VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA STROJNÍHO INŽENÝRSTVÍ
ÚSTAV MATEMATIKY

FACULTY OF MECHANICAL ENGINEERING
INSTITUTE OF MATHEMATICS

# APROXIMACE VÍCEROZMĚRNÝCH DAT METODOU NEJMENŠÍCH ČTVERCŮ

LEAST-SQUARES FITTING OF MULTIVARIATE DATA

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE / AUTHOR: PAVEL HRABEC

VEDOUCÍ PRÁCE / SUPERVISOR: Ing. PETR TOMÁŠEK, Ph.D.

BRNO 2013

**Abstrakt**
Práce shrnuje základní teorii potřebnou pro odvození aproximace pomocí metody nejmenších čtverců a aplikuje tuto metodu na data z technické praxe. V rámci práce byly vytvořeny programy, v prostředí MATLAB, pro řešení této úlohy. MATLAB už obsahuje několik algoritmů použitelných pro tuto problematiku. Práce zahrnuje porovnání vlastností a testování těchto algoritmů.

**Summary**
Thesis sumarizes basic theory required for inference of aproximation using the least squares method and aplicates this method on data obtained from engineering practice. Within thesis were created programs, in MATLAB enviroment, to solve this problem. MATLAB already contains several algorithms, wich are useful for solving this problem. Thesis includes comparsion of properties and test of those algorithms.

**Klíčová slova**
metoda nejmenších čtverců, QR rozklad, analýza dat

**Keywords**
least square method, QR decompsition, data analysis

HRABEC, P.*Aproximace vícerozměrných dat metodou nejmenších čtverců*. Brno: Vysoké učení technické v Brně, Fakulta strojního inženýrství, 2013. 30 s. Vedoucí Ing. Petr Tomášek, Ph.D.

Prohlašuji, že jsem bakalářskou práci na téma Aproximace vícerozměrných dat metodou nejmenších čtverců vypracoval samostatně pod vedením Ing. Petra Tomáška, Ph.D. Použitou literaturu a další podklady uvádím v seznamu použité literatury.

Pavel Hrabec

Rád bych zde poděkoval všem co se mi s prací snažili pomoci, zejména pak vedoucímu této bakalářské práce Ing. Petru Tomáškovi, Ph.D., konzultantce RNDr. Blance Šedivé, Ph.D. a Ing. Josefu Poláčkovi za jejich rady a čas, který mi věnovali při řešení této problematiky.

Pavel Hrabec

# Obsah

# 1. Úvod

V rámci sítě AMathNet byla navázána spolupráce mezi Ústavem matematiky Fakulty strojního inženýrství VUT v Brně a firmou UNIS a.s. Po konzultaci s výzkumnými pracovníky této firmy vyplynulo tématické zaměření této bakalářské práce.

Cílem bakalářské práce bylo nastudovat potřebnou teorii a algoritmy pro výpočet aproximace funkce metodou nejmenších čtverců, poté tyto poznatky využít při implementaci této problematiky do MATLABu a pomocí vytvořených programů aproximovat dodaná data.

V kapitole druhé jsou uvedeny základní pojmy, které úzce souvisejí s realizovanou analýzou.V kapitole třetí je uvedena obecná formulace úlohy pro proložení dat metodou nejmenších čtverců. O možnostech řešení takovéto úlohy v prostředí MATLAB pojednává kapitola čtvrtá. Stěžejní částí práce je kapitola pátá, kde je popsáno zpracování dodaných dat metodou nejmenších čtverců právě v MATLABu. Závěrečná šestá kapitola shrnuje dosažené výsledky a výstupy bakalářské práce. Práce je doplněna několika přílohami, a sice zdrojovými kódy vytvořených programů, tabulkou zpracovávaných dat a CD s programy.

# 2. Základní pojmy

Dříve než přistoupíme k formulaci a řešení úlohy proložení dat metodou nejmenších čtverců, uvedeme několik zásadních pojmů, které jsou nezbytné pro srozumitelnou formulaci této metody. Uvedené pojmy byly převzaty z [1], [2], [3], [6] a [7].

**Aproximace:** *Aproximací* budeme rozumět nahrazení funkce $b(\mathbf{x})$, $\mathbf{x} \in \Omega \subseteq \mathbb{R}^n$ pomocí funkce $\varphi(\mathbf{x})$, která je jí v jistém smyslu blízká. Potom píšeme $b(\mathbf{x}) \approx \varphi(\mathbf{x})$.

**Definice 1 (metrický prostor)** Metrickým prostorem *budeme nazývat libovolnou množinu* $\mathbf{X}$*, jejíž prvky nazýváme body, pokud je na množině* $\mathbf{X}$ *dána tvz.* metrika*, což je jakákoliv, jednoznačná nezáporná reálná funkce* $\varrho(x, y)$*, která je definovaná pro každou dvojici* $x, y \in \mathbf{X}$ *a která splňuje tyto tři podmínky:*
*1.* $\varrho(x, y) = 0 \Leftrightarrow x = y$,
*2.* $\varrho(x, y) = \varrho(y, x)$*, pro všechna* $x, y \in \mathbf{X}$ *(symetrie),*
*3.* $\varrho(x, y) + \varrho(y, z) \geq \varrho(x, z)$ *pro všechna* $x, y, z \in \mathbf{X}$ *(trojúhelníková nerovnost).*

**Příklad 1** *Příkladem metrického prostoru je množina všech uspořádaných dvojic reálných čísel* $x = [x_1, x_2] \in \mathbb{R}^2$ *s eukleidovskou metrikou ve tvaru*

$$\varrho(x, y) = \sqrt{(y_1 - x_1)^2 + (y_2 - x_2)^2}$$

**Definice 2 (vektorový prostor)** *Nechť* $\mathbf{V}$ *je neprázdná množina prvků* $x, y, z, \ldots$ *a nechť je splněno těchto osm podmínek:*

*I. Ke každým dvěma prvkům* $x, y \in \mathbf{V}$ *je jednoznačně přiřazen třetí prvek ležící ve* $\mathbf{V}$*, který je nazývaný jejich součet a označovaný* $x + y$*, přičemž platí tyto čtyři axiomy:*
*1.* $x + y = y + x$ *(komutativnost),*
*2.* $x + (y + z) = (x + y) + z$ *(asociativita),*
*3. ve* $\mathbf{V}$ *existuje takový prvek (označme jej* $o$*), že* $x + o = x$ *pro všechny prvky* $x \in \mathbf{V}$ *(existence nulového prvku),*
*4. ke každému* $x \in \mathbf{V}$ *existuje prvek, který značíme* $-x$*, že* $x + (-x) = o$ *(existence opačného prvku).*

*II. Ke každému číslu* $\alpha$ *nějakého číselného tělesa* $\mathbf{T}$ *a ke každému prvku* $x \in \mathbf{V}$ *je jednoznačně přiřazen prvek* $\alpha x \in \mathbf{V}$ *(tvz. součin prvku* $x$ *a čísla* $\alpha$*), přičemž platí tyto dva axiomy:*
*1.* $\alpha(\beta x) = (\alpha\beta)x$, $\alpha, \beta \in \mathbf{T}$, $x \in \mathbf{V}$,
*2.* $1 \cdot x = x$, $1 \in \mathbf{T}$, $x \in \mathbf{V}$.

