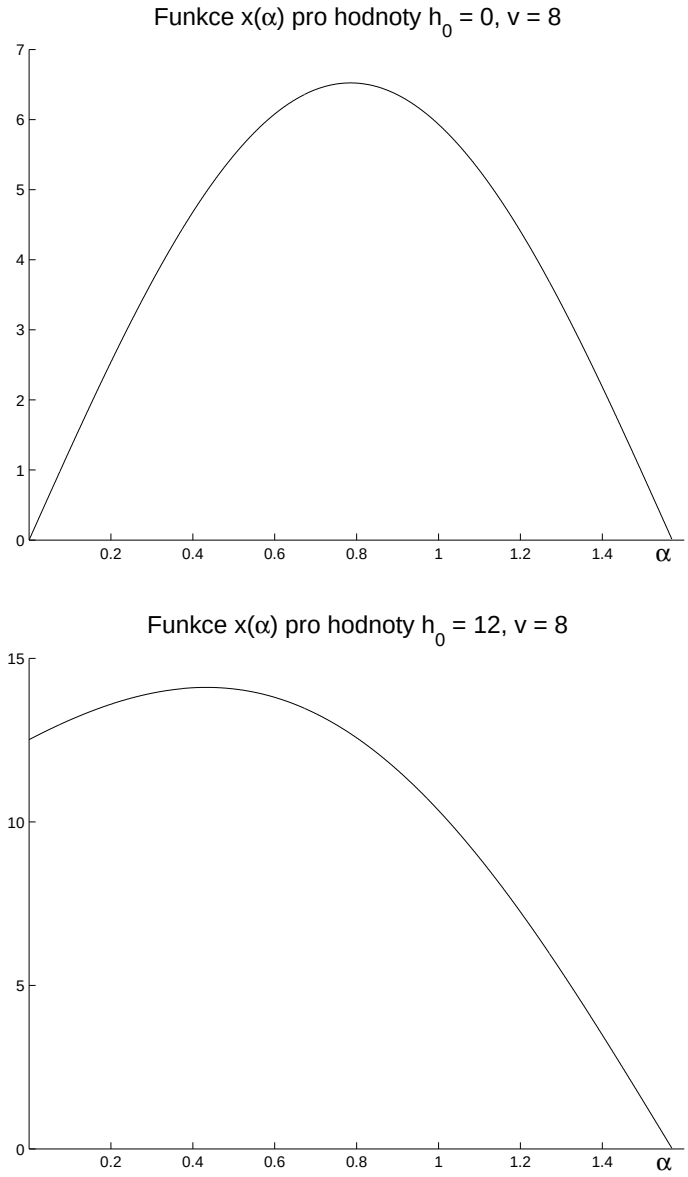
Vzhledem k zadání tohoto příkladu postačí, když budeme uvažovat pouze velikosti úhlu α z intervalu $\langle 0, \pi \rangle$. Funkce sinus nabývá své maximální hodnoty v bodě $\frac{\pi}{2}$, položíme tedy $2\alpha = \frac{\pi}{2}$, z čehož po jednoduché úpravě získáváme řešení

$$\alpha = \frac{\pi}{4} = 45°.$$

Úhel, pod kterým je potřeba míč vyhodit má velikost 45°, délku hodu pak udává vztah

$$d = \frac{v^2}{g}.$$

Výsledek, jejž jsme získali pro $h_0 = 0$ použijeme v obecném případě v roli počátečního odhadu maxima účelové funkce $x(\alpha)$.



Obrázek 10: Funkce $x(\alpha)$ pro různé hodnoty výšky h_0 a rychlosti v.

Chceme maximalizovat funkci $x(\alpha)$ danou vztahem (32). Za tímto účelem bychom mohli použít klasické metody optimalizace, avšak v tomto případě je patrné, že analytické řešení rovnice $x'(\alpha) = 0$ by bylo velmi pracné. Využijeme tedy Newtonovu optimalizační metodu.

Nejprve dosadíme do vztahu (32) konkrétní hodnoty ze zadání, tj. $h_0 = 12$,

$v = 8$ a $g = 9{,}81$, čímž dostáváme

$$x(\alpha) = \frac{8^2 \sin 2\alpha + 2 \cdot 8 \cos\alpha \sqrt{8^2 \sin^2 \alpha + 2 \cdot 9{,}81 \cdot 12}}{2 \cdot 9{,}81}.$$

Po úpravě pak získáme

$$x(\alpha) = \frac{3200}{981} \sin 2\alpha + \frac{800}{981} \cos\alpha \sqrt{64 \sin^2 \alpha + 235{,}44}.$$

Zvolíme nyní počáteční odhad $\alpha_0 = \frac{\pi}{4}$ a toleranci přesnosti $\varepsilon = 0{,}0001$. Dále budeme postupovat podle algoritmu Newtonovy metody (viz kapitola 4.3). K výpočtu použijeme funkci naprogramovanou v Matlabu (viz algoritmus 4.3).

V příkazovém okně Matlabu postupně zadáme:

```
a = sym('a');
f = 3200/981*sin(2*a) + 800/981*cos(a)*sqrt(64*(sin(a))^2+235.44);
[reseni,funkcni_hodnota,pocet_kroku] = newtonmax(f,a,pi/4,0.0001)
```

Matlab nám na to odpoví:

```
reseni =
    0.4332
funkcni_hodnota =       pocet_kroku =
   14.1116                    2
```

Tímto jsme zjistili velikost hledaného optimálního úhlu α^* s tolerancí ε, řešením je $\alpha^* \approx 0{,}4332$ rad, což odpovídá úhlu $24°49'14''$. Algoritmus k tomuto výsledku došel po dvou iteračních krocích, což potvrzuje vysokou rychlost konvergence.

Pokud bude míč vyhozen z okna pod úhlem $\alpha = 24°49'14''$, bude se jednat o nejdelší možný hod při dané rychlosti v a výšce h_0, jehož délka bude 14,1116 metrů.

Příklad 4.5. (Newtonova metoda pro kvadratickou funkci) Použijte Newtonovu metodu při hledání minima kvadratické funkce $f(x) = \frac{1}{2} x \mathbf{A} x^{\mathrm{T}} + b x^{\mathrm{T}} + c$, matice $\mathbf{A}$ je pozitivně definitní. Zvolte libovolný odhad $x_0 \in \mathbb{R}^n$.

Řešení: Předem víme, že minimum kvadratické funkce f existuje, neboť je Hessova matice $\mathbf{D}^2 f(x) = \mathbf{A}$, $\forall x \in \mathbb{R}^n$, pozitivně definitní. Budeme-li řešit tento příklad klasickou optimalizační metodou, tj. funkci f zderivujeme a položíme rovnu nulovému vektoru, čímž dostaneme

$$\nabla f(x) = x\mathbf{A} + b = o,$$

zjistíme, že

$$x^* = -b\mathbf{A}^{-1},$$

kde inverzní matice k matici $\mathbf{A}$ jistě existuje, poněvadž je matice $\mathbf{A}$ pozitivně definitní, a tudíž regulární.

Použijeme-li Newtonovu optimalizační metodu, zjistíme, že již po prvním kroku dosáhneme minima x^*.

1. Vyjádříme gradient a Hessovu matici funkce f v bodě x_0

$$\nabla f(x_0) = x_0\mathbf{A} + b, \quad \mathbf{D}^2 f(x_0) = \mathbf{A}.$$

2. Vypočteme Newtonův krok

$$d_k = -(x_0\mathbf{A} + b)\mathbf{A}^{-1}.$$

3. Určíme x_1, jednoduchými úpravami pak postupně získáme

$$\begin{aligned} x_1 &= x_0 - (x_0\mathbf{A} + b)\mathbf{A}^{-1} \\ x_1 &= x_0 - x_0 - b\mathbf{A}^{-1} \\ x_1 &= -b\mathbf{A}^{-1}. \end{aligned}$$

Z posledního vztahu vidíme, že hledané $x^* = x_1$.

Významnou vlastností Newtonovy metody, kterou jsme díky tomuto příkladu odhalili, je její vysoká efektivita při aplikaci na kvadratickou funkci či funkci jí svým průběhem podobnou.

4.4 Konvergence

Velmi významnou vlastností, dalo by se říci tou nejdůležitější, každé iterační metody je její konvergence k přesnému řešení úlohy. Rozhodně je nanejvýš podstatné vědět, jaké podmínky musí být splněny, aby minimalizující posloupnost konvergovala k přesnému řešení.

K posouzení efektivity metody dále potřebujeme znát rychlost konvergence minimalizující posloupnosti, nejlépe pak počet iteračních kroků potřebných k dosažení odhadu s požadovanou přesností.

Předpokládejme, že je dána účelová funkce f, jež je dvakrát spojitě diferencovatelná a ryze konvexní na okolí $\mathcal{U}$ bodu x^*. Tedy víme, že Hessova matice $\mathbf{D}^2 f(x)$ funkce f je pozitivně definitní a tudíž existují konstanty m, $M \in \mathbb{R}^+$ takové, že platí $m\mathbf{E} \preceq \mathbf{D}^2 f(x) \preceq M\mathbf{E}$, $\forall x \in \mathcal{U}$. Dále předpokládejme, že Hessova matice funkce f splňuje Lipschitzovu podmínku s konstantou $L \in \mathbb{R}_0^+$, tj.

$$\left\|\mathbf{D}^2 f(x) - \mathbf{D}^2 f(y)\right\|_2 \leq L\|x - y\|_2, \quad \forall x, y \in \mathcal{U}. \tag{34}$$

Máme tedy k dispozici tři reálné konstanty, tj. m, M, L, pomocí nichž budeme konvergenci Newtonovy minimalizující posloupnosti $\{x_k\}$ zkoumat.

Poznámka 4.2. Konstanta L vyjadřuje kvalitu aproximace funkce f pomocí Taylorovy aproximace druhého řádu, kterou jsme použili při konstrukci minimalizující posloupnosti (viz definice 4.1). Je-li účelová funkce f kvadratickou funkcí, pak Newtonova posloupnost dosáhne přesného řešení po prvním kroku (viz příklad 4.5). Zároveň je rozdíl $\mathbf{D}^2 f(x) - \mathbf{D}^2 f(y)$ roven nule, pro všechna x, $y \in \mathcal{U}$, a tedy $L = 0$. Z tohoto můžeme již nyní odhadnout, že čím je konstanta L menší, tím méně iteračních kroků Newtonovy metody bude zapotřebí.

Newtonova metoda konverguje ve dvou fázích, jedná se o **tlumenou** a **kvadratickou** fázi konvergence. Tyto dvě etapy, jak si později ukážeme, se od sebe liší zejména rychlostí konvergence Newtonovy posloupnosti a výběrem délky kroku t_k. Rovněž si prokážeme existenci čísla $\eta \in \mathbb{R}$ takového, že $0 < \eta \leq \frac{m^2}{L}$, podle kterého je možné rozhodnout, ve které fázi konvergence se právě nacházíme.

4.4.1 Tlumená fáze konvergence

Etapa konvergence Newtonovy metody, ve které algoritmus zpětného vyhledávání může zvolit délku kroku $t_k < 1$, se označuje jako tlumená fáze. Rychlost konvergence může být totiž „utlumena“ právě zkrácenou délkou k-tého kroku. Newtonova posloupnost konverguje pomaleji nežli v kvadratické fázi, avšak i přesto je zde konvergence velmi rychlá.

Věta 4.2. *Jestliže je v k-tém kroku Newtonovy metody splněno* $\|\nabla f(x_k)\|_2 \geq \eta$, *kde* $\eta \in \mathbb{R}$, $0 < \eta \leq \frac{m^2}{L}$, *potom se Newtonova metoda nachází v tlumené fázi konvergence a platí vztah*

$$f(x_k) - f(x_{k+1}) \geq \alpha\beta\eta^2 \frac{m}{M^2}, \tag{35}$$

přičemž koeficienty α, β *jsou parametry, jež volíme při hledání délky k-tého kroku.*

Důkaz: viz [1], str. 489.

Z této věty vyplývá konvergence posloupnosti $\{f(x_k)\}$ k hodnotě $f(x^*)$, neboť je $f(x_k) > f(x_{k+1}) > f(x^*)$, $\forall k \in \mathbb{N}_0$, a výraz $\alpha\beta\eta^2\frac{m}{M^2}$ je pevně dané reálné číslo, které určuje minimální rozdíl mezi hodnotami $f(x_k)$ a $f(x_{k+1})$. Přímým důsledkem je rovněž možnost odhadnout počet iterací Newtonovy metody v tlumené fázi.

