UNIVERZITA PALACKÉHO V OLOMOUCI
PŘÍRODOVĚDECKÁ FAKULTA

# DIPLOMOVÁ PRÁCE

## Ortogonální B-splajny a jejich použití

**Katedra matematické analýzy a aplikací matematiky**
Vedoucí diplomové práce: **doc. RNDr. Jitka Machalová, Ph.D.**
Vypracoval: **Bc. Petr Tománek**
Studijní program: N1101 Matematika
Studijní obor: Matematika a její aplikace
Forma studia: prezenční
Rok odevzdání: 2022

# BIBLIOGRAFICKÁ IDENTIFIKACE

**Autor:** Bc. Petr Tománek

**Název práce:** Ortogonální B-splajny a jejich použití

**Typ práce:** Diplomová práce

**Pracoviště:** Katedra matematické analýzy a aplikací matematiky

**Vedoucí práce:** doc. RNDr. Jitka Machalová, Ph.D.

**Rok obhajoby práce:** 2022

**Abstrakt:** Splajn se dá chápat jako funkce, jež je po částech složená z polynomů. Je důležitým nástrojem k aproximaci dat a dá se zapsat jako lineární kombinace bázových funkcí. Jedny z takových funkcí jsou právě B-splajny, které mají spoustu užitečných vlastností. Ortogonalita však mezi tyto vlastnosti nepatří. Cílem této práce je metody ortogonalizace B-splajnů představit a ukázat jejich užitečnost k aproximaci dat. Práce je navíc doplněna o kódy jednotlivých metod vytvořených v matematickém softwaru MATLAB.

**Klíčová slova:** B-splajn, ortogonalizace, splinet, software MATLAB

**Počet stran:** 70

**Počet příloh:** 1

**Jazyk:** český

# BIBLIOGRAPHICAL IDENTIFICATION

**Author:** Bc. Petr Tománek

**Title:** Orthogonal B-splines and their uses

**Type of thesis:** Master's

**Department:** Department of Mathematical Analysis and Application of Mathematics

**Supervisor:** doc. RNDr. Jitka Machalová, Ph.D.

**The year of presentation:** 2022

**Abstract:** Spline can be understood as a function, that is composed of polynomials in parts. It is an important tool for data approximation and it can be written as a linear combination of basis functions. One of these functions are B-splines, which have a lot of useful features. However, orthogonality is not one of those features. The goal of the thesis is to introduce the methods of orthogonalization of B-splines and present their uses for data approximation. Additionally, the thesis is supplemented by codes of each method, which were made in a mathematical software MATLAB.

**Key words:** B-spline, orthogonalization, splinet, software MATLAB

**Number of pages:** 70

**Number of appendices:** 1

**Language:** Czech

**Prohlášení**

Prohlašuji, že jsem diplomovou práci zpracoval samostatně pod vedením paní doc. RNDr. Jitky Machalové, Ph.D. a všechny použité zdroje jsem uvedl v seznamu literatury.

V Olomouci dne ..... .....

podpis

**Poděkování**

Rád bych zde poděkoval vedoucí práce doc. RNDr. Jitce Machalové, Ph.D. za spolupráci a čas, který mi věnovala při konzultacích. Dále děkuji mé rodině za nepřetržitou podporu během celého studia.

# Obsah

# Seznam použitých symbolů

| | |
|---|---|
| $\mathcal{C}^k[a,b]$ | prostor spojitě diferencovatelných funkcí až do řádu $k$ na intervalu $[a,b]$ |
| $\mathbb{N}$ | množina přirozených čísel |
| $\mathbb{R}$ | množina reálných čísel |
| $\mathbb{R}^n$ | množina všech reálných vektorů délky $n$ |
| $\langle .,. \rangle$ | diskrétní nebo spojitý skalární součin |
| $\|.\|$ | norma |
| $\lfloor . \rfloor$ | dolní celá část |

# Úvod

Problematika aproximace dat je nedílnou součástí matematické disciplíny. V dnešní době již existuje mnoho způsobů, jak správné aproximace docílit, a jedním z nich je pomocí takzvaných splajnů.

Splajn se dá stručně popsat jako funkce, která je po částech složená z polynomů. Za zakladatele matematické splajnové teorie se považuje Isaac Jacob Schoenberg, jenž poprvé použil slovo splajn v matematickém smyslu ve svém článku v polovině 20. století. Původně však tento termín byl znám pod jinou definicí. Již v 19. století se ke konstrukci lodí využívalo pružné pravítko, které se po upevnění v určitých bodech prohýbalo tak, že vytvářelo hladkou křivku. Tomuto pravítku se v angličtině říkalo „spline“. Později se zjistilo, že taková křivka je po částech polynomem třetího stupně, čili kubickým splajnem.