*III. Obě operace (tj. sčítání prvků a násobení prvku číslem) jsou svázány těmito dvěma distributivními zákony:*
*1.* $(\alpha + \beta)x = \alpha x + \beta x$, $\alpha, \beta \in \mathbf{T}$, $x \in \mathbf{V}$,
*2.* $\alpha(x + y) = \alpha x + \alpha y$, $\alpha \in \mathbf{T}$, $x \in \mathbf{V}^{\cdot}$
*Množinu* $\mathbf{V}$ *potom nazýváme vektorovým prostorem nad číselným tělesem* $\mathbf{T}$*. Podle toho*

*zda tělesem* $\mathbf{T}$ *rozumíme reálná nebo komplexní čísla, mluvíme o reálném nebo komplexním vektorovém prostoru. Dále budou naše úvahy platit pro reálné vektorové prostory.*

**Příklad 2** *Axiomy vektorového prostoru splňuje například množina všech uspořádaných* $n$*-tic reálných čísel, kde sčítání a násobení konstantou jsou definovány následovně*

$$(x_1, x_2, \ldots, x_n) + (y_1, y_2, \ldots, y_n) = (x_1 + y_1, x_2 + y_2, \ldots, x_n + y_n)$$
$$\alpha(x_1, x_2, \ldots, x_n) = (\alpha x_1, \alpha x_2, \ldots, \alpha x_n),\ \alpha \in \mathbb{R}$$

**Definice 3 (skalární součin)** Skalárním součinem *ve vektorové prostoru* $\mathbf{V}$ *nazýváme reálnou funkci* $(x, y)$*, která je definována pro každou dvojici prvků* $x, y \in \mathbf{V}$ *a splňuje tyto podmínky* ($x, y, z \in \mathbf{V}$, $\lambda \in \mathbb{R}$)*:*
*1.* $(x, y) = (y, x)$,
*2.* $(x + z, y) = (x, y) + (z, y)$,
*3.* $(\lambda x, y) = \lambda(x, y)$,
*4.* $(x, x) \geq 0$, *přičemž* $(x, x) = 0 \Leftrightarrow x = o$.

*Na vektorových prostorech se skalárním součinem zavádíme normu jako*

$$\|x\| = \sqrt{(x, x)}.$$

**Příklad 3** *Na vektorovém prostoru z příkladu 2 lze zavést například skalární součin*

$$(x, y) = (x_1 y_1 + x_2 y_2 + \cdots + x_n y_n) = \sum_{i=1}^{n} x_i y_i.$$

*Tento zřejmě vyhovuje podmínkám uvedeným v definici 3*

**Definice 4 (normovaný vektorový prostor)** *Vektorový prostor* $\mathbf{V}$ *se nazývá normovaný, jestliže každému prvku* $x \in \mathbf{V}$ *je přiřazeno reálné nezáporné číslo* $\|x\|$*, které se nazývá norma prvku* $x$*, přičemž platí:*
*1.* $\|x\| = 0 \Leftrightarrow x = o$,
*2.* $\|\alpha x\| = \alpha \cdot \|x\|$,
*3.* $\|x + y\| \leq \|x\| + \|y\|$,

**Příklad 4** *Pokud zavedeme na uspořádaných* $n$*-ticích reálných čísel s operacemi dle příkladu 2 eukleidovskou normu prostřednictvím skalárního součinu z definice 3*

$$\|x\| = \sqrt{\sum_{i=1}^{n} (x_i)^2}$$

*dostaneme normovaný vektorový prostor.*

Dále v textu má značení $\|x\|$ význam právě eukleidovské normy.

**Definice 5 (lineární kombinace)** *Nechť* $\mathbf{V} = (V, +)$ *je vektorový prostor a* $\mathbf{v}_1, \dots, \mathbf{v}_k \in V$, $k \in \mathbf{N}$, *kde* $\mathbf{N}$ *je množina přirozených čísel. Řekneme, že vektor* $\mathbf{v}$ *je* lineární kombinací *vektorů* $\mathbf{v}_1, \dots, \mathbf{v}_k$, *jestliže existují reálná čísla* $r_1, \dots, r_k$ *tak, že platí:*

$$\mathbf{v} = r_1 \mathbf{v}_1 + \cdots + r_k \mathbf{v}_k$$

**Definice 6 (lineární závislost)** *Prvky* $x, y, \dots, w$ *vektorového prostoru se nazývají* lineárně závislé, *jestliže existují takové konstanty* $\alpha, \beta, \dots, \lambda$, *z nichž alespoň jedna je různá od nuly, že*

$$\alpha x + \beta y + \cdots + \lambda w = o$$

*V opačném případě se prvky nazývají lineárně nezávislé.*

**Definice 7 (báze vektorového prostoru)** *Nechť* $\mathbf{V} = (V, +)$ *je vektorový prostor a* $\mathbf{v}_1, \dots, \mathbf{v}_k \in V (k \in \mathbb{N})$. *Soustava vektorů* $\mathbf{v}_1, \dots, \mathbf{v}_k$, *se nazývá* báze vektorového prostoru $\mathbf{V}$, *jestliže vektory* $\mathbf{v}_1, \dots, \mathbf{v}_k$ *jsou lineárně nezávislé a každý vektor* $\mathbf{v} \in \mathbf{V}$ *je lineární kombinací* $\mathbf{v}_1, \dots, \mathbf{v}_k$.

**Definice 8 (ortonormální matice)** Ortonormální matice*je reálná čtvercová matice* $\mathbf{Q}$ *jejíž transpozice je současně její inverzí.*

$$\mathbf{Q}^T = \mathbf{Q}^{-1}$$

Další pojmy a výsledky úzce související s tématem lze najít v [1], [2], [3], [6] a [7].

# 3. Formulace úlohy

Velká část formulace úlohy aproximace dat metodou nejmenších čtverců (MNČ) je převzata z [2]. Naším cílem je s pomocí $k$ měření závislosti $b(\mathbf{x})$, kde $\mathbf{x} = (x_1, x_2, \ldots, x_m)$ nahradit $b(\mathbf{x})$ pomocí soustavy $n$ lineárně nezávislých funkcí $\varphi_i(\mathbf{x})$, $n < k$. Tato soustava tvoří bázi vektorového prostoru, proto jsou dále nazývány jako bázové funkce.

$$b(\mathbf{x}) \approx f_1\varphi_1(\mathbf{x}) + f_2\varphi_2(\mathbf{x}) + \cdots + f_n\varphi_n(\mathbf{x})$$

Bázové funkce volíme podle předpokládaného průběhu neznámé funkce $b(\mathbf{x})$. Návrhová matice $\mathbf{A}$ vznikne dosazením hodnot bázových funkcí v bodech měření. $\mathbf{A}$ je matice typu $k \times n$.

$$\mathbf{A} = \begin{pmatrix} \varphi_1(\mathbf{x}_1) & \varphi_2(\mathbf{x}_1) & \ldots & \varphi_n(\mathbf{x}_1) \\ \varphi_1(\mathbf{x}_2) & \varphi_2(\mathbf{x}_2) & \ldots & \varphi_n(\mathbf{x}_2) \\ \vdots & \vdots & & \vdots \\ \varphi_1(\mathbf{x}_k) & \varphi_2(\mathbf{x}_k) & \ldots & \varphi_n(\mathbf{x}_k) \end{pmatrix}.$$

Koeficienty $f_1, \ldots, f_n$ hledáme jako řešení soustavy rovnic

$$\mathbf{A}\mathbf{f} = \mathbf{b}. \tag{3.1}$$

$\mathbf{b}$ je vektor naměřených hodnot v daných bodech

$$\mathbf{b} = (b(\mathbf{x}_1), b(\mathbf{x}_2), \ldots, b(\mathbf{x}_k))^T.$$

Tato soustava je přeurčená a nemá řešení, budeme se tedy snažit alespoň minimalizovat rezidua $\mathbf{r} = \mathbf{b} - \mathbf{A}\mathbf{f}$. Při řešení pomocí MNČ minimalizujeme součet čtverců reziduí, což se zároveň rovná druhé mocnině eukleidovské normy vektoru $\mathbf{r}$:

$$\|\mathbf{r}\|^2 = \sum_{i=1}^{k} r_i^2 \to \text{ min.}$$

V dalším předpokládejme, že bázové funkce $\varphi_i(\mathbf{x})$ nejsou lineárně závislé, tedy ani sloupce matice $\mathbf{A}$ nejsou lineárně závislé a matice $\mathbf{A}$ má plnou sloupcovou hodnost.