Důsledek 4.1. Počet iterací Newtonovy metody v tlumené fázi je shora omezen výrazem

$$\frac{M^2}{\alpha\beta\eta^2 m}(f(x_0) - f(x^*)). \tag{36}$$

4.4.2 Kvadratická fáze konvergence

Druhou etapou konvergence Newtonovy metody je kvadratická fáze. Tato někdy bývá rovněž označována jako „čistá“ či „ryzí“ fáze konvergence, neboť algoritmus zpětného vyhledávání vždy určí délku kroku $t_k = 1$. Jak již název etapy

napovídá, bude v této fázi Newtonova posloupnost konvergovat k řešení kvadratickou rychlostí.

Věta 4.3. *Předpokládejme, že v k-tém kroku Newtonovy metody je splněno* $\|\nabla f(x_k)\|_2 < \eta$, *kde* $\eta \in \mathbb{R}$, $0 < \eta \leq \frac{m^2}{L}$, *pak platí následující tvrzení.*

1. *Posloupnost* $\{x_k\}_{k=0}^{\infty}$ *konverguje k bodu* x^* *kvadratickou rychlostí,*
2. *posloupnost* $\{\|\nabla f(x_k)\|_2\}_{k=0}^{\infty}$ *konverguje k nule rovněž kvadraticky.*

Důkaz: viz [1], str. 490.

Poznámka 4.3. Lze ukázat, že $\eta \in \mathbb{R}$ z vět 4.2 a 4.3 je možné vyjádřit ve tvaru

$$\eta = \min\{1, 3(1-2\alpha)\}\frac{m^2}{L}, \tag{37}$$

přičemž je zřejmé, že platí $0 < \eta \leq \frac{m^2}{L}$.

Věta 4.4. *Počet iteračních kroků Newtonovy metody v kvadratické fázi není větší než*

$$\log_2\left(\log_2\frac{\varepsilon_0}{\varepsilon}\right),$$

kde $\varepsilon_0 = \frac{2m^3}{L^2}$, ε *je zadaná tolerance přesnosti.*

Důkaz: Ukážeme si pouze náznak důkazu v jednotlivých krocích. Podrobněji viz [1], str. 488. V prvním kroku důkazu se ukáže platnost vztahu

$$\frac{L}{2m^2}\|\nabla f(x_{k+1})\|_2 \leq \left(\frac{L}{2m^2}\|\nabla f(x_k)\|_2\right)^2. \tag{38}$$

Odtud vyplyne, že je-li $\|\nabla f(x_k)\|_2 < \eta$, $k \in \mathbb{N}_0$, pak $\|\nabla f(x_l)\|_2 < \eta$, $\forall l \in \mathbb{N}_0$, $l \geq k$. Jinými slovy, dostane-li se Newtonova posloupnost do kvadratické fáze, potom již stále bude konvergovat kvadraticky.
Chceme nyní určit počet iteračních kroků $l - k$ tak, aby platilo

$$f(x_l) - f(x^*) \leq K = \varepsilon, \quad l \in \mathbb{N}_0,\ K \in \mathbb{R}_0^+.$$

Aplikujeme vztah (38) rekurzivně, čímž je možno dojít k nerovnosti

$$\frac{L}{2m^2}\|\nabla f(x_l)\|_2 \leq \left(\frac{L}{2m^2}\|\nabla f(x_k)\|_2\right)^{2^{l-k}} \leq \left(\frac{1}{2}\right)^{2^{l-k}}, \quad \forall l \geq k,$$

a tedy s využitím vztahu

$$\lambda^2(x_l) = \nabla f(x_l)\left[\mathbf{D}^2 f(x_l)\right]^{-1}\nabla f(x_l)^{\mathrm{T}} \leq \frac{1}{m}\|\nabla f(x_l)\|_2^2$$

platí

$$f(x_l) - f(x^*) \approx \frac{\lambda^2(x_l)}{2} \leq \frac{1}{2m}\|\nabla f(x_l)\|_2^2 \leq \frac{2m^3}{L^2}\left(\frac{1}{2}\right)^{2^{l-k+1}} = \varepsilon. \qquad (39)$$

Zde již vidíme, že zlogaritmujeme-li dvakrát vztah (39), dostáváme tvrzení věty.

□

4.4.3 Shrnutí

Newtonova metoda konverguje ve dvou etapách:

1. **Tlumená fáze konvergence** ($t_k \leq 1$)

   $\|\nabla f(x_k)\|_2 \geq \min\{1, 3(1-2\alpha)\}\frac{m^2}{L}$

   počet iteračních kroků je menší než $\quad \frac{M^2L^2}{m^5\alpha\beta\min\{1,9(1-2\alpha)^2\}}(f(x_0) - f(x^*))$

2. **Kvadratická fáze konvergence** ($t_k = 1$)

   $\|\nabla f(x_k)\|_2 < \min\{1, 3(1-2\alpha)\}\frac{m^2}{L}$

   počet iteračních kroků je menší než $\quad \log_2\left(\log_2 \frac{2m^3}{L^2\varepsilon}\right)$

Přihlédneme-li k faktu, že výraz $\log_2\left(\log_2 \frac{\varepsilon_0}{\varepsilon}\right)$ roste velmi pomalu se zmenšujícím se ε, můžeme jej považovat za konstantu. Již šest iterací v kvadratické fázi způsobí, že $\varepsilon \approx 5 \cdot 10^{-20}\varepsilon_0$ (viz [1], str. 489). Tedy můžeme psát

$$\log_2\left(\log_2 \frac{\varepsilon_0}{\varepsilon}\right) \approx 6.$$

Věta 4.5. *Celkový počet iterací Newtonovy optimalizační metody není větší než výraz*

$$\frac{M^2L^2}{m^5\alpha\beta\min\{1, 9(1-2\alpha)^2\}}(f(x_0) - f(x^*)) \quad + \quad \log_2\left(\log_2\frac{2m^3}{L^2\varepsilon}\right) \tag{40}$$

Důkaz: Plyne z důsledku 4.1 a věty 4.4.

Na příkladě si ukážeme závislost chyby Newtonovy metody v k-tém kroku, tj. $f(x_k) - f(x^*)$, na počtu iteračních kroků k. Zvolme si nějakou funkci, jež splňuje předpoklady této kapitoly. Nebudeme volit kvadratickou funkci, poněvadž již víme, že při libovolném počátečním odhadu x_0 metoda najde přesné řešení po prvním kroku (viz příklad 4.5).

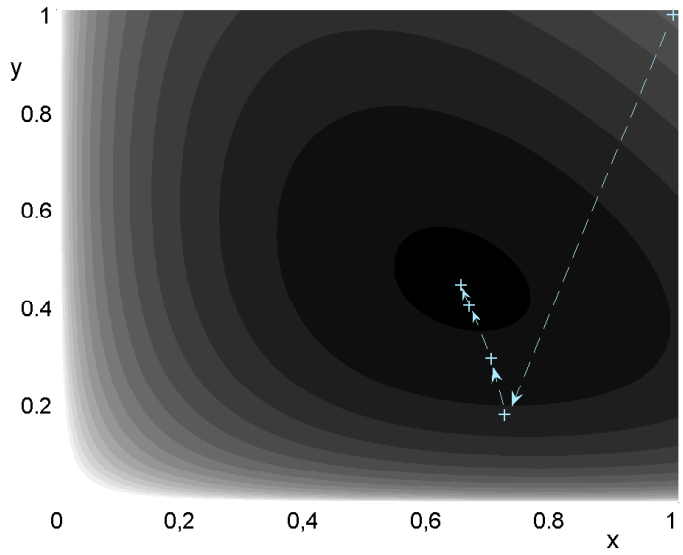
Příklad 4.6. Pomocí Newtonovy optimalizační metody najděte minimum funkce $f(x, y) = (x-1)^2 + (x+y)^2 - \ln(xy)$, určete, při jakém počtu kroků dosáhneme přesnosti $\varepsilon = 10^{-15}$.

Řešení: Zvolme počáteční odhad $x_0 = [1, 1]$ a zadejme data do Matlabu.

```
x = sym('[x1,x2]'); x0 = [1,1];
fce = (x(1) - 1).^2 - log(x(1).*x(2)) + (x(1) + x(2)).^2;
[xk,f_xk,k] = newtonmin(fce,x,x0,10^-15);
```

Matlab odpoví:

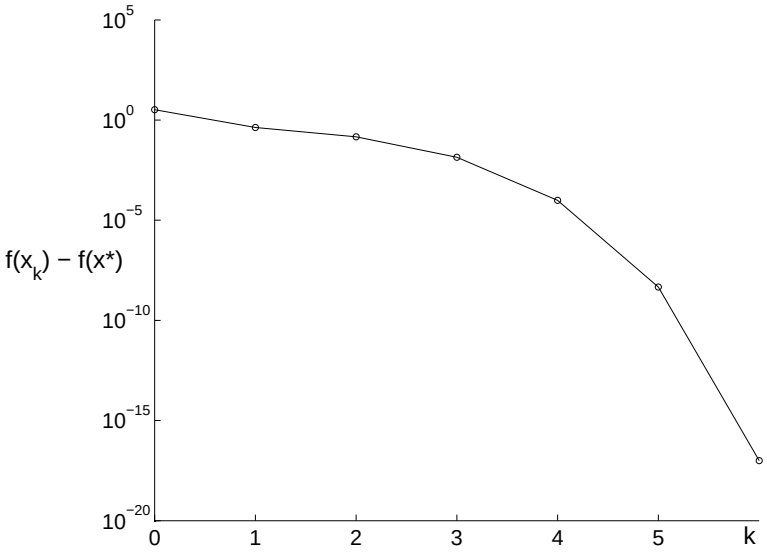
```
xk =
    0.6556    0.4516
f_xk =
    2.5617
k =
     6
```



Obrázek 11: Graf funkce $f(x, y) = (x-1)^2 + (x+y)^2 - \ln(xy)$, promítnutý do roviny Oxy, a Newtonovy posloupnosti $\{x_k\}$ (první čtyři iterace).

Tímto jsme zjistili, že bodem minima funkce f je bod [0,6556; 0,4516], v němž funkce nabývá hodnotu 2,5617, a to s přesností 10^{-15}. Necháme-li si v každém kroku vypsat kvadrát Newtonova úbytku $\lambda^2(x_k)$, získáme vektor `lambda`, který vydělíme dvěma, zadáme do Matlabu a nakreslíme graf závislosti $f(x_k) - f(x^*)$ na počtu iteračních kroků k (viz obrázek 12).

```
lambda = [3.2727,0.4231,0.1457,0.0138,9.8423e-005,...
          4.5850e-009,9.8695e-018];
lambda = lambda ./ 2
         [1.6364,0.2115,0.0728,0.0069,4.9212e-005,...
          2.2925e-009,4.9348e-018];


k = [0:6];
plot(k,lambda);
```



Obrázek 12: Závislost rozdílu $f(x_k) - f(x^*)$ na počtu iteračních kroků k při použití Newtonovy metody na účelovou funkci $f(x, y) = (x-1)^2 + (x+y)^2 - \ln(xy)$.

Zjistili jsme tedy, že již po šesti krocích dosáhneme přesnosti 10^{-15}, což vypovídá o velmi vysoké rychlosti konvergence algoritmu.

Při volbě jiného počátečního odhadu, například bodu $x_0 = [5; 0{,}5]$, dostaneme naprosto stejný výsledek a to již po pěti iteračních krocích.

Otestujeme-li další různé počáteční odhady, zjistíme, že okolí bodu minima, ze kterého lze volit počáteční odhad tak, aby metoda k tomuto bodu konvergovala je relativně velké, neboť například pro $x_0 = [x_0^1; x_0^2]$, kde $x_0^1 \in \mathbb{R}^+$ a $x_0^2 \in (0, 1\rangle$, metoda konverguje k bodu [0,6556; 0,4516]. Později si však ukážeme, že takováto volnost ve volbě počátečního odhadu obecně neplatí.

Příklad 4.7. Najděte minimum funkce $f(x) = -\ln(1 - ax^{\mathrm{T}}) - \sum_{i=1}^{20} \ln(1 - x_i^2)$, kde $x \in \mathbb{R}^{20}$, s přesností $\varepsilon = 10^{-20}$. Použijte Newtonovu optimalizační metodu a nakreslete graf závislosti rozdílu $f(x_k) - f(x^*)$ na počtu iteračních kroků k. Vektor $a \in \mathbb{R}^{20}$ je náhodný reálný vektor, $a_i \in \langle 0; 10\rangle$, $\forall i = 1, \ldots, 20$.