Splajny lze matematicky zapsat jako lineární kombinaci bázových funkcí. Jedny z takových bázových funkcí jsou i bázové splajny (zkráceně B-splajny). Ty mají spoustu užitečných vlastností, ale obecně nejsou ortogonální. Cílem mé diplomové práce je čtenáře s ortogonálními bázovými splajny seznámit a ukázat jejich výpočet či využití na příkladech v softwaru MATLAB.

Práce je rozdělena do čtyř kapitol. V první kapitole se seznámíme či připomeneme pojmy z numerických metod, které bude nutné znát v následujících kapitolách. Konkrétně se bude jednat o definice a vlastnosti splajnů či B-splajnů a jejich využití k aproximaci dat. Ve druhé kapitole se podíváme na ortogonální B-splajny a základní metody, jak takové ortogonalizace dosáhnout. Tyto základní metody budou celkem dvě. Téma třetí kapitoly bude o novém způsobu ortogonalizace B-splajnů pomocí tzv. splinetů. Závěrečná čtvrtá kapitola se bude zabývat názorným využitím ortogonálních B-splajnů k aproximaci dat.

Práce je doplněna o kódy a obrázky příkladů jednotlivých metod ortogonalizací či aproximací, které byly vytvořeny v matematickém softwaru MATLAB R2017a. V daných kódech byly hlavně využity funkce z balíčku `spline`. V práci se předpokládá, že je čtenář alespoň obeznámen se základy softwaru MATLAB.

Kódy se nacházejí na CD, které je součástí přílohy.

# 1. Přípravná kapitola

Tato kapitola slouží k seznámení či zopakování pojmů, které budeme potřebovat v následujících kapitolách. Přitom se předpokládá, že je čtenář již obeznámen se základními poznatky z lineární algebry a numerických metod.

## 1.1. Splajny

Poznatky a definice z podkapitoly 1.1 byly čerpány z [2] a [3].

**Definice 1.1.** Nechť je dán interval $[a, b]$ na $\mathbb{R}$ a na něm $n + 2$ bodů $\lambda_i$, kde $n \in \mathbb{N}$. Jestliže pro tyto body platí

$$a = \lambda_0 < \lambda_1 < \cdots < \lambda_n < \lambda_{n+1} = b,$$

pak je můžeme nazývat sítí uzlů. Síť uzlů tvořenou body $\lambda_i$, $i = 0, \ldots, g + 1$, budeme značit $(\Delta\lambda)$.

**Definice 1.2.** Nechť je na intervalu $[a, b]$ daná síť uzlů

$$(\Delta\lambda) : a = \lambda_0 < \lambda_1 < \cdots < \lambda_g < \lambda_{g+1} = b.$$

Na dané síti $(\Delta\lambda)$ budeme polynomickým splajnem stupně $k$ s defektem $d$ rozumět funkci $s_{kd}(x)$, pro kterou platí:

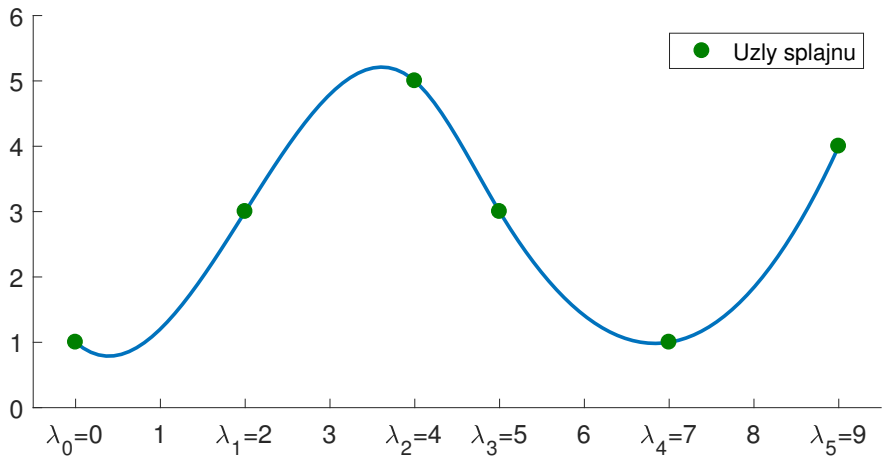
1. $s_{kd}(x)$ je na každém intervalu $[\lambda_i, \lambda_{i+1}]$, $i = 0, \ldots, g$, polynomem stupně nejvýše $k$