## 3.1. Normální rovnice

Normální rovnice je asi nejjednodušší způsob řešení MNČ uváděný ve spoustě učebnic mimo jiné [2] zejména asi proto, že je jednoduché je odvodit například z nutné podmínky pro extrém součtu čtverců reziduí

$$\frac{\partial \|r\|^2}{\partial f_l} = \frac{\partial}{\partial f_l} \sum_{i=1}^{k} (b_i - \sum_{j=1}^{n} a_{ij} f_j)^2 = 0 \text{ pro } l = 1, 2 \ldots, n.$$

Po zderivování dostáváme

$$\frac{\partial\|r\|^2}{\partial f_l} = 2\sum_{i=1}^{k}(b_i - \sum_{j=1}^{n} a_{ij}f_j)(-a_{il}) = 0$$

a po úpravě vyjde

$$\sum_{j=1}^{n}(\sum_{i=1}^{k} a_{il}a_{ij})f_j = \sum_{i=1}^{k} a_{il}b_i \text{ pro } l = 1, 2, \ldots, n$$

což lze maticově zapsat jako

$$\mathbf{A}^T\mathbf{A}\mathbf{f} = \mathbf{A}^T\mathbf{b}$$

a právě této soustavě říkáme normální rovnice. Další možné odvození je pomocí projekce vektoru **b** do prostoru tvořeného sloupci matice **A**. V případě, že sloupce matice **A** jsou lineárně nezávislé mají normální rovnice právě jedno řešení, soustava už není přeurčená a je možné ji řešit jakýmkoli algoritmem pro řešení soustav lineárních rovnic. Nevýhodou tohoto postupu je, že tato úloha je často špatně podmíněná, tudíž numerické řešení je dosti nepřesné a nejspíš právě proto v MATLABu není řešení MNČ pomocí normálních rovnic implementováno a je nahrazeno numericky stabilnějšími způsoby popsanými v textu dále.

# 4. Metody řešení MNČ v prostředí MATLAB

V této kapitole popisuji metody řešení problému MNČ v prostředí MATLAB a rozdíly mezi nimi. Testy jsem prováděl na notebooku HP ProBook 4530s s procesorem Intel(R) Pentium(R) CPU B950 @ 2.10 GHz, 4GB paměti RAM a 64 bitovým operačním systémem Windows 7 pro matici $\mathbf{A}$ typu $63 \times 15$, protože právě taková byla vytvořena v rámci realizace zpracování dat.

## 4.1. Operátor \

Operátor \je vestavěnou funkcí MATLABU pro řešení soustav lineárních rovnic. Pokud mu však zadáme přeurčenou soustavu najde její řešení ve smyslu MNČ. K řešení přeurčených soustav používá QR rozklad. QR rozklad rozloží matici $\mathbf{A}$ na součin matic $\mathbf{Q}$ a $\mathbf{R}$, kde $\mathbf{Q}$ je ortonormální matice řádu $k$ a $\mathbf{R}$ je horní trojúhelníková matice typu $k \times n$.

$$\mathbf{A} = \mathbf{QR}.$$

Protože platí:

$$\| \mathbf{r} \|^2 = \mathbf{r}^T\mathbf{r} = \mathbf{r}^T\mathbf{Q}\mathbf{Q}^T\mathbf{r} = (\mathbf{Q}^T\mathbf{r})^T\mathbf{Q}^T\mathbf{r} = \| \mathbf{Q}^T\mathbf{r} \|^2$$

je minimalizace $\| \mathbf{r} \|^2$ ekvivalentní minimalizaci $\| \mathbf{Q}^T\mathbf{r} \|^2$. Po provedení QR rozkladu a vynásobením rovnice (3.1) zleva $\mathbf{Q}^T$ dostaneme:

$$\mathbf{Rf} = \mathbf{Q}^T\mathbf{b}.$$

Označme

$$\mathbf{R} = \begin{pmatrix} \mathbf{R}_1 \\ \mathbf{R}_2 \end{pmatrix}, \mathbf{d} = \mathbf{Q}^T\mathbf{b} = \begin{pmatrix} \mathbf{d}_1 \\ \mathbf{d}_2 \end{pmatrix}$$

kde $\mathbf{R}_1$ je čtvercová matice řádu $n$, $\mathbf{R}_2$ je nulová matice typu $(k-n) \times n$, $\mathbf{d}_1$ je sloupcový vektor o $n$ prvcích a $\mathbf{d}_2$ je sloupcový vektor o $k-n$ prvcích. Pak

$$\| \mathbf{Q}^T\mathbf{r} \| = \left\| \begin{matrix} \mathbf{R}_1\mathbf{f} - \mathbf{d}_1 \\ \mathbf{R}_2\mathbf{f} - \mathbf{d}_2 \end{matrix} \right\| = \left\| \begin{matrix} \mathbf{R}_1\mathbf{f} - \mathbf{d}_1 \\ -\mathbf{d}_2 \end{matrix} \right\|$$

zřejmě nabývá minima pokud $\mathbf{f}$ splňuje $\mathbf{R}_1\mathbf{f} = \mathbf{d}_1$ a $\min \| \mathbf{r} \| = \| \mathbf{d}_2 \|$.
Protože $\mathbf{A}$ má plnou sloupcovou hodnost je $\mathbf{R}_1$ regulární a rovnice $\mathbf{R}_1\mathbf{f} = \mathbf{d}_1$ má právě jedno řešení.

Některé algoritmy QR rozkladu najdete například v [1].
Syntaxe v MATLABu: $\mathbf{f} = \mathbf{A}\backslash\mathbf{b}$. Podle mojí zkoušky pro matici soustavy $\mathbf{A}$ plné sloupcové hodnosti typu $63 \times 15$ je tato metoda nejrychlejší z uvedených, a tudíž nejméně výpočetně náročná. Doba trvání výpočtu byla podle MATLABu $0,000106$ sekund.

## 4.2. Funkce pinv

Funkce pinv($\mathbf{A}$) spočítá Mooreovu-Penroseovu pseudoinverzi matice $\mathbf{A}$. Pseudoinverzní matice $\mathbf{A}^+$ se definuje jako jediná matice $\mathbf{X}$ typu $k \times n$, která splňuje tyto čtyři podmínky:

**(I)** $\mathbf{AXA} = \mathbf{A}$,

**(II)** $\mathbf{XAX} = \mathbf{X}$,

**(III)** $(\mathbf{AX})^T = \mathbf{AX}$,

**(IV)** $(\mathbf{XA})^T = \mathbf{XA}$.

Po vynásobení rovnice (3.1) maticí $\mathbf{A}^+$ zleva dostaneme

$$\mathbf{f} = \mathbf{A}^+\mathbf{b}.$$

V MATLABu se pseudoinverze počítá pomocí singulárního rozkladu. Singulární rozklad rozloží matici $\mathbf{A}$ na součin matic $\mathbf{U\Sigma V}^T$, kde $\mathbf{U}$ je ortonormální matice řádu $k$, $\mathbf{V}$ je ortonormální matice řádu $n$ a $\mathbf{\Sigma}$ je diagonální matice typu $k \times n$. Pseudoinverze $\mathbf{A}^+$ je potom ve tvaru:

$$\mathbf{A}^+ = \mathbf{V\Sigma}^+\mathbf{U}^T$$

kde $\mathbf{\Sigma}^+$ je matice typu $n \times k$ a na diagonále jsou převrácené hodnoty diagonály matice $\mathbf{\Sigma}$.
Syntaxe v MATLABu: $\mathbf{f}$ =pinv($\mathbf{A}$)$*\mathbf{b}$
Podle mojí zkoušky pro matici soustavy $\mathbf{A}$ plné sloupcové hodnosti typu $63 \times 15$ je tato metoda nejpomalejší z uvedených, a tudíž nejvíce výpočetně náročná. Doba trvání tohoto výpočtu byla podle MATLABu $0,011725$ sekund. Rozdíl oproti předchozímu způsobu řešení se ale projeví pouze pokud matice $\mathbf{A}$ nemá plnou sloupcovou hodnost. Více o výpočtu řešení MNČ pro matice, které nemají plnou sloupcovou hodnost, lze najít například v [1] nebo v [5].