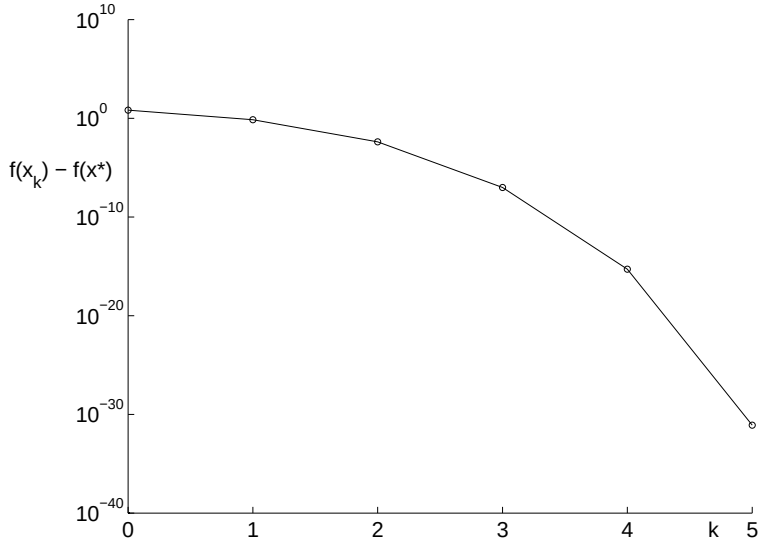
Řešení: V Matlabu si napíšeme následující skript, na jehož konci spustíme M-file `newtonmin`. Jako počáteční odhad zvolíme $x_0 = (-0{,}5\ ;\ -0{,}5\ ;\ \ldots\ ;\ -0{,}5) \in \mathbb{R}^{20}$.

```
n = 20; x = sym('[ ]'); fce = sym('0');
%vytvoříme vektor proměnných x a počáteční odhad x0
for (i = 1:n)
    x(i) = sym(['x',int2str(i)]);
    x0(i) = -0.5;
end
a = 10 .* rand(n,1);
%vytvoříme účelovou funkci
fce = fce - log(1 - x * a);
for(i = 1:n)
    fce = fce - log(1 - x(i).^2);
end

[xk,f_xk,k] = newtonmin(fce,x,x0,10^-20)
```

Spustíme skript a zjistíme tak, že pro konkrétní, náhodně vybraný vektor a je řešení úlohy ve tvaru

```
xk =
   -0.1634   -0.2067   -0.2372   -0.1938   -0.0480 ...
   -0.1093   -0.2404   -0.2361   -0.1105   -0.2307 ...
   -0.0158   -0.0954   -0.2118   -0.0027   -0.0378 ...
   -0.0551   -0.0540   -0.1604   -0.0738   -0.0541
f_xk =
   -2.4465
k =
     5
```

Graf závislosti rozdílu $f(x_k) - f(x^*)$ na počtu iteračních kroků k nakreslíme pomocí stejného postupu jako v příkladě 4.6.



Obrázek 13: Závislost rozdílu $f(x_k) - f(x^*)$ na počtu iteračních kroků k při použití Newtonovy metody na účelovou funkci $f(x) = -\ln(1 - ax^{\mathrm{T}}) - \sum_{i=1}^{20} \ln(1 - x_i^2)$.

Vidíme, že postačí pouze pět iteračních kroků, abychom dosáhli výborné aproximace minima účelové funkce. Právě rychlá konvergence metody je její největší výhodou.

Poznámka 4.4. V příkladě 4.7 pracujeme ve vektorovém prostoru s dimenzí dvacet, a tedy se nabízí otázka, zda není vhodnější použít M-file `numernewtonmin`, jenž využívá numerickou derivaci k urychlení výpočtu, namísto klasického M-souboru `newtonmin`.

Použitím funkce `numernewtonmin` dojdeme k srovnatelně kvalitnímu výsledku v mnohonásobně kratším „počítačovém“ čase, avšak spuštění funkce `newtonmin` je jednodušší a máme větší jistotu, že se při zadávání účelové funkce nedopustíme chyby. Navíc používání numerických derivací může vést i k nejistotě ohledně konvergence metody.

4.5 Výhody a nevýhody metody

Řekli jsme si již, jak se Newtonova optimalizační metoda používá, podle jakého „receptu“ postupujeme při konstrukci minimalizující posloupnosti a také jsme odhadli rychlost konvergence metody. Je na čase, abychom si tyto poznatky shrnuli a zamysleli se nad efektivitou jejího praktického využití.

Newtonova metoda má několik velmi podstatných výhod, díky kterým je často používána, avšak i zde platí známé „něco za něco“, a tudíž se musíme potýkat i s jistými nevýhodami této metody.

Výhody:

+ Konvergence Newtonovy metody je obecně velmi rychlá a dosahuje rychlosti až kvadratické poblíž bodu x^* (viz příklad 4.6 a 4.7).

+ Dostane-li se Newtonova metoda do kvadratické fáze konvergence, pak po přibližně šesti iteracích dosáhne velmi vysoké přesnosti odhadu (viz příklad 4.6).

+ Newtonova metoda je invariantní vzhledem ke změnám souřadnic.

+ Výkonnost Newtonovy metody je podobná v prostorech různě velkých dimenzí. Například rozdíl mezi chováním metody v $\mathbb{R}^2$ a $\mathbb{R}^{20}$ je pouze ve větší časové náročnosti pro výpočet Hessovy matice účelové funkce v daném bodě (viz příklad 4.7). Při přechodu mezi prostory $\mathbb{R}^{10}$ a $\mathbb{R}^{1000}$ je možno pozorovat mírné navýšení počtu potřebných iteračních kroků.

+ Newtonova metoda nezávisí na volbě parametrů, které by mohly ovlivnit její efektivitu.

Nevýhody:

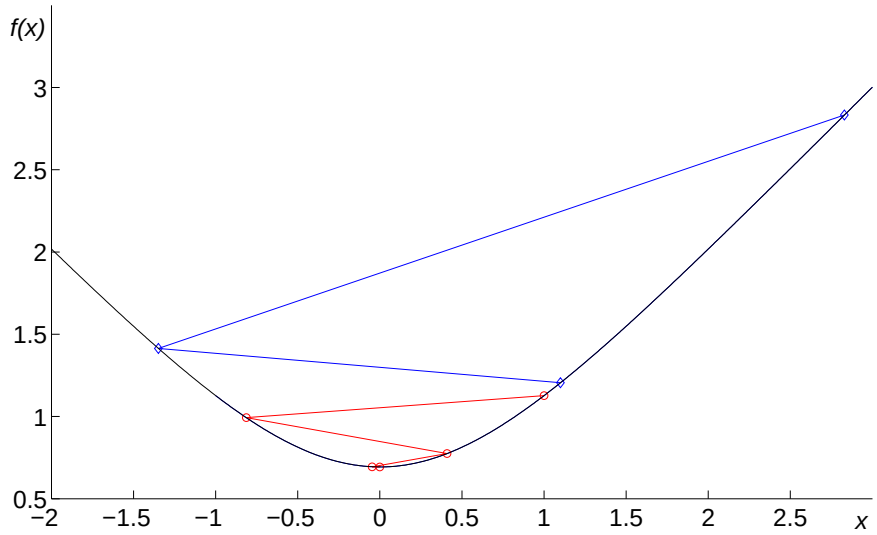
– Newtonova metoda je paměťově velmi náročná zejména z důvodu výpočtů Hessovy matice účelové funkce a Newtonova kroku (viz M-file 4.2).

– Požadavek dobrého zvolení počátečního odhadu je u Newtonovy metody velmi podstatný, jelikož by se mohlo stát, že by metoda začala divergovat

(viz příklad 4.8) nebo bychom konvergovali do míst, kam jsme vůbec nechtěli (viz příklad 4.9). Nároky na kvalitu počátečního odhadu se zvyšují se zvyšujícím se počtem nezávisle proměnných účelové funkce.

Podívejme se v závěru této kapitoly ještě na dva příklady, které poukazují na důležitost volby počátečního odhadu při aplikaci Newtonovy optimalizační metody.

Příklad 4.8. Funkce $f(x) = \ln(e^x + e^{-x})$ má jediné minimum, a to bod $x^* = 0$. Najděte jej pomocí Newtonovy optimalizační metody, použijte jako počáteční odhad a) $x_0 = 1$, resp. b) $x_0 = 1{,}1$.



Obrázek 14: Graf funkce $f(x) = \ln(e^x + e^{-x})$. Červená čára znázorňuje postup Newtonovy metody při volbě počátečního odhadu $x_0 = 1$, modrá pak její postup při volbě $x_0 = 1{,}1$.

Řešení:

a) Zadáme data do Matlabu a použijeme M-file `newtonmin`:

```
x = sym('x'); f = log(exp(x) + exp(-x));
[xk,f_xk,k] = newtonmin(f,x,1,10^-15)
```

Tímto po pěti krocích dosáhneme odhadu minima $x_k = -2{,}3471 \cdot 10^{-13} \approx 0$.

b) Použijeme-li stejný postup jako v a) zjistíme, že metoda pro takto zvolený počáteční odhad diverguje. Při použití funkce `newtonmin` nám tuto skutečnost Matlab ohlásí.

Příklad 4.9. Najděte lokální minimum funkce $f(x, y) = \sqrt{x} + \sin(2x) \sin y$ pomocí Newtonovy optimalizační metody s přesností 10^{-15}. Zvolte jako počáteční odhad a) $x_0 = [4{,}1;\ 4{,}3]$, resp. b) $x_0 = [4{,}3;\ 4{,}3]$.

Řešení: Budeme postupovat stejně jako v předchozích příkladech, tedy využijeme M-file `newtonmin`.

a)
```
x = sym('[x1,x2]');
fce = sin(2*x(1)).*sin(x(2)) + sqrt(x(1));
[xk,f_xk,k] = newtonmin(fce,x,[4.1,4.3],10^-15)
```

Dozvíme se tak, že

```
xk =
    3.8632    4.7124
f_xk =                  k =
    0.9736                  4
```

Naše účelová funkce f má lokální minimum v bodě [3,8632; 4,7124].

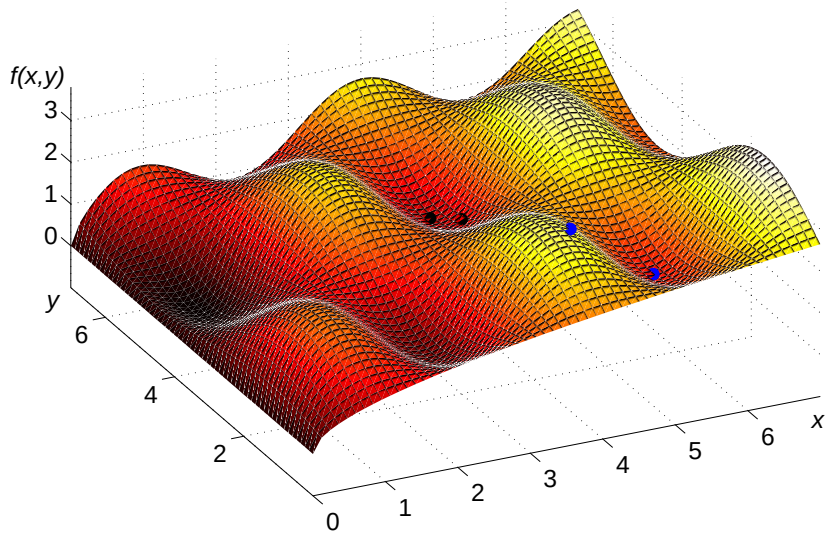
b) Zadáme-li

```
[xk,f_xk,k] = newtonmin(fce,x,[4.3,4.3],10^-15)
```

Zjistíme, že

```
xk =
    5.4441    1.5708
f_xk =                    k =
    1.3390                    4
```

V tomto případě jsme našli lokální minimum v bodě [5,4441; 1,5708].



Obrázek 15: Graf funkce $f(x, y) = \sqrt{x} + \sin(2x) \sin y$. Na obrázku jsou vyznačeny počáteční odhady x_0 a odhady řešení x^*, v případě a) černě, b) modře.

Z tohoto příkladu je zřejmé, že může nastat situace, kdy relativně malá změna počátečního odhadu zapříčiní konvergenci metody k různým bodům lokálního minima, přičemž mnohdy postačí, bude-li změněna pouze jedna komponenta počátečního odhadu.

V současné době se řeší i otázka redukce paměťové náročnosti metody. Jednou z možností může být například použití **Choleského rozkladu** (L-U rozklad symetrické matice) či **metoda proměnné metriky** (viz [2] str. 119), jinou pak aplikace **kvazi-Newtonovských metod** (viz [7] str. 192). Tímto se však zde hlouběji zabývat nebudeme.