2. $s_{kd}(x) \in C^{k-d}[a, b]$, tj. v uzlech má $s_{kd}(x)$ spojité derivace až do řádu $m - d$

Symbolem $S_{kd}^{\Delta\lambda}$ značíme lineární prostor splajnů $s_{kd}(x)$ na dané síti uzlů $(\Delta\lambda)$. Jestliže jsou v bodech $x_j \in [a, b]$ předepsány hodnoty $y_j$, kde $j = 1, \ldots, m$, pak splajn $s_{kd}(x) \in S_{kd}^{\Delta\lambda}$ splňující

$$s_{kd}(x_j) = y_j, \ j = 1, \ldots, m,$$

nazýváme interpolačním splajnem.




Obrázek 1.1: Polynomický splajn stupně 3 s defektem 1

**Poznámka 1.1.** Příklad polynomického splajnu stupně 3 s defektem 1 lze vidět na obrázku 1.1.

**Poznámka 1.2.** Pro určení dimenze prostoru $S_{kd}^{\Delta\lambda}$ je nutno vzít v potaz následující:

- Na každém intervalu $[\lambda_i, \lambda_{i+1}]$ je splajn dán polynomem stupně nejvýše $k$. Každý takový polynom je tedy určen $k+1$ koeficienty. Jelikož na $[a, b]$ máme celkem $g + 1$ takových intervalů, dohromady potřebujeme $(g + 1)(k + 1)$ parametrů.

- V $g$ vnitřních uzlech sítě požadujeme splnění podmínek na hladkost splajnu, tj. požadujeme, aby v daných uzlech měl splajn spojité derivace až do řádu $k - d$. Každý vnitřní uzel tedy musí splňovat $k - d + 1$ podmínek. Celkem máme $g(k - d + 1)$ podmínek.

Pro dimenzi $S_{kd}^{\Delta\lambda}$ pak platí

$$\begin{aligned}\dim S_{kd}^{\Delta\lambda} &= (g+1)(k+1) - g(k-d+1) = \\ &= gk + g + k + 1 - gk + gd - g = \\ &= gd + k + 1\end{aligned}$$

**Poznámka 1.3.** Prostor $S_{k1}^{\Delta\lambda}$ budeme pro jednoduchost značit $S_k^{\Delta\lambda}$ a funkce z takového prostoru budeme značit $s_k(x)$.

### 1.1.1. Splajny jako lineární kombinace bázových funkcí

Vzhledem k tomu, jak jsou polynomické splajny definované, je přirozené uvažovat nad bázovou reprezentací těchto funkcí. Budeme proto chtít splajny $s_k(x)$ z prostoru $S_k^{\Delta\lambda}$ jednoznačně vyjádřit jako lineární kombinaci bázových funkcí. Typů bázových funkcí je mnoho, ale ne všechny je z numerického hlediska vhodné zvolit. Pro naše účely bude nejvýhodnější zvolit takové bázové funkce, jejichž intervaly, na kterém jsou funkce nenulové, jsou poměrně malé. Interval, na kterém funkce nabývá nenulových hodnot, nazýváme nosič dané funkce. Jednou z takových funkcí je například takzvaná useknutá mocnina.

Následující definice 1.3 a věta 1.1 byly převzaty z [3].

**Definice 1.3.** Funkci $(t-u)_+^k$ proměnné $t$ a pevně daným $u$ nazýváme useknutou mocninou, jestliže pro ni platí:

$$(t-u)_+^k = \begin{cases} (t-u)^k & t > u \\ 0 & t \leq u \end{cases}$$

**Věta 1.1.** *Funkce $x^j$, $j = 0, \ldots, k$ a $(x-\lambda_i)_+^k$, $i = 1, \ldots, g$ spolu tvoří bázi prostoru $S_k^{\Delta\lambda}$.*

Je tedy zřejmé, že $s_k(x) \in S_k^{\Delta\lambda}$ lze jednoznačně vyjádřit jako

$$s_k(x) = \sum_{j=0}^{k} b_j x^j + \sum_{j=1}^{g} c_j (x-\lambda_j)_+^k$$

Z důvodu nestability při numerických výpočtech však není vhodné s touto bázovou reprezentací pracovat. Pomocí useknuté mocniny si však můžeme definovat vhodnější bázi a to takzvané B-splajny.