## 4.3. Funkce lscov

Funkce lscov počítá řešení MNČ stejnou metodou jako \, tedy pomocí QR rozkladu, ale navíc umožňuje mimo jiné zadat váhu pro jednotlivá měření (pokud nejsou stejné). Dostaneme pak váženou MNČ. Váženou MNČ je vhodné použít, pokud některá měření jsou přesnější než jiná, nebo naopak. Při použití vážené MNČ zvolíme vektor $\mathbf{w}$ o $k$ prvcích, jehož každý prvek $w_i > 0$ a minimalizujeme

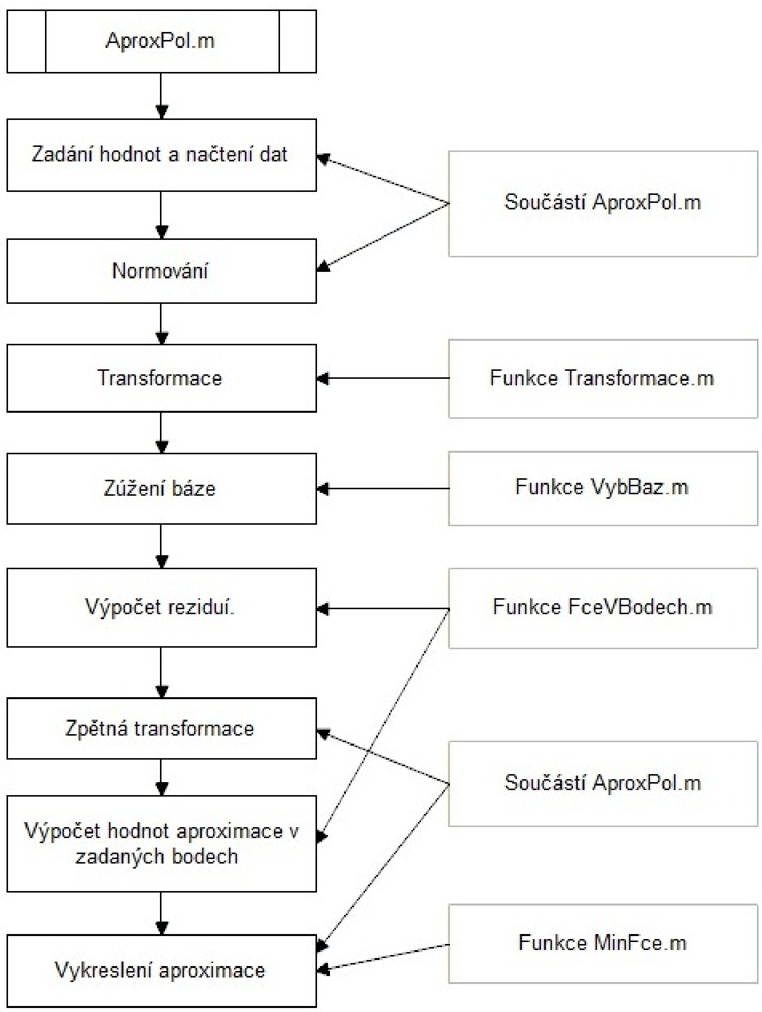
$$\sum_{i=1}^{k} w_i r_i^2 \to \text{ min.}$$

O dalším využití lscov se dozvíte v [4].
Syntaxe v MATLABu: $\mathbf{f}$ =lscov($\mathbf{A}, \mathbf{b}$), případně pro váženou MNČ: $\mathbf{f}$ =lscov($\mathbf{A}, \mathbf{b}, \mathbf{w}$).
Podle mojí zkoušky pro matici soustavy $\mathbf{A}$ plné sloupcové hodnosti typu $63 \times 15$ je tato metoda druhá nejrychlejší z uvedených, přesto byla ve skriptu použita právě pro další možnosti, které tato funkce uživateli nabízí, zejména možnost použití vážené MNČ. Doba trvání tohoto výpočtu byla podle MATLABu $0,000319$ sekund.

# 5. Realizace zpracování dat

Tato kapitola popisuje implementaci úlohy aproximace dvourozměrných dat polynomem pomocí MNČ do MATLABu. Data k tomuto příkladu byla dodána firmou UNIS a.s. Jedná se o závislost parametru účinnosti dmychadla na $b$ redukovaném hmotnostním průtoku vzduchu $x_1$ a redukovaných otáčkách $x_2$. Strukturu výpočtu popisuje diagram obr.1. Více k jednotlivým krokům v příslušných podkapitolách. Výsledky jsou pro aproximaci polynomem čtvrtého stupně

$$
\begin{aligned}
b(x_1,x_2) \approx f_1 + f_2x_2 + f_3x_2^2 + f_4x_2^3 + f_5x_2^4 + f_6x_1 + f_7x_1x_2 + f_8x_1x_2^2 + f_9x_1x_2^3 + \\
+ f_{10}x_1^2 + f_{11}x_1^2x_2 + f_{12}x_1^2x_2^2 + f_{13}x_1^3 + f_{14}x_1^3x_2 + f_{15}x_1^4.
\end{aligned}
$$




obr.1: Struktura výpočtu

## 5.1. Načtení dat a zadání parametrů

Data byla dodána formou tabulky pro MS Excel, kterou naleznete v příloze. Do MATLABu se načtou pomocí příkazu xlsread. První sloupec tabulky je označen $\mathbf{x_1}$, druhý sloupec $\mathbf{x_2}$ a třetí sloupec $\mathbf{b}$. Dále je třeba zadat nejvyšší přípustný stupeň polynomu, kterým chceme aproximovat, významné body, ve kterých chceme znát hodnotu aproximace a pokud chceme počítat váženou MNČ, tak i vektor vah $\mathbf{w}$.

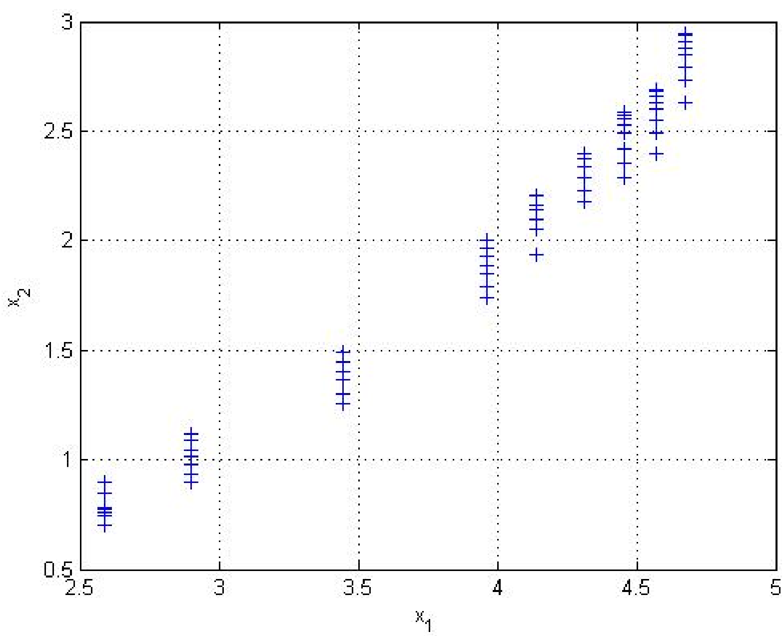
## 5.2. Normování

Data je vhodné normovat, tak aby byla alespoň stejného řádu. Děláme to pro zvýšení přesnosti numerických výpočtů, ale hlavně proto, že při výpočtu hodnot členů matice $\mathbf{A}$ mohou být některé její členy jako mocniny složek vektorů $\mathbf{x_1}, \mathbf{x_2}$ neúnosně velké nebo naopak příliš blízké nule, tudíž matice je blízko ke ztrátě své plné sloupcové hodnosti a pro strojovou přesnost sloupcovou hodnost ztrácí. Zde byla data $\mathbf{x_1}$ podělena 10000.