5 Sebeomezující funkce

V kapitole 4.4, jež se věnovala konvergenci Newtonovy optimalizační metody, jsme se dozvěděli, že výsledný odhad počtu kroků Newtonovy metody při zvolené toleranci přesnosti závisí na trojici reálných konstant m, M a L. Nedostatkem ovšem je skutečnost, že tyto konstanty většinou neznáme, a tedy nedokážeme ani konkrétněji specifikovat hledaný odhad počtu kroků metody. Navíc se musíme smířit s tím, že se tyto konstanty mění při změně souřadnicového systému, což vzhledem k afinní invarianci samotné Newtonovy metody může vést k menší průhlednosti či neestetičnosti analýzy této metody.

Z těchto důvodů si můžeme položit otázku, zda existuje ekvivalentní vyjádření předpokladů

$$m\mathbf{E} \preceq \mathbf{D}^2 f(x) \preceq M\mathbf{E}, \quad \left\|\mathbf{D}^2 f(x) - \mathbf{D}^2 f(y)\right\|_2 \leq L\|x - y\|_2,$$

které by bylo nezávislé na změně souřadnic a mohli bychom je použít k analýze Newtonovy metody. Kladně tuto otázku zodpověděli pánové Nesterov a Nemirovski, kteří jako první definovali **sebeomezující funkce**[1].

Cílem této kapitoly je seznámení čtenáře s problematikou sebeomezujících funkcí, zejména s jejich základními vlastnostmi a následně pak jejich využitím, jež je úzce spojeno nejen s Newtonovou metodou, ale i s dalšími optimalizačními metodami.

5.1 Definice

Jak tedy vypadá taková sebeomezující funkce? Začněme s její definicí, jde-li o funkci jedné proměnné.

Definice 5.1. Třikrát spojitě diferencovatelná konvexní funkce $f : \mathcal{D}(f) \to \mathbb{R}$, kde $\mathcal{D}(f) \subseteq \mathbb{R}$, se nazývá **sebeomezující na množině** $\mathcal{A} \subseteq \mathcal{D}(f)$, jestliže platí vztah

$$|f'''(x)| \leq 2[f''(x)]^{\frac{3}{2}}, \quad \forall x \in \mathcal{A}. \tag{41}$$

[1] překl. z angl. originálu *self-concordant functions*

Je-li $\mathcal{A} = \mathcal{D}(f)$, říkáme, že funkce f je **sebeomezující**.

Poznámka 5.1. Z definice 5.1 plyne, že pro každou sebeomezující funkci platí

$$f''(x) \geq 0, \quad \forall x \in \mathcal{D}(f).$$

Poznámka 5.2. Je zřejmé, že všechny lineární a konvexní kvadratické funkce jsou sebeomezující, neboť jejich třetí derivace jsou identicky nulové.

Skutečnost, že se v definici 5.1 vyskytuje ve vztahu (41) právě konstanta 2, je zavedená konvence pro zjednodušení výrazů, jež budou následovat. Bez újmy na obecnosti bychom mohli namísto této konstanty dosadit do vztahu (41) libovolné kladné reálné číslo K. Je zřejmé, že $K > 0$, neboť v opačném případě, pro $K \leq 0$, by definice sebeomezující funkce, vzhledem k nezápornosti levé strany nerovnosti (41), ztratila svůj původní smysl.

Věta 5.1. *Buď dána funkce $\hat{f} : \mathcal{D}\left(\hat{f}\right) \to \mathbb{R}$, kde $\mathcal{D}\left(\hat{f}\right) \subseteq \mathbb{R}$. Jestliže je splněna nerovnost*

$$\left|\hat{f}'''(x)\right| \leq K\left[\hat{f}''(x)\right]^{\frac{3}{2}}, \quad \forall x \in \mathcal{D}\left(\hat{f}\right), \tag{42}$$

pro nějaké kladné reálné číslo K, pak je funkce $f(x) = \frac{K^2}{4}\hat{f}(x)$ sebeomezující.

Důkaz: Chceme ukázat, že funkce f splňuje podmínku (41). Jistě platí vztahy

$$|f'''(x)| = \frac{K^2}{4}\left|\hat{f}'''(x)\right| \leq \frac{K^3}{4}\left[\hat{f}''(x)\right]^{\frac{3}{2}} =$$

$$= \frac{K^3}{4}\left[\frac{4}{K^2}f''(x)\right]^{\frac{3}{2}} = 2[f''(x)]^{\frac{3}{2}},$$

a tedy tvrzení věty je pravdivé.

□

Definice 5.2. Jestliže třikrát spojitě diferencovatelná funkce $\hat{f} : \mathcal{D}\left(\hat{f}\right) \to \mathbb{R}$, kde $\mathcal{D}\left(\hat{f}\right) \subseteq \mathbb{R}$, splňuje vztah (42) pro všechna $x \in \mathcal{D}\left(\hat{f}\right)$, řekneme, že $\hat{f}$ je **K-sebeomezující**.

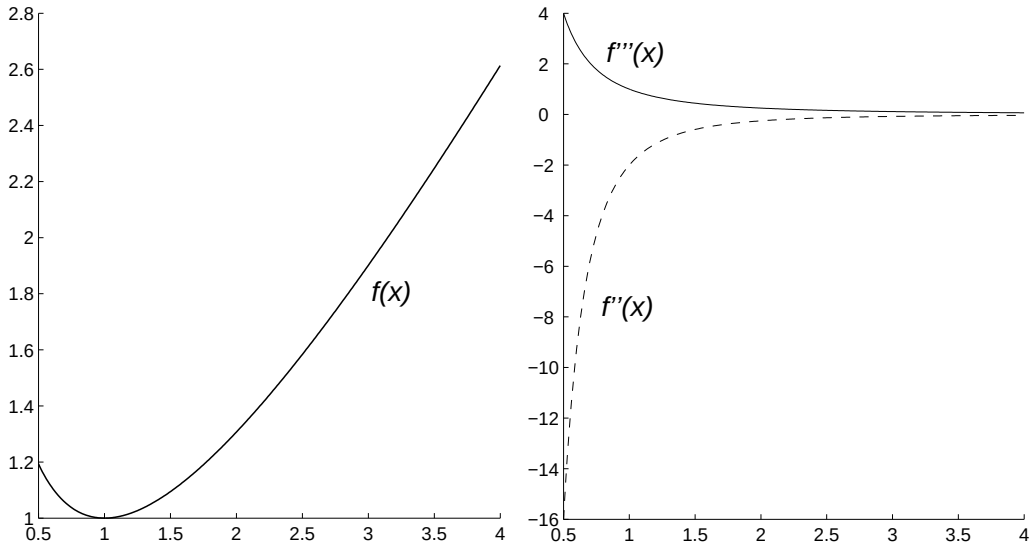
Dále budeme zkoumat pouze sebeomezující funkce, neboť díky větě 5.1 můžeme říci, že vlastnosti K-sebeomezujících funkcí jsou v podstatě stejné jako vlastnosti sebeomezujících funkcí.

Příklad 5.1. Ukažte, že funkce $f(x) = x - \ln x$ je sebeomezující.

Řešení: Nejdříve vypočteme postupně první tři derivace funkce $f(x)$. Zjistíme, že $f'(x) = 1 - \frac{1}{x}$, $f''(x) = \frac{1}{x^2}$, $f'''(x) = -\frac{2}{x^3}$. Nyní ověříme, zda je splněna podmínka (41). Pouhým dosazením a jednoduchou úpravou zjistíme, že

$$2[f''(x)]^{\frac{3}{2}} = 2\left(\frac{1}{x^2}\right)^{\frac{3}{2}} = \frac{2}{x^3} = |f'''(x)|\,, \quad \forall x \in \mathcal{D}(f)\,.$$

Vzhledem k tomu, že vztah (41) povoluje i rovnost, je funkce $f(x) = x - \ln x$ sebeomezující.



Obrázek 16: Graf funkce $f(x) = x - \ln x$ (vlevo) a grafy její druhé a třetí derivace (vpravo).

Příklad 5.2. Funkce $g(x) = x \ln x - \ln x$, kde $x \in \mathbb{R}^+$ a také $h(x) = -\ln x^p$, kde $x \in \mathbb{R}^+$, pro $p \in \langle 1; +\infty)$, jsou sebeomezující funkce. Při ověřování tohoto tvrzení lze postupovat analogicky jako v příkladě 5.1.

Při rozhodování o sebeomezenosti dané funkce f na intervalu (a, b) můžeme rovněž využít M-file, jenž vychází z podobného principu jako vykreslení grafu funkce v Matlabu. Tedy vygenerujeme si jemnou síť pro nezávisle proměnnou x na intervalu (a, b) a otestujeme pro všechny tyto hodnoty podmínku (41). Platí-li tato podmínka ve všech případech, můžeme, vzhledem ke spojitosti funkce f i jejích derivací až do třetího řádu, říci, že je daná funkce na (a, b) sebeomezující.

M-soubor 5.1. (Testování sebeomezenosti funkce na omezeném intervalu)

```
function [] = isselfc(f,interval)

%isselfc
%   isselfc(F,INTERVAL) testuje, zda je funkce F sebeomezující
%   na daném INTERVALU
%   vstup: F        - symbolická funkce jedné proměnné
%          INTERVAL - [a,b], kde a, b jsou reálná čísla
%   výstup: Matlab napíše, zda funkce F je či není sebeomezující
%           na daném intervalu.

% kontrola, zda je interval správně zadán
if(interval(1) > interval(2))
    inter = interval(1);
    interval(1) = interval(2);
    interval(2) = inter;
end

v = [(interval(1) + 10^-32):0.01:(interval(2) - 10^-32)];
df = diff(f);
ddf = diff(df);
dddf = diff(ddf);
kriterium = 2.*(subs(ddf,v)).^(3/2) - abs(subs(dddf,v));
kriterium = round(10^10.*kriterium)./10^10;
```

```
% testování podmínky
i = 1;
while(i <= length(v) & kriterium(i) >= 0)
    i = i + 1;
end
% rozhodnutí o výsledku
if(i == (length(v) + 1))
  disp('Testovaná funkce je na daném intervalu sebeomezující.');
else
  disp('Testovaná funkce není na daném intervalu sebeomezující.');
end
```

Následující definice nám ukazuje, jaká podmínka musí být splněna, má-li být funkce více proměnných sebeomezující. Stručně řečeno musí být tato funkce sebeomezující podél každého směru na svém definičním oboru.

Definice 5.3. Řekneme, že funkce $f : \mathcal{D}(f) \to \mathbb{R}$, kde $\mathcal{D}(f) \subseteq \mathbb{R}^n$, $n \in \mathbb{N}$, $n > 1$, je **sebeomezující**, jestliže je sebeomezující funkce jedné proměnné definovaná předpisem $\hat{f}(t) = f(x_0 + t\alpha)$ pro libovolné $\alpha \in \mathbb{R}^n$ takové, že x_0, $x_0 + t\alpha \in \mathcal{D}(f)$.

Příklad 5.3. Ukažte, že funkce $f : \mathbb{R}^+ \times \mathbb{R} \to \mathbb{R}$, jež má explicitní vyjádření ve tvaru $f(x, y) = y - \ln x$, je sebeomezující.

Řešení: Postupovat budeme tak, že ověříme platnost definice 5.3. Nejprve zvolme libovolný vektor $\alpha = (\alpha_1, \alpha_2)$ a bod (x_0, y_0) vyhovující již zmiňované definici. Funkci $\hat{f}(t)$ definujeme vztahem

$$\hat{f}(t) = f(x_0 + t\alpha_1, y_0 + t\alpha_2) = y_0 + t\alpha_2 - \ln(x_0 + t\alpha_1).$$

Dále $\hat{f}(t)$ zderivujeme, čímž dostáváme

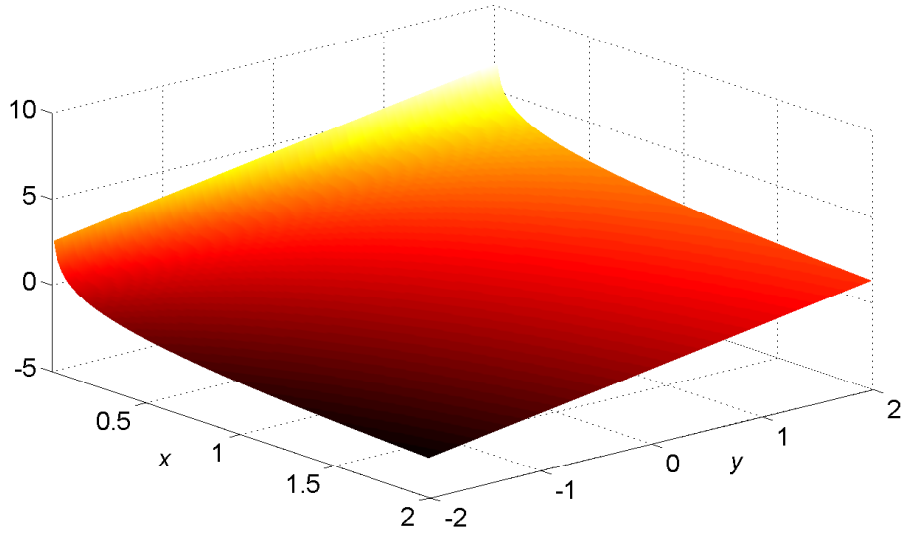
$$\hat{f}''(t) = (x_0 + t\alpha_1)^{-2} \alpha_1^2,$$

$$\hat{f}'''(t) = -2\left(x_0 + t\alpha_1\right)^{-3}\alpha_1^3.$$

Jistě platí rovnosti

$$2[\hat{f}''(t)]^{\frac{3}{2}} = 2[\left(x_0 + t\alpha_1\right)^{-2}\alpha_1^2]^{\frac{3}{2}} = 2\left(x_0 + t\alpha_1\right)^{-3}\alpha_1^3 = \left|\hat{f}'''(t)\right|,$$

což znamená, že funkce $f(x, y)$ je sebeomezující.