## 5.3. Transformace

Na obrázku obr.2 jsou zobrazeny data $\mathbf{x_1}, \mathbf{x_2}$. V tomto případě je vhodné transformovat souřadnice dat $\mathbf{x_1}, \mathbf{x_2}$ z důvodu snížení vlivu závislosti těchto dvou proměnných.

Nový souřadný systém se určuje pomocí jednorozměrné MNČ. Nejprve proloží přímku daty $\mathbf{x_1}, \mathbf{x_2}$ a poté souřadnice otočí tak, aby nová osa $x_1$ byla rovnoběžná s vypočtenou přímkou a nová osa $x_2$ na ni byla kolmá. Následuje posun v každém směru o průměr otočených souřadnic, aby se po transformaci nacházely naměřené body v okolí nuly.




obr.2: $\mathbf{x_1}, \mathbf{x_2}$ před transformací

K tomuto účelu jsem vytvořil funkci Transformace.m, která z hodnot dat $\mathbf{x_1}$ a $\mathbf{x_2}$ spočítá matici přechodu $\mathbf{TR}$. Funkce nejprve napočítá koeficienty přímky $x_2 = \hat{f}_1 + \hat{f}_2 x_1$. Pro zadané hodnoty

$$\hat{f}_1 = -1,7838,\ \hat{f}_2 = 0,9549.$$

Poté spočítá potřebnou matici rotace

$$\mathbf{R} = \begin{pmatrix} \cos(\arctan(-\hat{f}_2)) & -\sin(\arctan(-\hat{f}_2)) & 0 \\ \sin(\arctan(-\hat{f}_2)) & \cos(\arctan(-\hat{f}_2)) & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

Pro tuto úlohu tedy obdržíme

$$\mathbf{R} = \begin{pmatrix} 0,723 & -0,6906 & 0 \\ 0,6906 & 0,723 & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

Pak určíme potřebné posunutí v každém směru jako průměr příslušné otočené souřadnice pomocí funkce mean a dosadíme do matice translace

$$\mathbf{T} = \begin{pmatrix} 1 & 0 & -\text{mean}(\mathbf{x_1}) \\ 0 & 1 & -\text{mean}(\mathbf{x_2}) \\ 0 & 0 & 1 \end{pmatrix}.$$

V našem případě je

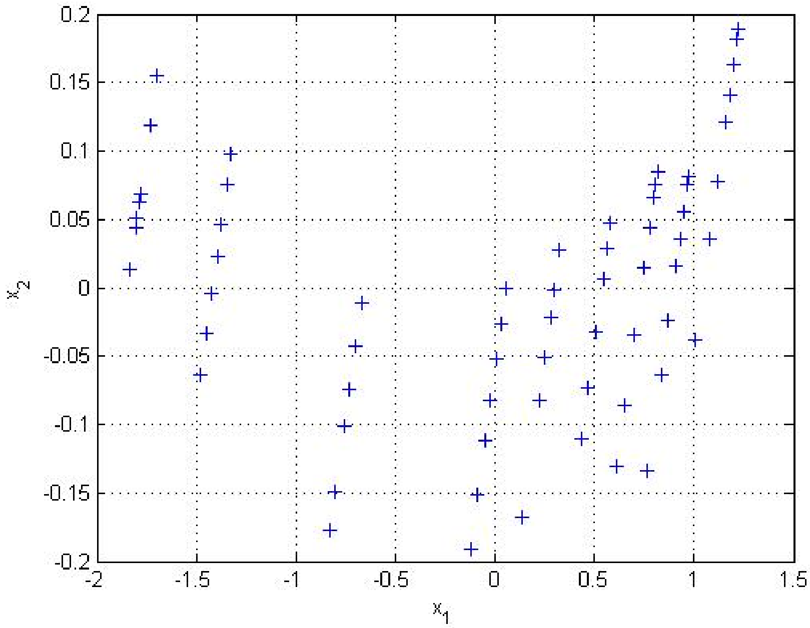
$$\mathbf{T} = \begin{pmatrix} 1 & 0 & 3,9226 \\ 0 & 1 & 1,9618 \\ 0 & 0 & 1 \end{pmatrix}.$$

Výsledná matice přechodu je potom součinem matice translace $\mathbf{T}$ a matice rotace $\mathbf{R}$ nutně v tomto pořadí, protože násobení matic není komutativní. Naše matice přechodu tedy bude

$$\mathbf{TR} = \begin{pmatrix} \cos(\arctan(-\hat{f}_2)) & -\sin(\arctan(-\hat{f}_2)) & -\text{mean}(\mathbf{x_1}) \\ \sin(\arctan(-\hat{f}_2)) & \cos(\arctan(-\hat{f}_2)) & -\text{mean}(\mathbf{x_2}) \\ 0 & 0 & 1 \end{pmatrix} =$$

$$= \begin{pmatrix} 0,723 & -0,6906 & 3,9226 \\ 0,6906 & 0,723 & 1,9618 \\ 0 & 0 & 1 \end{pmatrix}.$$

Samotná transformace se pak provádí až ve skriptu, kde se do anonymních funkcí $xt_1$ a $xt_2$(zavedených v MATLABu operátorem @), podle matice přechodu zapíše předpis pro nové souřadnice a pak se do něj dosadí. Další výpočty probíhají v transformovaných souřadnicích. Data $\mathbf{x_1}, \mathbf{x_2}$ po transformaci jsou na obrázku obr.3.




obr.3: $\mathbf{x_1}, \mathbf{x_2}$ po transformaci

## 5.4. Výpočet pro nejširší možnou bázi

Uživatel by měl zadat nejvyšší přípustný stupeň polynomu, kterým budeme danou závislost aproximovat. Je třeba ale mít na paměti, kolik měření máme k dispozici, abychom se nedostali do situace, že soustava rovnic, kterou řešíme, není přeurčená. Pak bychom měli více neznámých než rovnic, a tudíž nekonečně mnoho řešení, nebo stejně rovnic jako neznámých a jednalo by se o interpolaci.

Zvolíme tedy nejvyšší možný stupeň aproximačního polynomu. Skript vytvoří anonymní funkci pro každý člen polynomu a pomocí příkazu cell je seřadí do vektoru bázových funkcí $\varphi$ (v AproxPol.m je značen `phi`). Poté pomocí tohoto vektoru $\varphi$ spočítá návrhovou matici $\mathbf{A}$ a pomocí příkazu lscov vyřeší soustavu rovnic $\mathbf{Af} = \mathbf{b}$ ve smyslu MNČ. Skript pokračuje napočítáním vektoru $\mathbf{G}$, který je vektorem hodnot aproximace ve všech zadaných bodech. Vektor $\mathbf{G}$ počítá vytvořená funkce FceVBodech.m. Její vstupní parametry jsou bázové funkce $\varphi$, vypočtené koeficienty $\mathbf{f}$ a hodnoty $\mathbf{x_1}, \mathbf{x_2}$. FceVBodech.m potom pouze hodnoty $\mathbf{x_1}, \mathbf{x_2}$ dosadí do získaného předpisu $f_1\varphi_1 + f_2\varphi_2 + \ldots + f_n\varphi_n$. Rozdílem vektoru naměřených funkčních hodnot $\mathbf{b}$ a vektoru $\mathbf{G}$ je vektor reziduí $\mathbf{r}$.

## 5.5. Pokus o zúžení báze

Ne vždy musí být nejlepší aproximací polynom zvoleného stupně. Například pokud se snažíme aproximovat konstantní funkci $b(x_1, x_2) = 2$ něčím jiným než konstantou ostatní koeficienty vyjdou téměř nulové, ale různé od nuly, díky čemuž je součet čtverců větší než při aproximaci jen konstantou.

Proto jsem vytvořil funkci VybBaz.m, která se pokusí vyloučit ze současné báze některé členy tak, že napočítá aproximaci postupně bez každého členu báze a pokud je nový součet čtverců $r2sumn$ menší než původní $r2sum$, je tento člen vyřazen. Tento proces se opakuje dokud se báze zmenšuje. Výstupem této funkce je tedy báze ve tvaru polynomu nejvýše zadaného stupně s nejmenším součtem čtverců reziduí.

## 5.6. Výpočet reziduí

Abychom mohli alespoň trochu posoudit vhodnost transformace je nutné spočítat rezidua. Toto provedeme ještě před zpětnou transformací, protože při každé transformaci se dopouštíme určité chyby a zvolená transformace rezidua nijak nemění.

AproxPol.m vypočítá absolutní hodnotu největšího rezidua $rmax$, absolutní hodnotu nejmenšího rezidua $rmin$, součet absolutních hodnot reziduí $rsum$ a součet čtverců reziduí $r2sum$.
Součet čtverců v tomto případě vychází

$$r2sum = 0,0011,$$

absolutní hodnota maximální odchylky

$$rmax = 0,0132$$

a absolutní hodnota minimální odchylky

$$rmin = 6,3030 \cdot 10^{-5}.$$

Součet absolutních hodnot odchylek pak je

$$rsum = 0,1962.$$

Následující tabulka obsahuje získané číselné hodnoty koeficientů $f_i$, $i = 1, \ldots, 15$. Ve druhém řádku jsou uvedeny odpovídající bázové funkce $\varphi_1, \ldots, \varphi_{15}$

| $i$: | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|
| $\varphi_i(x_1, x_2)$: | 1 | $x_2$ | $x_2^2$ | $x_2^3$ | $x_2^4$ | $x_1$ | $x_1x_2$ |
| $f_i$ | $0,4817$ | $-0,3033$ | $-2,2508$ | $-3,2325$ | $-6,8090$ | $0,0267$ | $0.2403$ |

| 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|
| $x_1x_2^2$ | $x_1x_2^3$ | $x_1^2$ | $x_1^2x_2$ | $x_1^2x_2^2$ | $x_1^3$ | $x_1^3x_2$ | $x_1^4$ |
| $0,7416$ | $0,4157$ | $0,0163$ | $0,2105$ | $0,8135$ | $-0,0290$ | $-0,0301$ | $-0,0159$ |

V transformovaných souřadnicích tedy získaná aproximace je

$$b(x_1, x_2) \approx \sum_{i=1}^{15} f_i \varphi_i(x_1, x_2).$$

## 5.7. Zpětná transformace

Pro zpětnou transformaci máme v podstatě vše připraveno. Stačí vnořením anonymních funkcí $xt_1, xt_2$ do bázových funkcí $\varphi_i$ transformovat zpět bázové funkce a poté dosazením hodnot inverzní matice k matici přechodu $\mathbf{TR}^{-1}$ do $xt_1, xt_2$ vrátit souřadnice zpět.

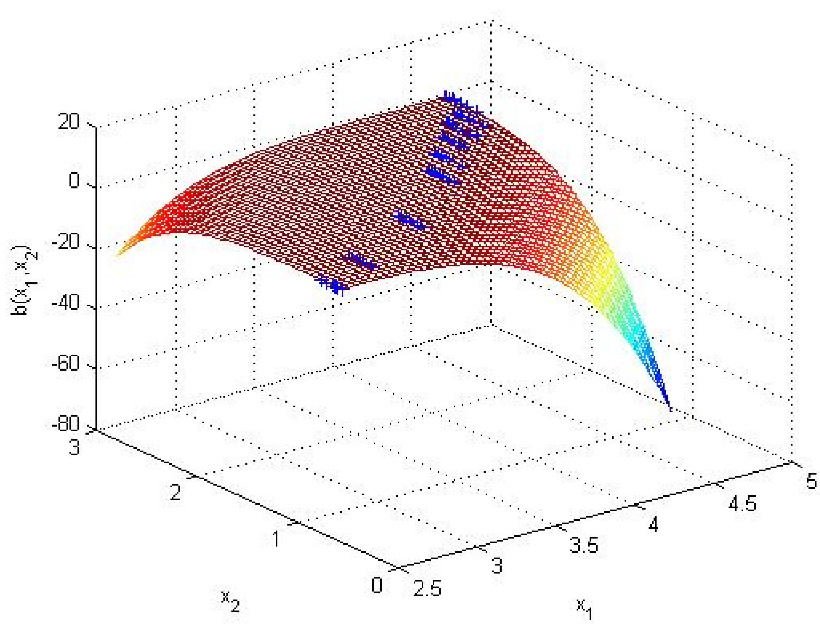
## 5.8. Výpočet aproximace v zadaných bodech

Uživatel může ve skriptu AproxPol.m zadat body, ve kterých chce znát hodnotu aproximace. Tento výpočet probíhá opět v rámci funkce FceVBodech.m. Změní se pouze vstupní parametry se souřadnicemi bodů. Výsledné hodnoty se zapíšou do vektoru **G**. Například pro body $[x_1; x_2] = [30000; 1]$ a $[x_1; x_2] = [32000; 1, 5]$ získáme hodnoty aproximace v daném pořadí $G_1 = 0, 4985$ a $G_2 = 0, 3404$.

## 5.9. Vykreslení aproximace

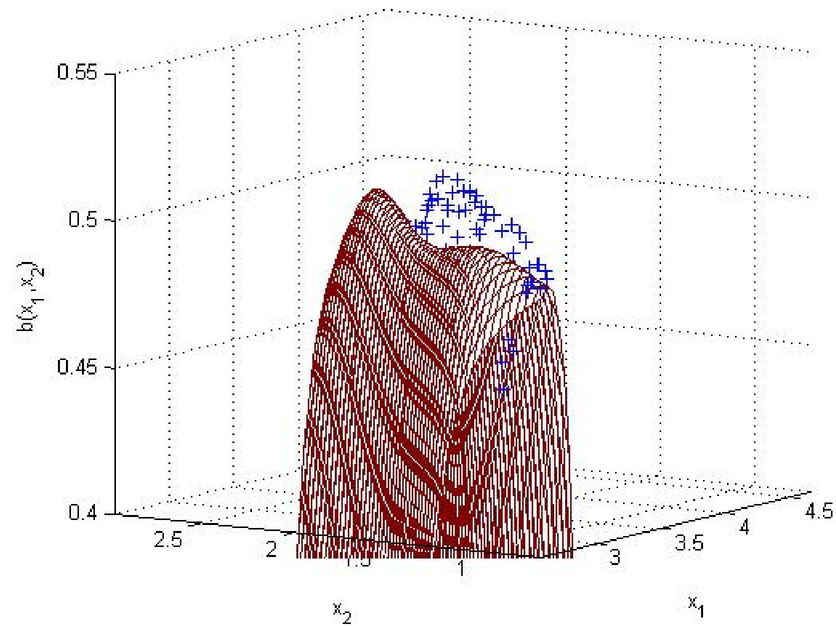
Poslední částí skriptu je vykreslení nalezeného řešení. Skript vykreslí nalezenou plochu, zadané body a nakonec zvlášť pouze zadané body.