Obrázek 17: Graf funkce $f(x, y) = y - \ln x$.

5.2 Vlastnosti sebeomezujících funkcí

Sebeomezující funkce mají několik zajímavých vlastností, jež nám mohou posloužit nejen k vytváření dalších sebeomezujících funkcí, ale i k lepší orientaci mezi těmito funkcemi a také k rozhodování, zda daná funkce je či není sebeomezující.

Nejdříve se zaměříme na množinu všech sebeomezujících funkcí $f : \mathcal{D}(f) \to \mathbb{R}$, kde $\mathcal{D}(f) \subseteq \mathbb{R}^n$. Označme ji $\mathcal{S}^n$, $n \in \mathbb{N}$, a platí, že

$$\mathcal{S}^n \subseteq \left\{f \in \mathcal{C}^3(\mathcal{D}(f)) \ ; \ \mathcal{D}(f) \subseteq \mathbb{R}^n\right\}.$$

Věta 5.2. *Množina $\mathcal{S}^n$, $\forall n \in \mathbb{N}$, je uzavřená vzhledem k operacím sčítání funkcí a násobení funkcí reálným číslem $\alpha \geq 1$.*

Důkaz:

(a) Větu nejprve dokážeme pro $n = 1$.

Pro libovolné funkce f, $g \in \mathcal{S}^1$ a pro všechna $x \in \mathcal{D}(f) \cap \mathcal{D}(g)$ jsou splněny nerovnosti

$$|f'''(x) + g'''(x)| \leq |f'''(x)| + |g'''(x)| \leq 2\left(f''(x)^{\frac{3}{2}} + g''(x)^{\frac{3}{2}}\right).$$

Vzhledem k tomu, že sebeomezující funkce mají vždy nezápornou druhou derivaci, využijeme nerovnost

$$a^{\frac{3}{2}} + b^{\frac{3}{2}} \leq (a+b)^{\frac{3}{2}}, \quad \forall a,\ b \geq 0.$$

Dostáváme tak

$$|f'''(x) + g'''(x)| \leq 2\left(f''(x) + g''(x)\right)^{\frac{3}{2}},$$

a tedy funkce $f + g$ je sebeomezující.

Rovněž pro libovolné $\alpha \in \mathbb{R}$, $\alpha \geq 1$ platí, že funkce $\alpha f \in \mathcal{S}^1$, neboť

$$|\alpha f'''(x)| \leq 2\alpha\left(f''(x)\right)^{\frac{3}{2}} \leq 2\left(\alpha f''(x)\right)^{\frac{3}{2}}, \quad \forall x \in \mathcal{D}(f).$$

(b) Pro $n \geq 2$ plyne tvrzení věty ihned z definice 5.3 a části (a).

Spojením částí (a), (b) získáme platnost dokazovaného tvrzení.

□

Důsledek 5.1. Nechť jsou dány funkce $f(x) \in \mathcal{S}^n$, $g(y) \in \mathcal{S}^m$ $(m,\ n \in \mathbb{N})$ a jejich definiční obory $\mathcal{D}(f)$, $\mathcal{D}(g)$, pak funkce $h(x, y) = f(x) + g(y)$, definovaná na množině $\mathcal{D}(f) \times \mathcal{D}(g) \subseteq \mathbb{R}^{n+m}$, je sebeomezující.

Důkaz: Přihlédneme-li k faktu, že se na funkce $f(x)$, $g(y)$ můžeme dívat jako na funkce $(n+m)$-proměnných, je tvrzení přímým důsledkem věty 5.2.

□

Poznámka 5.3. Pro $\alpha \in (0;1)$ a sebeomezující funkci f nedokážeme obecně říci, zda funkce αf je či není sebeomezující. Je-li $\alpha < 0$, je zřejmé, že funkce αf sebeomezující není.

Definice 5.4. Nechť je dána sebeomezující funkce f. Funkci αf, kde $\alpha \in (0;1\rangle$, nazveme **minimální sebeomezující funkcí** k funkci f právě tehdy, když funkce αf je sebeomezující a platí, že funkce $r\alpha f$ není sebeomezující pro žádné $r \in (0;1\rangle$.

Vlastnost funkce „sebeomezenost“ je nezávislá na změně souřadnic. Tedy další užitečnou vlastností sebeomezujících funkcí je jejich afinní invariance, což říká následující věta.

Věta 5.3. *Mějme dánu funkci $f(x) \in \mathcal{S}^n$, kde $n \in \mathbb{N}$, dále pak $\mathbf{A} \in \mathbb{R}^{n\times n}$, $b \in \mathbb{R}^n$. Pak i funkce $f(x\mathbf{A}+b)$ je sebeomezující.*

Důkaz: Tvrzení postačí dokázat pro $n = 1$. Platnost věty pro obecné $n \in \mathbb{N}$ pak bude podle definice 5.3 zaručena.

Nechť je tedy dána sebeomezující funkce $f : \mathcal{D}(f) \to \mathbb{R}$, kde $\mathcal{D}(f) \subseteq \mathbb{R}$. Ukážeme, že i funkce $\hat{f}(x) = f(ax+b) \in \mathcal{S}^1$, přičemž $a, b \in \mathbb{R}$, $a \neq 0$. Pro derivace složené funkce $\hat{f}(x)$ platí

$$\hat{f}''(x) = a^2 f''(ax+b),$$

$$\hat{f}'''(x) = a^3 f'''(ax+b).$$

Navíc je splněna nerovnost

$$\left|a^3 f'''(ax+b)\right| \leq 2\left[a^2 f''(ax+b)\right]^{\frac{3}{2}},$$

a tedy je funkce $\hat{f}(x)$ opravdu sebeomezující.

□

Příklad 5.4. Rozhodněte, zda jsou následující funkce sebeomezující.

(a) $f(x) = (2x-1) - \ln(2x-1)$, $\mathcal{D}(f) = (\frac{1}{2};+\infty)$

(b) $f(x,y) = x - \ln x + y^2 - y + 1$, $\mathcal{D}(f) = \mathbb{R}^+ \times \mathbb{R}$

Řešení:

(a) Využijeme větu 5.3 a zavedeme lineární substituci $t = 2x - 1$. Z příkladu 5.1 víme, že funkce $\hat{f}(t) = t - \ln t$ je sebeomezující, z čehož vyplývá, že i funkce $f(x)$ je sebeomezující.

(b) Funkci $f(x, y)$ můžeme zapsat ve tvaru $f(x, y) = f_1(x) + f_2(y)$, přičemž $f_1(x) = x - \ln x$ a $f_2(y) = y^2 - y + 1$ jsou sebeomezující funkce (viz příklad 5.1 a poznámka 5.2). Tudíž podle důsledku 5.1 plyne, že i funkce $f(x, y)$ je sebeomezující.

5.3 Newtonova metoda pro sebeomezující funkce

Předpokládejme, že účelová funkce f je sebeomezující, má na svém definičním oboru stacionární bod $x^* \in \mathcal{D}(f)$, tj. $\nabla f(x^*) = o$, a její Hessova matice je pozitivně definitní. Všechny úvahy a tvrzení, které jsme si uvedli ve čtvrté kapitole tak zůstávají v platnosti. Nyní na tuto problematiku navážeme. Nejdříve se zaměříme na **konvergenci** Newtonovy metody a poté se pokusíme odstranit nejistotu při volbě **počátečního odhadu**.

5.3.1 Konvergence

Podobně jako v kapitole 4.4, kde jsme analyzovali konvergenci Newtonovy metody obecně, se teď zaměříme na tuto analýzu z pohledu sebeomezujících funkcí. Newtonova metoda opět konverguje ve dvou fázích – tlumené a kvadratické. Tvrzení, jež si uvedeme, jsou analogická větám z kapitoly 4.4, avšak s tím rozdílem, že roli gradientu převezme Newtonův úbytek a navíc se obejdeme bez neznámých konstant M, m, L. Pro srovnání s předchozím je vždy u věty, důsledku či poznámky uvedeno, které tvrzení v kapitole 4.4 je obdobné.

Mějme dánu nějakou ryze konvexní sebeomezující funkci $f : \mathcal{D}(f) \to \mathbb{R}$, $\mathcal{D}(f) \subseteq \mathbb{R}^n$, která má v bodě $x^* \in \mathcal{D}(f)$ stacionární bod. (Tento fakt implikuje, jak si později ukážeme, že f má v bodě x^* ostré globální minimum.)

Věta 5.4. *(věta 4.2) Předpokládejme, že je v k-tém kroku Newtonovy metody splněno* $\lambda(x_k) > \eta$, *kde* $\eta \in \mathbb{R}$, $0 < \eta < \frac{1}{4}$, *potom se Newtonova metoda nachází v tlumené fázi konvergence a platí vztah*

$$f(x_k) - f(x_{k+1}) \geq \alpha\beta\frac{\eta^2}{1+\eta}, \tag{43}$$

přičemž koeficienty α, β *jsou parametry, jež volíme při hledání délky k-tého kroku.*

Důkaz: viz [1], str. 504.

Předchozí věta nám říká, že rozdíl funkčních hodnot v po sobě jdoucích členech Newtonovy posloupnosti je větší než pevně dané kladné reálné číslo, z čehož vyplývá následující důsledek.

Důsledek 5.2. (důsledek 4.1) Počet iterací Newtonovy metody v tlumené fázi je shora omezen výrazem

$$\frac{1+\eta}{\alpha\beta\eta^2}(f(x_0) - f(x^*)). \tag{44}$$

Věta 5.5. *(věta 4.4) Jestliže je splněno* $\lambda(x_k) < \eta$, $\eta \in \mathbb{R}$, $0 < \eta < \frac{1}{4}$, *pak se Newtonova metoda nachází v kvadratické fázi. Počet iteračních kroků Newtonovy metody v této fázi není větší než*

$$\log_2\left(\log_2\frac{1}{\varepsilon}\right),$$

kde ε *je zadaná tolerance přesnosti.*

Důkaz: viz [1], str. 503.

Poznámka 5.4. (poznámka 4.3) Lze ukázat, že $\eta \in \mathbb{R}$ z vět 5.4 a 5.5 je možné vyjádřit ve tvaru

$$\eta = \frac{(1-2\alpha)}{4}. \tag{45}$$

Složíme vše dohromady a získáme tak finální celkové omezení na počet iteračních kroků Newtonovy optimalizační metody.

Věta 5.6. *(věta 4.5) Celkový počet iterací Newtonovy optimalizační metody není větší nežli výraz*

$$\frac{20 - 8\alpha}{\alpha\beta(1-2\alpha)^2}(f(x_0) - f(x^*)) + \log_2\left(\log_2 \frac{1}{\varepsilon}\right). \tag{46}$$

Důkaz: plyne z důsledku 5.2 a věty 5.5.

Výraz uvedený v předchozí větě závisí pouze na parametrech zpětného vyhledávání a požadované přesnosti řešení, což znamená, že pro konkrétní zvolená čísla α, β, ε jsme schopni tento výraz vyčíslit. Získáme tak určitý násobek rozdílu $f(x_0) - f(x^*)$ a k němu přičítáme konstantu, kterou můžeme nahradit číslem šest, poněvadž jak jsme si již dříve řekli, funkce $\log_2(\log_2 x)$ roste velmi pomalu.

Například při klasické volbě $\varepsilon = 10^{-10}$, $\alpha = 0{,}1$ a $\beta = 0{,}8$ obdržíme horní hranici na počet iterací ve tvaru

$$375(f(x_0) - f(x^*)) + 6.$$

Je důležité si uvědomit, že toto omezení nám ukazuje, co se může stát v nejhorším případě. Ve skutečnosti je počet potřebných iteračních kroků téměř vždy značně menší. Lze říci, že i hrubý odhad

$$f(x_0) - f(x^*) + 6$$

je vcelku slušným přiblížením počtu potřebných iterací.