Pro vykreslení bodů ve třech dimenzích jsem zvolil příkaz scatter3. Vykreslení aproximační plochy probíhá tak, že se pomocí příkazu linspace nadělí prostor mezi maximem a minimem $\mathbf{x_1}$ a $\mathbf{x_2}$ na zadaný počet dílků. Tím vzniknou vektory $\mathbf{xp_1}$ a $\mathbf{xp_2}$ tvořící síť, na které pomocí funkce MinFce.m napočítáme hodnoty aproximace ve všech bodech sítě. Funkce MinFce.m je velmi podobná funkci FceVBodech.m, dokonce vyžaduje i stejné argumenty, ale na rozdíl od ní počítá hodnoty aproximace ve všech bodech kartézského součinu zadaných vektorů, proto je jejím výstupem čtvercová matice **F**. Vykreslení plochy je realizováno pomocí příkazu mesh a výsledný graf je na obr. 4




obr.4: aproximace funkce $b(x_1, x_2)$ polynomem čtvrtého stupně

Je ale nutné podotknout, že spočítaná aproximace je smysluplná pouze na blízkém okolí měření. Mimo tuto oblast o funkci $b(x_1, x_2)$ nevíme vůbec nic a tedy nelze odhadovat chybu aproximace. Co se týká chyby získané aproximace, lze odhadnout na základě předpokladů kladených na funkci $b(x_1, x_2)$. Pokud graf přiblížíme k zadaným hodnotám, dostaneme obr.5




obr.5: aproximace funkce $b(x_1, x_2)$ polynomem čtvrtého stupně v okolí měření

Tímto je ukončena realizace MNČ v MATLABu. Zdrojové kódy jsou součástí přílohy bakalářské práce. Funkčnost vytvořených programů byla na několika testovacích úlohách prověřena.

# 6. Závěr

V této práci jsme shrnuli základní pojmy potřebné pro formulaci a řešení problému vícerozměrných aproximací.Po formulaci této úlohy jsme analyzovali možnosti řešení této problematiky v MATLABu. Vybrané metody jsme srovnali a na základě této analýzy a testovacích výpočtů jsme vybrané metody implementovali v našem programu AproxPol.m.

Vytvořili jsme skript AproxPol.m, který po dodání měření funkce dvou proměnných tuto funkci aproximuje polynomem zvoleného stupně. Navíc spočítá rezidua a některé jejich parametry.

Velkým přínosem práce pro autora bylo výrazné prohloubení znalostí o možnostech prostředí MATLAB a zejména spolupráce s firmou UNIS a.s. a práce s jejich reálnými daty.

# Literatura

[1] ČERMÁK, Libor. ÚSTAV MATEMATIKY FSI VUT V BRNĚ. *Vybrané statě z numerických metod.* Brno, 2011. Dostupné z: http://www.math.fme.vutbr.cz/default.aspx?catalog=3&catsrtext=30&catsrfield=38

[2] ČERMÁK, Libor a Rudolf HLAVIČKA. *Numerické metody: teoretická část.* Vyd. 2. Brno: Akademické nakladatelství CERM, 2008, 110 s. ISBN 978-80-214-3752-4.

[3] KARÁSEK, Jiří. *Lineární algebra: teoretická část.* 1. vyd.. Brno: CERM, 2005, 179 s. ISBN 80-214-3100-8.

[4] MATLAB online documentation, http://www.mathworks.com/help/matlab/

[5] MOLER, Cleve B. *Numerical computing with MATLAB.* Rev. reprint. Philadelphia: Society for Industrial and Applied Mathematics, 2008. ISBN 978-089-8716-603.

[6] RALSTON, A. *Základy numerické matematiky.* Praha, 1973, 636 s.

[7] ŽENÍŠEK, Alexander. *Lebesgueův integrál a základy funkcionální analýzy.* 1. vyd. Brno: PC-DIR, 1998, 168 s. Učební texty vysokých škol (Vysoké učení technické v Brně). ISBN 80-214-1162-7.

# 7. Seznam použitých zkratek a symbolů

| | |
|---|---|
| MNČ | metoda nejmenších čtverců |
| $\|x\|^2$ | druhá mocnina eukleidovské normy $x$ |
| $\mathbf{A}^T$ | transpozice $A$ |
| $\mathbf{A}^+$ | pseudoinverze matice $A$ |

# 8. Přilohy

## 8.1. Zdrojový kód AproxPol.m

```
%Skript AproxPol aproximuje měření závislosti dvou proměnných polynomem nejvýše
%zvoleného stupně k metodou nejmenších čtverců. Spočítá hodnoty aproximace
%v zadaných bodech (vektor G), rezidua (vektor r), součet čtverců reziduí (r2sum)
%největší odchylku (rmax), nejmenší odchylku (rmin), součet absolutních
%hodnot odchylek (rsum) a vykreslí aproximační plochu. Data načítá z prvních
%tří sloupců tabulky v MS Excel. Tento skript je součástí bakalářské práce
%na téma: Aproximace vícerozměrných dat metodou nejmenších čtverců
%
%tento skript využívá autorem vytvořených funkcí Transformace.m,
%FceVBodec.m,VybByz.m, je tedy potřeba je mít ve stejné složce jako
%AproxPol.m
%
%
%autor: Pavel Hrabec


%smazání Comand Window, starých proměných, zavření všech figure
clc;clear;close all;
```

### prosím zadejte:

```
%maximalní stupeň bazových polynomů
k=4;
%zadejte body ve kterých chcete znát hodnotu aproximace jako sloupcové
%vektory (oddělovat ;)
xk1=[30000;32000]; %1. souřadnice bodů
xk2=[1;1.5];%2. souřadnice bodů

%v případě, že chcete použít váženou MNČ zadejte sloupcový vektor vah w
%počet členů vektoru vah musí být shodný s počtem měření
%v případě, že nechcete používat váženou metodu ponechte:
w=ones(63,1);

%do příkazu xlsread do apostrofů prosím vepište název tabulky MS excel s
%daty. Soubor musí být ve stejné složce jako skript a data musí být v
%prvních třech sploupcích seřazena tak, že zřetí sloupec jsou funkční
%hodnoty funkce kterou chceme aproximovat.
S=xlsread('data.xls');
 b=S(:,3);
 x1=S(:,1);
 x2=S(:,2);
```

## normování

```
x1=x1/10000;
xk1=xk1/10000;
```

## transformace x1,x2

```
%pro snížení závislosti x1,x2
TR=Transformace(x1,x2);
xt1=@(x1,x2,TR)(TR(1,1)*x1+TR(1,2)*x2+TR(1,3));
xt2=@(x1,x2,TR)(TR(2,1)*x1+TR(2,2)*x2+TR(2,3));
xT1=x1;
x1=xt1(x1,x2,TR);
x2=xt2(xT1,x2,TR);
```

## zadaní členů báze

```
m=0;
for i=0:k
    for j=0:k-i
         m=m+1;
         ex1(m)=i;
         ex2(m)=j;
    end
end

%bázové polynomy potom budou:
phi=cell(m,1);
for j=1:m
phi{j}=@(x1,x2)(power((x1),ex1(j)).*power((x2),ex2(j)));
end
```

## výpočet návrhové matice A

```
A=ones(numel(x1),m);
for j=1:m
A(:,j)=phi{j}(x1,x2);
end
```

## metoda nejmenších čtverců

```
f=lscov(A,b,w);
```

## hodnoty aproximace v bodech x1 x2

```
G=FceVBodech(phi,f,x1,x2);
%výpočet reziduí
r=b-G;
```

## pokus zúžit bázi

```
[phi,f,r]=VybBaz(phi,r,x1,x2,b,f,w);
```

## výpočet parametrů reziduí

```
rmax=max(abs(r));
rmin=min(abs(r));
rsum=sum(abs(r));
r2sum=r'*r;
```

## transformace zpět

```
%tranformace bázových funkcí
for j=1:numel(phi)
phi{j}=@(x1,x2)(power((xt1(x1,x2,TR)),ex1(j)).*power((xt2(x1,x2,TR)),ex2(j)));
end
%transformace bodů
TR=inv(TR);
xT1=x1;
x1=xt1(x1,x2,TR);
x2=xt2(xT1,x2,TR);
```

## výpočet hodnot aproximace v zadaných bodech

```
G=FceVBodech(phi,f,xk1,xk2);
```

## vykreslení aproximace

zadejte počet bodů pro vykreslení aproximace na jedne ose (síť bude o počtu bodů body^2)

```
body=50;
xp1=linspace(min(x1),max(x1),body)';
xp2=linspace(min(x2),max(x2),body)';
F=MinFce(phi,f,xp1,xp2);

ap=figure;
set(ap,'name','aproximace')
scatter3(x1,x2,b,'+');

hold on
mesh(xp1,xp2,F);
axis([min(x1) max(x1) min(x2) max(x2) min(b)-0.07 max(b)+0.07]);
xlabel('x_1');
ylabel('x_2');
zlabel('b(x_1,x_2)')
```

```
hold off;

%vykreslení zadání
zadano=figure;
set(zadano,'name','body')
scatter3(x1,x2,b,'+g');
xlabel('x_1');
ylabel('x_2');
zlabel('b(x_1,x_2)')
```

## 8.2. Zdrojový kód FceVBodech.m

```
%Tato funkce spočítá hodnotu aproximační funkce v bodech zadaných dvojicí
%souřadnic vektorů [x1,x2].Aproximační funkce je zadaná pomocí anonymních
%funkcí v phi (cell) a koeficientů f. Hodnoty se zapíšou do vektoru F.
%Funkce je součástí skriptu AproxPol.m.
%
%Funkce je součástí bakalářské práce na téma: Aproximace vícerozměrných
%dat metodou nejmenších čtverců.
%
%
%autor: Pavel Hrabec

function [ G ] = FceVBodech( phi,f,x1,x2 )
% F je vektor hodnot aproximace ve vsech bodech [x1,x2]
Fc=zeros(numel(x1),numel(phi));
G=zeros(numel(x1),1);
for j=1:numel(x1)

         for i=1:numel(phi);
            Fc(j,i)=phi{i}(x1(j),x2(j))*f(i);
            G(j)=G(j)+Fc(j,i);
         end

end
end
```

## 8.3. Zdrojový kód MinFce.m

```
%Tato funkce spočítá hodnotu aproximační funkce na síti bodů zadaných dvojicí
%vektorů x1,x2. Aproximační funkce je zadaná pomocí anonymních
%funkcí v phi (cell) a koeficientů f. Hodnoty se zapíší do čtvercové matice
%F. Funkce je součástí skriptu AproxPol.m.
%
%
%Funkce je součástí bakalářské práce na téma: Aproximace vícerozměrných
```

```
%dat metodou nejmenších čtverců.
%
%
%autor: Pavel Hrabec


function [ F ] = MinFce( phi,f,x1,x2 )
% F je matice hodnot aproximace ve vsech bodech [x1,x2]
Fc=zeros(numel(x1),numel(x2),numel(phi));
F=zeros(numel(x1));
for j=1:numel(x1)
for k=1:numel(x2)
         for i=1:numel(phi);
            Fc(j,k,i)=phi{i}(x1(j),x2(k))*f(i);
            F(j,k)=F(j,k)+Fc(j,k,i);
         end
end
end
end
```

## 8.4. Zdrojový kód VybBaz.m

```
%Tato funkce se pokusí zmenšit počet členů polynomiální báze pro aproximaci
%závislosti dvou proměnných tak, že vždy z báze vynechá jeden člen, napočítá
%novou aproximaci a pokud je nový součet čtverců (r2sumn) < starý (r2sum)
%pak je tento člen z báze vyloučen. Tento proces se opakuje dokud se báze
%mnění. Funkce je součástí skriptu AproxPol.m.
%
%Funkce je součástí bakalářské práce na téma: Aproximace vícerozměrných
%dat metodou nejmenších čtverců.
%
%
%autor: Pavel Hrabec


function [ phi,f,r ] = VybBaz( phi,r,x1,x2,b,f,w)
r2sum=r'*r;
n=numel(phi);
for i=1:n-1

for l=1:numel(phi)


    if l==1
        phin=phi(2:numel(phi));
```

```
    elseif l==numel(phi)
        phin=phi(1:numel(phi)-1);

    else
        phin=[phi(1:l-1);phi(l+1:numel(phi))];

    end

    %vypocet matice soustavy A

    A=ones(numel(x1),numel(phin));
    for j=1:numel(phin)
        A(:,j)=phin{j}(x1,x2);
    end

    %metoda nejmensich ctvercu
    fn=lscov(A,b,w);
    %minimalni funkce
    G=FceVBodech(phin,fn,x1,x2);
    %vypocet rezidui
    rn=b-G;
    %rsum=sum(abs(r));
    r2sumn=rn'*rn;

    if r2sumn<r2sum
        phi=phin;
        r2sum=r2sumn;

        f=fn;
        r=rn;
        break


    end

end
if r2sumn>r2sum

    break
end
end


end
```

## 8.5. Tabulka dat

| | | |
|---|---|---|
| 25866 | 0,9 | 0,455889714 |
| 25866 | 0,85 | 0,469047576 |
| 25866 | 0,78 | 0,48947172 |
| 25866 | 0,772 | 0,491828352 |
| 25866 | 0,756 | 0,493203054 |
| 25866 | 0,747 | 0,493203054 |
| 25866 | 0,7039 | 0,491239194 |
| 28970 | 1,117 | 0,4601 |
| 28970 | 1,086 | 0,468324219 |
| 28970 | 1,047 | 0,483795831 |
| 28970 | 1,014 | 0,490242336 |
| 28970 | 0,977 | 0,493416 |
| 28970 | 0,937 | 0,495002832 |
| 28970 | 0,895 | 0,492820938 |
| 34448 | 1,491 | 0,467 |
| 34448 | 1,447 | 0,4991 |
| 34448 | 1,403 | 0,4968 |
| 34448 | 1,366 | 0,494 |
| 34448 | 1,299 | 0,4867 |
| 34448 | 1,26 | 0,4817 |
| 39644 | 2,001 | 0,4778 |
| 39644 | 1,966 | 0,4871 |
| 39644 | 1,929 | 0,4968 |
| 39644 | 1,888 | 0,4942 |
| 39644 | 1,847 | 0,4888 |
| 39644 | 1,792 | 0,4816 |
| 39644 | 1,737 | 0,4723 |
| 41386 | 2,206 | 0,4828 |
| 41386 | 2,165 | 0,4925 |
| 41387 | 2,138 | 0,499 |
| 41388 | 2,098 | 0,4974 |

| | | |
|---|---|---|
| 41389 | 2,055 | 0,4914 |
| 41390 | 1,937 | 0,4764 |
| 43110 | 2,398 | 0,4837 |
| 43110 | 2,373 | 0,4889 |
| 43110 | 2,342 | 0,4954 |
| 43110 | 2,288 | 0,4963 |
| 43110 | 2,232 | 0,4926 |
| 43110 | 2,18 | 0,4869 |
| 44550 | 2,588 | 0,4816 |
| 44550 | 2,574 | 0,4878 |
| 44550 | 2,561 | 0,4918 |
| 44550 | 2,531 | 0,4961 |
| 44550 | 2,491 | 0,4984 |
| 44550 | 2,422 | 0,4978 |
| 44550 | 2,352 | 0,4938 |
| 44550 | 2,29 | 0,4891 |
| 45697 | 2,692 | 0,4708 |
| 45697 | 2,684 | 0,4783 |
| 45697 | 2,657 | 0,4846 |
| 45697 | 2,629 | 0,488 |
| 45697 | 2,602 | 0,4888 |
| 45697 | 2,547 | 0,4871 |
| 45697 | 2,492 | 0,485 |
| 45697 | 2,394 | 0,4798 |
| 46779 | 2,944 | 0,4608 |
| 46779 | 2,935 | 0,4682 |
| 46779 | 2,908 | 0,4744 |
| 46779 | 2,878 | 0,4778 |
| 46779 | 2,85 | 0,4785 |
| 46779 | 2,79 | 0,4769 |
| 46779 | 2,733 | 0,4747 |
| 46779 | 2,63 | 0,4697 |

## 8.6. Přílohy na cd

Na přiloženém cd najdete vytvořené programy AproxPol.m, FceVBode.m, MinFce.m a VybBaz.m a soubor se zadáním praktického příkladu data.xls.