5.3.2 Počáteční odhad

Pro jednoduchost budeme pracovat s funkcí jedné proměnné. Závěry, které si zde uvedeme, by ovšem bylo možné zobecnit i pro libovolnou sebeomezující funkci, přičemž namísto s absolutní hodnotou bychom pak pracovali s nějakou vhodně zvolenou normou.

Přihlédneme-li k faktu, že všechny sebeomezující funkce jsou konvexní, tj. $f''(x) \geq 0$, $\forall x \in \mathcal{D}(f)$, platí pro ně následující věta.

Věta 5.7. *Nechť je funkce $f : \mathcal{D}(f) \to \mathbb{R}$, kde $\mathcal{D}(f) \subseteq \mathbb{R}$, sebeomezující, pak:*

(i) Libovolné lokální minimum funkce f na $\mathcal{D}(f)$ je i globálním minimem.

(ii) Množina bodů z $\mathcal{D}(f)$, v nichž funkce f nabývá na $\mathcal{D}(f)$ svého minima, je konvexní. Je-li navíc funkce f ryze konvexní na $\mathcal{D}(f)$, pak je tato množina nejvýše jednoprvková.

(iii) Je-li $x^* \in \mathcal{D}(f)$ stacionárním bodem funkce f, pak je jejím globálním minimem.

Důkaz: Věta je přímým důsledkem obecnějšího tvrzení uvedeného i s důkazem například ve [2], str. 29.

□

Kdybychom zjistili, pro jakou volbu počátečního odhadu Newtonova posloupnost konverguje k přesnému řešení optimalizačního problému, získali bychom větší jistotu při používání Newtonovy metody.

Věta 5.8. *Nechť je dána sebeomezující účelová funkce $f : \mathcal{D}(f) \to \mathbb{R}$, kde $\mathcal{D}(f) \subseteq \mathbb{R}$, která má v bodě $x^* \in \mathcal{D}(f)$ lokální minimum. Jestliže existuje okolí $\mathcal{U}$ bodu x^* takové, že platí vztah*

$$2\,|f'(x)| < \sqrt{f''(x)}, \quad \forall x \in \mathcal{U}, \tag{47}$$

pak Newtonova minimalizující posloupnost $\{x_k\}$ s libovolným počátečním odhadem $x_0 \in \mathcal{U}$ konverguje k bodu x^*.

Důkaz: Definujme nejprve funkci

$$g(x) = x - \frac{f'(x)}{f''(x)}, \quad \forall x \in \mathcal{D}(f),\ f''(x) \neq 0.$$

Je zřejmé, že posloupnost $\{x_k\}$, jež je dána rekurentním vztahem $x_{k+1} = g(x_k)$, $\forall k \in \mathbb{N}$, je Newtonova minimalizující posloupnost. Bod x^* je pevným bodem

funkce g, tj. $g(x^*) = x^*$, neboť $f'(x^*) = 0$. Poněvadž je funkce f sebeomezující, tedy $f \in \mathcal{C}^3(\mathcal{D}(f))$, je funkce g spojitě diferencovatelná a její derivace má tvar

$$g'(x) = \frac{f'''(x)f'(x)}{[f''(x)]^2}.$$

Předpokládejme, že existuje okolí $\mathcal{U}$ bodu x^* takové, že je splněna podmínka (47), $\forall x \in \mathcal{U}$, pak s využitím sebeomezenosti funkce f dostáváme

$$|g'(x)| = |f'''(x)| \frac{|f'(x)|}{[f''(x)]^2} \leq 2[f''(x)]^{\frac{3}{2}} \frac{|f'(x)|}{[f''(x)]^2} = 2\frac{|f'(x)|}{\sqrt{f''(x)}} < 1 \ . \qquad (48)$$

Jistě existuje reálná konstanta $M \in (0; 1)$ taková, že

$$|g'(x)| \leq M < 1, \quad \forall x \in \mathcal{U}.$$

Zvolme libovolný bod $x_0 \in \mathcal{U}$. Podle věty o střední hodnotě diferenciálního počtu existuje y_1 ležící mezi body x_0 a x^* takové, že platí

$$|x_1 - x^*| = |g(x_0) - g(x^*)| = |g'(y_1)| \, |x_0 - x^*| \, .$$

Vzhledem k tomu, že rovněž $y_1 \in \mathcal{U}$ je

$$|x_1 - x^*| = |g'(y_1)| \, |x_0 - x^*| \leq M \, |x_0 - x^*| \, ,$$

z čehož vidíme, že bod $x_1 \in \mathcal{U}$ je blíže bodu x^*. Aplikujeme tento postup dále pro x_2, což znamená

$$|x_2 - x^*| = |g'(y_2)| \, |x_1 - x^*| \leq M \, |x_1 - x^*| \leq M^2 \, |x_0 - x^*| \, ,$$

a tak dále

$$|x_k - x^*| = |g'(y_k)| \, |x_{k-1} - x^*| \leq M^k \, |x_0 - x^*| \, , \quad \forall k \in \mathbb{N}.$$

Odtud vidíme, že platí nerovnosti

$$0 \leq \lim_{k \to \infty} |x_k - x^*| \leq |x_0 - x^*| \lim_{k \to \infty} M^k,$$

a tedy z věty o limitě tří posloupností plyne, že $\{x_k\}$ konverguje k bodu x^*, poněvadž pro $M \in (0;\,1)$ je $\lim_{k\to\infty} M^k = 0$.

□

Z předchozí věty vyplývá zřejmý důsledek vztahující se k aplikaci této věty obecně na K-sebeomezující funkce.

Důsledek 5.3. Buď dána K-sebeomezující účelová funkce $f : \mathcal{D}(f) \to \mathbb{R}$, kde $\mathcal{D}(f) \subseteq \mathbb{R}$, která má v bodě $x^* \in \mathcal{D}(f)$ lokální minimum. Pokud existuje okolí $\mathcal{U}$ bodu x^* takové, že platí vztah

$$K\,|f'(x)| < \sqrt{f''(x)}, \quad \forall x \in \mathcal{U}, \tag{49}$$

pak Newtonova minimalizující posloupnost $\{x_k\}$ s libovolným počátečním odhadem $x_0 \in \mathcal{U}$ konverguje k bodu x^*.

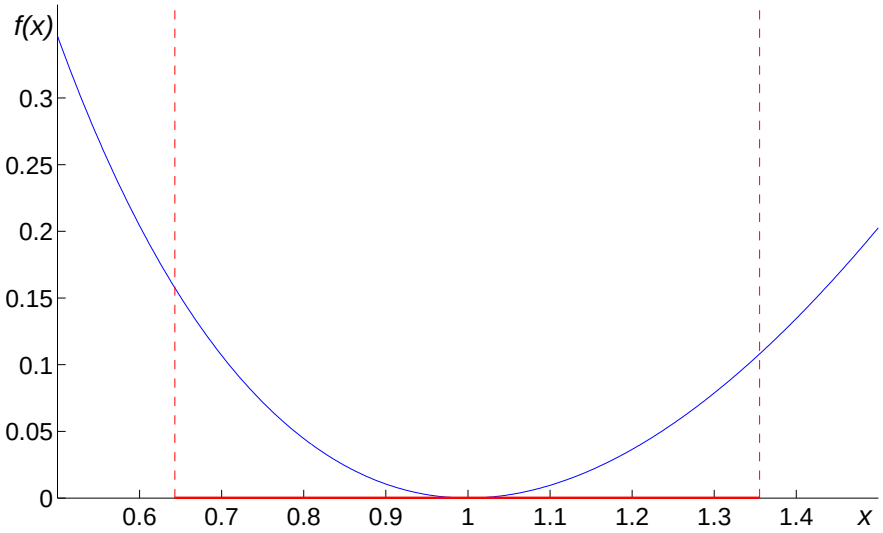
Důkaz: Postupujeme analogicky jako v důkaze věty 5.8, přičemž ve vztahu (48) figuruje konstanta K namísto dvojky.

□

Příklad 5.5. Najděte minimum sebeomezující funkce $f(x) = x \ln x - \ln x$.

Řešení: Pokud zvolíme jako počáteční odhad bod $x_0 = 2{,}1$ a použijeme M-file `newtonmin` stejně jako v předchozích příkladech, zjistíme, že Newtonova metoda diverguje.

Avšak například s počátečním odhadem $x_0 = 1{,}35$ víme, že metoda bude konvergovat, poněvadž tento bod splňuje podmínky věty 5.8. Vskutku, při takto zvoleném počátečním odhadu zjistíme pomocí funkce `newtonmin`, že $x^* = 1$ a také $f(x^*) = 0$.



Obrázek 18: Graf funkce $f(x) = x \ln x - \ln x$, červeně je vyznačena množina počátečních odhadů, pro něž Newtonova posloupnost konverguje podle věty 5.8.

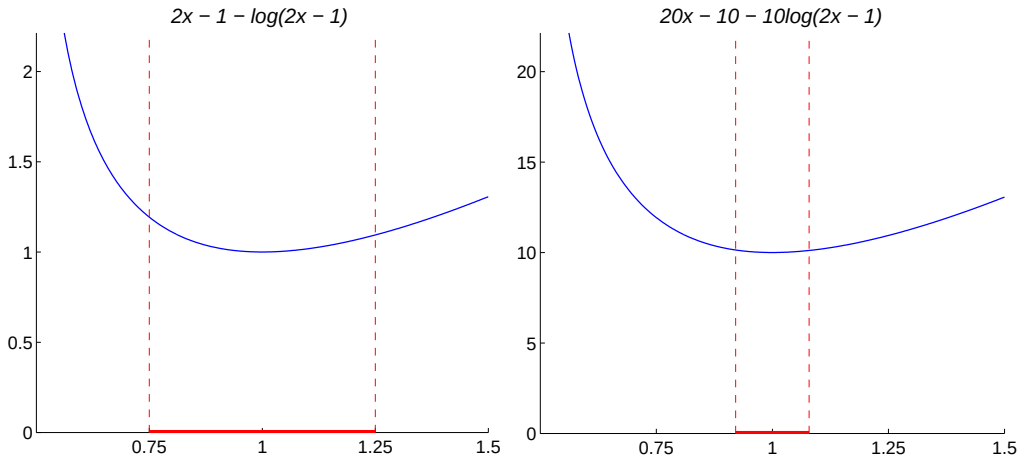
V některých příkladech lze využít faktu, že funkce f a rf, $r \in \mathbb{R}^+$, mají globální minimum ve stejném bodě. Navíc konverguje-li Newtonova minimalizující posloupnost s počátečním odhadem x_0 k přesnému řešení optimalizační úlohy pro funkci f, konverguje i pro funkci rf. Toto lze jednoduše ukázat, poněvadž platí rovnosti

$$x_{k+1} = x_k - \frac{f'(x_k)}{f''(x_k)} = x_k - \frac{rf'(x_k)}{rf''(x_k)}, \quad \forall k \in \mathbb{N}_0,$$

a tudíž jsou tyto Newtonovy posloupnosti totožné.

Příklad 5.6. Je dána funkce $f(x) = 20x - 10 - 10 \ln(2x - 1)$, najděte její globální minimum.

Řešení: Nejprve definujme funkci $g(x) = \frac{1}{10} f(x) = 2x - 1 - \ln(2x - 1)$. Tato funkce je sebeomezující (viz příklad 5.4) a zároveň má stejné globální minimum jako funkce $f(x)$. Použijeme Matlab a v něm pro tento účel naprogramovanou funkci `newtonmin`. Zvolme za počáteční odhad bod $x_0 = 1{,}2$. Metoda bude jistě konvergovat, což bychom v původním případě nevěděli. Výsledkem je bod $x^* = 1$.



Obrázek 19: Graf funkce $g(x) = 2x - 1 - \ln(2x-1)$ (vlevo) a $f(x) = 20x - 10 - 10\ln(2x-1)$ (vpravo), červeně jsou vyznačeny množiny počátečních odhadů, pro něž Newtonova posloupnost konverguje podle věty 5.8.

Na tomto příkladu můžeme vidět, že je velmi užitečné najít k účelové funkci f minimální sebeomezující funkci αf, kde $\alpha \in (0;1\rangle$, a tu pak minimalizovat, protože tak získáme větší jistotu ve volbě počátečního odhadu, jak je zřejmé z obrázku 19, a výsledek bude stejný jako u minimalizace funkce f.

Důsledek 5.4. Předpokládejme, že $f : \mathcal{D}(f) \to \mathbb{R}$, kde $\mathcal{D}(f) \subseteq \mathbb{R}$, je sebeomezující účelová funkce, která má v bodě $x^* \in \mathcal{D}(f)$ lokální minimum. Jestliže existuje okolí $\mathcal{U}$ bodu x^* a kladné reálné číslo $r \geq 2$ takové, že platí vztah

$$r\,|f'(x)| < \sqrt{f''(x)}, \quad \forall x \in \mathcal{U}, \tag{50}$$

pak Newtonova minimalizující posloupnost $\{x_k\}$ s libovolným počátečním odhadem $x_0 \in \mathcal{U}$ konverguje k bodu x^*.

Důkaz: Předpokládejme, že $r \geq 2$, pak platí nerovnosti

$$2\,|f'(x)| \leq r\,|f'(x)| < \sqrt{f''(x)}, \quad \forall x \in \mathcal{U},$$

a tudíž podle věty 5.8 Newtonova posloupnost konverguje k x^*.

□

5.4 Využití sebeomezujících funkcí

Zjistili jsme již, že Newtonova metoda funguje obecně velmi dobře pro ryze konvexní funkce. Toto tvrzení můžeme obhájit nejen empiricky získanými výsledky, ale i analýzou konvergence Newtonovy metody z kapitoly 4.4. Nevýhodou však je skutečnost, že konstanty, na kterých závisí odhad maximálního počtu iteračních kroků Newtonovy metody, ve většině případů neznáme.

Omezíme-li se při analýze Newtonovy metody pouze na sebeomezující funkce, můžeme o konvergenci metody říci více. Zejména již zmiňovaný odhad v tomto případě nezávisí na neznámých konstantách a lze jej vyjádřit jako součet malé konstanty a násobku rozdílu $f(x_0) - f(x^*)$.

Dosud není zcela známo, zda Newtonova metoda pracuje lépe pro sebeomezující funkce. I přesto mají tyto funkce své uplatnění, a to zejména v podmíněné optimalizaci.

5.4.1 Použití sebeomezujících funkcí v bariérové metodě

Uvažujme problém matematického programování ve tvaru

$$\min_x f(x), \quad \text{za podmínek } \; c_i(x) \geq 0, \forall i = 1, \ldots, m, \tag{51}$$

kde funkce $f : \mathbb{R}^n \to \mathbb{R}$ a $c_i : \mathbb{R}^n \to \mathbb{R}$, $\forall i = 1, \ldots, m$, jsou dvakrát spojitě diferencovatelné. Přípustná množina $\mathcal{D} = \{x \in \mathbb{R}^n \mid c_i(x) \geq 0, \forall i = 1, \ldots, m\}$ již obecně není celý definiční obor účelové funkce, a tudíž se jedná o podmíněnou optimalizaci. Nemůžeme tedy jednoduše použít Newtonovu metodu, jak jsme si to uvedli v případě optimalizace nepodmíněné.

Jedním z možných přístupů k řešení úlohy (51) je tvorba „bariér“ na hranici přípustné množiny. Jedná se o vhodnou úpravu účelové funkce tak, abychom měli jistotu, že při volbě počátečního odhadu $x_0 \in \mathcal{D}$ budou všechny prvky minimalizující posloupnosti rovněž ležet v přípustné množině. Tento postup se označuje jako **bariérová metoda**, přičemž roli bariér mohou hrát právě sebeomezující funkce.

Definice 5.5. Nechť $\vartheta \in \mathbb{R}^+$. Sebeomezující funkci $f : \mathcal{D} \to \mathbb{R}$, kde $\mathcal{D} \subseteq \mathbb{R}$ je uzavřená a neprázdná množina, nazýváme **sebeomezující bariérovou funkcí na $\mathcal{D}$ s parametrem** ϑ (jednoduše ϑ-sebeomezující bariérou na $\mathcal{D}$) právě tehdy, když je splněna nerovnost

$$|f'(x)| \leq \sqrt{\vartheta f''(x)}, \quad \forall x \in \mathcal{D}. \tag{52}$$

Příklad 5.7. Ukažte, že funkce $f(x) = -\ln x$ je 1-sebeomezující bariéra na $\mathbb{R}^+$.

Řešení: Snadno nahlédneme, podobně jako v příkladě 5.1, že funkce $f(x)$ je sebeomezující na $\mathbb{R}^+$.
Dále vypočteme první a druhou derivaci funkce $f(x)$, tedy

$$f'(x) = -\frac{1}{x}, \quad f''(x) = \frac{1}{x^2}$$

a nyní ověříme platnost vztahu (52), přičemž položíme $\vartheta = 1$. Jistě platí rovnosti

$$\sqrt{\vartheta f''(x)} = \frac{1}{\sqrt{x^2}} = \frac{1}{|x|} = |f'(x)|\,,$$

a tedy opravdu je funkce $f(x)$ 1-sebeomezující bariéra na $\mathbb{R}^+$.

Poznámka 5.5. Sebeomezující bariéry na $\mathcal{D} \subseteq \mathbb{R}^n$ se definují analogicky jako sebeomezující funkce více proměnných (viz definice 5.3). Tedy je potřeba, aby vztah (52) platil podél každého směru na $\mathcal{D}$.

Sebeomezující bariéry mají spoustu užitečných vlastností. Například lze ukázat (viz [6], str. 42), že funkce

$$g(x) = -\ln(ax + b), \quad \text{kde } \mathcal{D}(g) \subseteq \mathbb{R},\ a,\ b \in \mathbb{R},$$
$$h(x) = -\ln(ax^{\mathrm{T}} + b), \quad \text{kde } \mathcal{D}(h) \subset \mathbb{R}^n,\ a \in \mathbb{R}^n,\ b \in \mathbb{R}$$

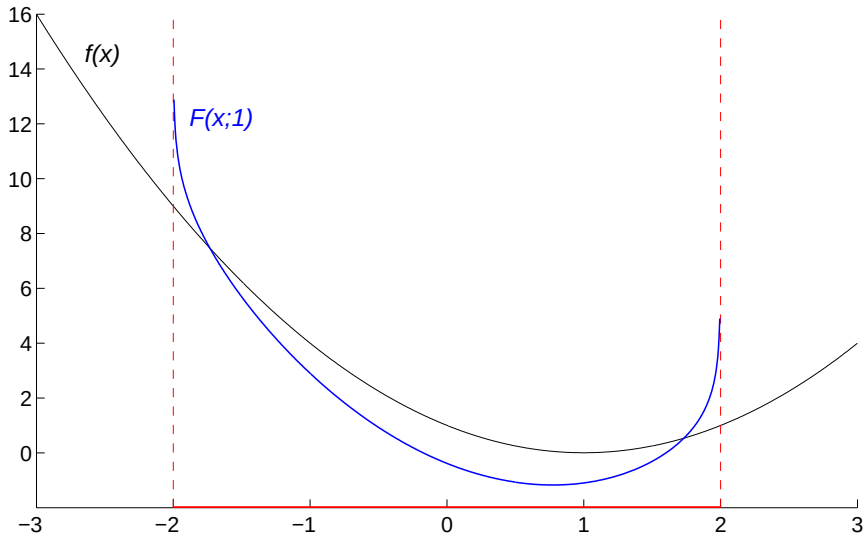
jsou, stejně jako funkce $f(x) = -\ln x$ uvedená v příkladu 5.7, 1-sebeomezujícími bariérami. Pro lepší seznámení se s těmito funkcemi je možno nahlédnou například do [6], str. 41.

Vzhledem k tomu, že je bariérová metoda popsána v mnoha publikacích (např. v [1], str. 568) a není náplní této práce, uvedeme si zde pouze poznatky nutné k pochopení v čem metoda spočívá a jak jsou v ní použity sebeomezující funkce.

Zjednodušeně řečeno, spočívá bariérová metoda v tom, že definujeme novou funkci $g(x)$, která je téměř nulová na přípustné množině a zároveň při přibližování se k hranici přípustné množiny roste funkce nade všechny meze. V důsledku pak použijeme libovolnou metodu nepodmíněné optimalizace, například Newtonovu metodu, a najdeme minimum funkce $f(x) + g(x)$.

Definice 5.6. Logaritmickou bariérovou funkcí úlohy (51) nazveme funkci

$$\mathcal{F}(x;\mu) = f(x) - \mu \sum_{i=1}^{m} \ln\left(c_i(x)\right), \quad \mu \in \mathbb{R}^+. \tag{53}$$

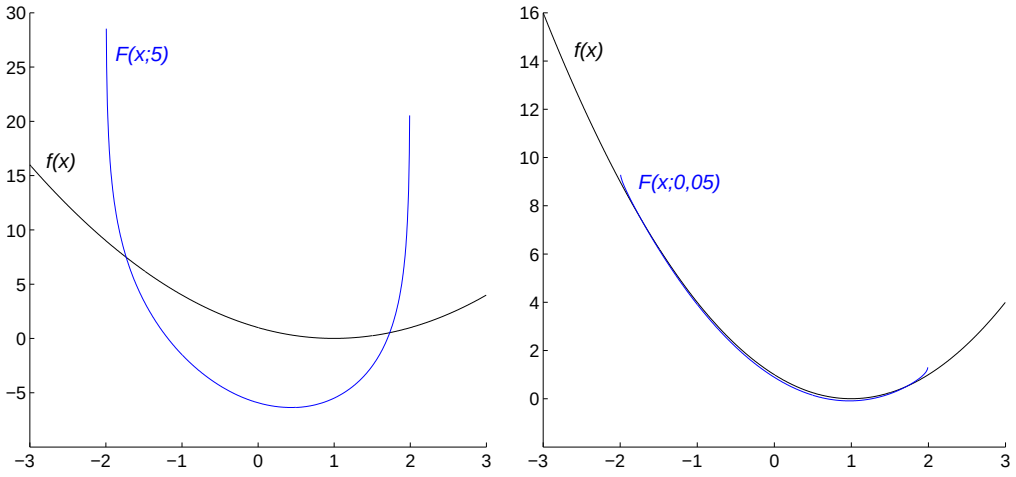


Obrázek 20: Grafická interpretace bariérové metody.

Předpokládejme, že bod x^* je lokální minimum funkce $f(x)$ na $\mathcal{D} \subset \mathbb{R}^n$. Potom funkce $\mathcal{F}$ má lokální minimum poblíž bodu x^* pro všechna dostatečně malá $\mu \in \mathbb{R}^+$. Převedeme tedy problém (51) na úlohu

$$\min_x \mathcal{F}(x;\mu), \quad x \in \mathbb{R}^n, \tag{54}$$

kterou již můžeme řešit Newtonovou metodou. Dostaneme tak odhad minima x^*. Jeho kvalita však závisí na hodnotě konstanty μ, kterou zvolíme. Je-li μ moc velké, dostaneme velmi špatný odhad. Naopak je-li moc malé, nemusí mít bariéra žádné účinky a Newtonova posloupnost nám může „vyskočit“ z přípustné množiny, což bychom rozhodně nechtěli (viz obrázek 21).



Obrázek 21: Situace při různých volbách konstanty μ.

Tento problém lze jednoduše vyřešit tak, že si zvolíme posloupnost $\{\mu_k\}$ konvergující shora k nule. Postupně pak řešíme úlohu (54) pro jednotlivá μ_k, přičemž za počáteční odhad x_0 v k-tém spuštění Newtonovy metody vždy volíme odhad minima, jejž jsme získali v předchozím kroku.

Podívejme se nyní na konkrétní příklad, který budeme řešit pomocí bariérové metody, při níž aplikujeme sebeomezující funkce.

Příklad 5.8. Najděte minimum funkce $f(x, y, z) = x - 4y + z$ na množině $\mathcal{D}$ prvků z $\mathbb{R}^3$, pro něž platí nerovnosti

$$\begin{aligned} x + y &\geq 1 \\ -3x + 2y &\geq -1 \\ y \leq 5,\ x \geq 0,\ z &\geq 0 \end{aligned} \tag{55}$$

Řešení: Nejprve sestavíme logaritmickou bariérovou funkci, jež má tvar

$$\mathcal{F}(x,y,z;\mu) = x - 4y + z - \mu\ln(x+y-1) - \mu\ln(-3x+2y+1) - \\ -\mu\ln(-y+5) - \mu\ln(x) - \mu\ln(z),$$

Dále zvolme počáteční odhad $[x_0; y_0; z_0] \in \mathcal{D}$, například bod $[1; 2; 1]$, a posloupnost $\{\mu_k\} = \left\{10^{-0,2k+1}\right\}_{k=0}^{\infty}$. Je zřejmé, že tato posloupnost konverguje k nule.

V Matlabu si sestavíme skript, který bude v cyklu používat Newtonovu metodu na účelovou funkci $\mathcal{F}(x,y,z;\mu_k)$ způsobem, který jsme si popsali výše.

```
x = sym('[x1,x2,x3]');
fx = x(1) - 4.*x(2) + x(3)^2;


c1 = x(1) + x(2) - 1;
c2 = - 3*x(1) + 2.*x(2) + 1;
c3 = x(1);
c4 = - x(2) + 5;
c5 = x(3);


xk = [1,2,1];
for(k = -1:0.2:8)
Fx = fx - (10^-k)*(log(c1)+log(c2)+log(c3)+log(c4)+log(c5));
[xk,f_xk,k] = newtonmin(Fx,x,xk,10^(-16))
end
```

Získáme tak jednoduše výsledek

```
xk =
    0.0000    5.0000    0.0001
f_xk =
  -20.0000
```

Díky tomuto postupu můžeme říci, že funkce $f(x,y,z)$ má na množině $\mathcal{D}$ minimum v bodě $[0; 5; 0]$ a nabývá zde hodnoty mínus dvacet.

Závěrem se nabízí otázka, z jakého důvodu je výhodné používat v bariérové metodě právě sebeomezující bariéry.

Sebeomezující bariéry zahrnují širokou škálu logaritmických funkcí, jsou ryze konvexní na svém definičním oboru, a tedy „nedělají problémy“ při volbě počátečního odhadu. Navíc pro sebeomezující funkce obecně známe odhad maximálního počtu iteračních kroků Newtonovy metody.

Metoda, kterou jsme si v této kapitole popsali je jednou z mnoha sloužících k podmíněné optimalizaci a je nutné se podle zadání úlohy rozhodnout, zda ji použijeme či nikoliv.

6 Příklad

Motivace

Známým matematickým problémem je aproximace funkce $f : \mathbb{R} \to \mathbb{R}$ či dat $\{[x_i, f(x_i)] \in \mathbb{R}^2; i = 0, \ldots, n\}$, $n \in \mathbb{N}$, daných tabulkou aproximačním polynomem $P_m(x)$ stupně nejvýše $m \in \mathbb{N}_0$, kde $m < n$. Tento proces je označován jako **polynomiální aproximace**. Pokud by nastala situace $m = n$, jednalo by se o polynomiální interpolaci.

Aproximace se využívá zejména ke zjednodušení implicitního zadání funkce. Tohoto lze posléze využít při přibližném výpočtu určitého integrálu funkce nebo derivace funkce v daném bodě.

Je-li dána množina bodů $[x_i, f(x_i)] \in \mathbb{R}^2$, $i = 0, \ldots, n$, chceme znát vzájemnou závislost mezi hodnotami x_i a $f(x_i)$. Může se například jednat o měření teploty vzduchu každou celou hodinu, přičemž úkolem je zjistit přibližné hodnoty teplot mezi těmito měřeními. Proložíme tedy měřením získané body určitou vhodnou křivkou. Metody, jenž tento problém řeší, hledají právě tuto křivku. V našem případě, poněvadž se zaměříme na polynomiální aproximaci, se bude jednat o vhodný polynom $P_m(x)$.

Jednou z aproximačních metod je **metoda nejmenších čtverců**. Jedná se o postup hledající polynom $P_m(x)$, jenž minimalizuje výraz

$$\rho^2(f, P_m) = \sum_{i=0}^{n} w_i \left[f(x_i) - P_m(x_i)\right]^2, \tag{56}$$

kde $w_i \geq 0$ jsou váhy, $i = 0, \ldots, n$. Funkce $\rho^2(f, P_m)$ je tedy součet obsahů čtverců o stranách $f(x_i) - P_m(x_i)$ vynásobených příslušnými vahami.

Zobecníme tento problém následovně. Nechť je dána množina uspořádaných k-tic $\left\{[x_i, f(x_i)] \in \mathbb{R}^k; i = 0, \ldots, n\right\}$, kde $x_i \in \mathbb{R}^{k-1}$. Úkolem je najít takový polynom $P_m(x)$, že minimalizuje výraz (56), přičemž $x \in \mathbb{R}^{k-1}$. Polynom $P_m(x)$ je stupně nejvýše $m \in \mathbb{N}_0$, $m < n$, a lze jej obecně zapsat ve tvaru

$$P_m(x_1, \ldots, x_{k-1}) = \sum_{i_1 + \cdots + i_{k-1} = 0}^{m} a_{i_1 \cdots i_{k-1}} {x_1}^{i_1} \cdots {x_{k-1}}^{i_{k-1}},$$

kde $a_{i_1 \cdots i_{k-1}} \in \mathbb{R}$, $i_1, \cdots, i_{k-1} \in \mathbb{N}_0$.

Tuto úlohu lze vyřešit pomocí Newtonovy optimalizační metody, což si ukážeme na následujícím příkladě.

Zadání: Aproximujte data daná tabulkou polynomem druhého stupně ve smyslu metody nejmenších čtverců.

Tabulka 4

x_i	10	15	20	25	30	35	40	45	50	55
y_i	9	10	10,5	11	11,5	12	12,5	11,5	10	9,5
$f(x_i, y_i)$	1	1,5	2,2	3	10	4,5	5	3,5	5,5	2
$w(x_i)$	1	1	1	1	1	1	1	1	1	1

Postup řešení

Hledaný polynom $P_2(x, y)$ můžeme zapsat obecně ve tvaru

$$P_2(x, y) = (x, y) \begin{pmatrix} a_1 & a_2 \\ a_3 & a_4 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + (b_1, b_2) \begin{pmatrix} x \\ y \end{pmatrix} + c,$$

přičemž $\mathrm{A} = \begin{pmatrix} a_1 & a_2 \\ a_3 & a_4 \end{pmatrix}$ je reálná symetrická matice, tedy $a_2 = a_3$, a také koeficienty b_1, b_2, $c \in \mathbb{R}$. Ekvivalentně můžeme rovněž psát

$$P_2(x, y) = a_1 x^2 + a_4 y^2 + 2a_2 xy + b_1 x + b_2 y + c.$$

Nejprve sestavíme funkci $\rho^2(f, P_2)$ danou vztahem (56), jež v tomto případě má tvar

$$\rho^2(f, P_2) = \sum_{i=0}^{9} 1 \cdot \left[f(x_i, y_i) - (a_1 x_i^2 + a_4 y_i^2 + 2a_2 x_i y_i + b_1 x_i + b_2 y_i + c) \right]^2.$$

Na funkci $\rho^2(f, P_2)$ se můžeme dívat jako na funkci šesti proměnných (a_1, a_2, a_4, b_1, b_2, c) a budeme ji chtít minimalizovat. Za tímto účelem použijeme Newtonovu optimalizační metodu. V Matlabu si napíšeme skript:

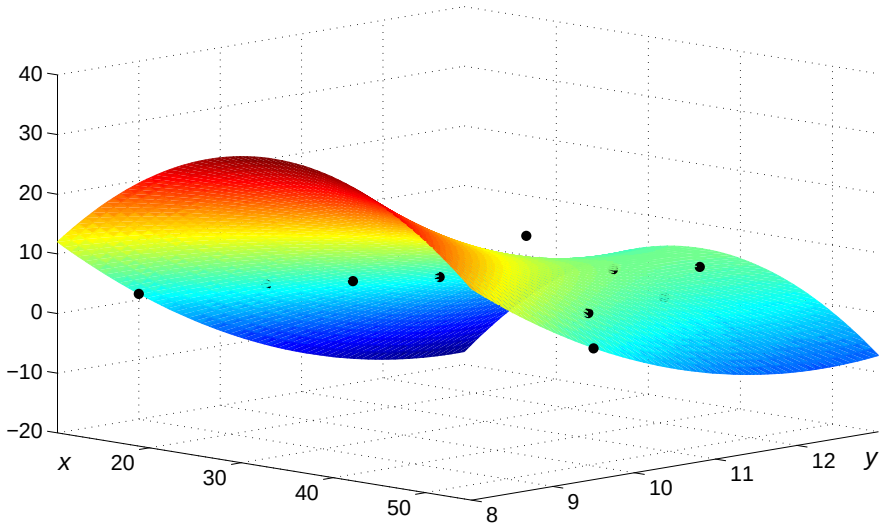
```
xx = [10:5:55];
yy = [9,10,10.5,11,11.5,12,12.5,11.5,10,9.5];
zz = [1,1.5,2.2,3,10,4.5,5,3.5,5.5,2];

prom = sym('[x,y]');
A = sym('[a1,a2;a2,a4]');
b = sym('[b1,b2]');
c = sym('c');
polynom = prom*A*transpose(prom) + b*transpose(prom) + c;

%sestavení funkce, kterou chceme minimalizovat
ro = sym('0');
for (i = 1:10)
    ro = ro + (subs(polynom,prom,[xx(i),yy(i)]) - zz(i)).^2;
end

%hledání optimálních hodnot
[optim,fcni_hodnota,pocet_kroku] = newtonmin(ro,...
    [A(1,1),A(1,2),A(2,2),b,c],[0,0,0,0,0,0],0.0001)
```

Spustíme skript a Matlab nám odpoví:

```
optim =
   -0.0356    0.0160    1.2429    2.1331  -32.3934  171.2770
fcni_hodnota =          pocet_kroku =
   24.7746                  1
```

Tímto jsme zjistili, že aproximační funkce druhého stupně pro data z tabulky 4 má tvar

$$P_2(x,y) = -0{,}0356x^2 + 1{,}2429y^2 + 0{,}0320xy + 2{,}1331x - 32{,}3934y + 171{,}2770.$$



Obrázek 22: Graf aproximačního polynomu $P_2(x, y)$ a bodů $[x_i, y_i, f(x_i, y_i)]$ daných tabulkou 4.

Závěr

V úvodních kapitolách této práce jsme se seznámili se základy teorie optimalizace. Dále jsme se věnovali popisu a analýze Newtonovy optimalizační metody pro úlohu nepodmíněné optimalizace, což považuji za klíčovou část celé práce. Zjistili jsme například, že hlavní výhody Newtonovy optimalizační metody jsou v její rychlosti, univerzálnosti a afinní nezávislosti na zvoleném souřadnicovém systému. Naproti tomu podstatnou nevýhodou této metody je její paměťová náročnost při výpočtu a také důležitost volby počátečního odhadu. V textu jsou uvedeny některé varianty možného vylepšení Newtonovy metody a bylo by jistě zajímavé se touto problematikou dále zabývat.

Pátá kapitola je věnována sebeomezujícím funkcím. Zde po obecném úvodu následují odpovědi na otázky konvergence a volby počátečního odhadu při aplikaci Newtonovy metody na sebeomezující funkce. Pokud se totiž omezíme pouze na sebeomezující funkce, můžeme o konvergenci Newtonovy metody říci více než v obecném případě. Následně je zde uvedena jedna z možností využití sebeomezujících funkcí, a to v roli sebeomezujících bariér, což nám zprostředkovává možnost řešit i úlohy podmíněné optimalizace.

Práci rámcově dotváří nejen názorné obrázky a poutavé příklady, jejichž řešení by se bez použití numerické, zde konkrétně Newtonovy, metody rozhodně neobešlo, ale i původní zdrojové kódy a M-soubory sestavené právě k tomuto účelu.

Zpracovávání tématu zabývajícího se optimalizací mě zavedlo do říše matematické analýzy v akci. Dozvěděl jsem se mnoho zajímavého a řekl bych i užitečného do budoucna. Rád bych se i nadále teorií optimalizace zabýval a to nejen klasickou lokální numerickou optimalizací jako v tomto textu, nýbrž i globální či fuzzy optimalizací. Zejména jako zajímavé problémy se mi jeví využití sebeomezujících funkcí v oblasti podmíněné optimalizace a také aplikace teorie fuzzy množin v optimalizaci.

Literatura

[1] Boyd S., Vandenberghe L.: *Convex Optimization*, Cambridge University Press, Cambridge, 2004

[2] Došlý O.: *Základy konvexní analýzy a optimalizace v* $\mathbb{R}^n$, Masarykova univerzita, Brno, 2005

[3] Fišer J.: *Diskrétní dynamické systémy*, skriptum Univerzity Palackého v elektronické podobě, 2009

[4] Jukl M., *Bilineární a kvadratické formy*, Univerzita Palackého v Olomouci, Olomouc, 2000

[5] Míka S.: *Matematická optimalizace*, Západočeská univerzita, Plzeň, 1997

[6] Nesterov Y., Nemirovski A.: *Interior-Point Polynomial Algorithms in Convex Programming*, SIAM, Philadelphia, 1994

[7] Nocedal J., Wright S. J.: *Numerical Optimization*, Springer, New York, 1999

[8] Phelps R. R.: *Convex Functions, Monotone Operators and Differentiability*, Springer, Berlín ,1993

[9] Rachůnek L., Rachůnková I.: *Diferenciální počet více proměnných*, Univerzita Palackého v Olomouci, Olomouc, 2004