## 1.2. B-splajny

Nyní se podíváme na výhodnější reprezentaci bázových funkcí. Definice 1.4 a 1.5, věta 1.2, poznámka 1.4, poznatky a věty z podkapitoly 1.2.1 byly čerpány z [2] a [3], zatímco poznatky z podkapitol 1.2.2 – 1.2.4 byly čerpány z [5] a [6].

**Definice 1.4.** Nechť je na intervalu $[a, b]$ daná síť uzlů $(\Delta x)$ a nechť je funkce $f$ v těchto bodech definovaná. Poměrná diference 1. řádu funkce $f$ na $(\Delta x)$ je definovaná vztahem

$$f[x_i, x_{i+1}] = \frac{f(x_{i+1}) - f(x_i)}{x_{i+1} - x_i}$$

pro $i = 0, \ldots, n$.
Poměrná diference $k$-tého řádu funkce $f$ na $(\Delta x)$ je definovaná vztahem

$$f[x_i, \ldots, x_{i+k}] = \frac{f[x_{i+1}, \ldots, x_{i+k}] - f[x_i, \ldots, x_{i+k-1}]}{x_{i+k} - x_i}$$

pro $i = 0, \ldots, n - k + 1$.

**Definice 1.5.** B-splajn $B_i^{k+1}(x)$ stupně $k$ s uzly $\lambda_i, \ldots, \lambda_{i+k+1}$ je definován jako

$$B_i^{k+1}(x) = (\lambda_{i+k+1} - \lambda_i)(t - x)_+^k \left[\lambda_i, \ldots, \lambda_{i+k+1}\right]$$

**Věta 1.2.** *$B_i^{k+1}(x)$ má následující vlastnosti:*

1. *$B_i^{k+1}(x)$ je na každém intervalu $[\lambda_j, \lambda_{j+1}]$ polynomem stupně nejvýše $k$*
2. *$B_i^{k+1}(x) \in C^{k-1}[\lambda_i, \lambda_{i+k+1}]$*
3. *$B_i^{k+1}(x) \geq 0$ pro $x \in [a, b]$*
4. *Nosičem $B_i^{k+1}(x) = 0$ je interval $[\lambda_i, \lambda_{i+k+1}]$. Tedy:*
   *$B_i^{k+1}(x) = 0$ pro $x \notin [\lambda_i, \lambda_{i+k+1}]$*
5. *$\left(B_i^{k+1}\right)^{(l)}(\lambda_i) = \left(B_i^{k+1}\right)^{(l)}(\lambda_{i+k+1}) = 0$ pro $\forall l = 0, \ldots, k - 1$*

6. $\left(B_i^{k+1}(x)\right)' = k\left(\frac{B_i^k(x)}{\lambda_{i+k}-\lambda_i} - \frac{B_{i+1}^k(x)}{\lambda_{i+k+1}-\lambda_{i+1}}\right)$

7. *Pro* $i = 0, \ldots, g+1$ *platí následující rekurentní vztahy:*

$$B_i^1(x) = \begin{cases} 1 & \lambda_i \leq x < \lambda_{i+1} \\ 0 & jinak \end{cases}$$

$$B_i^{k+1}(x) = \frac{x-\lambda_i}{\lambda_{i+k}-\lambda_i} B_i^k(x) + \frac{\lambda_{i+k+1}-x}{\lambda_{i+k+1}-\lambda_{i+1}} B_{i+1}^k(x), \ k = 1, 2, \ldots$$

**Poznámka 1.4.** B-splajny jsou lineárně nezávislé funkce.

### 1.2.1. Splajny jako lineární kombinace B-splajnů

Vzhledem k lineární nezávislosti B-splajnů jsou tyto funkce vhodné k vytvoření báze. Na dané síti uzlů ($\Delta\lambda$) lze však sestrojit pouze $g-k+1$ lineárně nezávislých B-splajnů $B_i^{k+1}(x)$, kde $i = 0, \ \ldots, \ g-k$. Abychom pomocí B-splajnů obdrželi celou bázi prostoru $S_k^{\Delta\lambda}$, budeme muset sestrojit celkem $2k$ lineárně nezávislých B-splajnů navíc. Zavedeme tedy $2k$ pomocných uzlů $\lambda_{-k}, \ldots, \lambda_{-1}$ a $\lambda_{g+2}, \ldots, \lambda_{g+k+2}$ splňující následující nerovnosti:

$$\lambda_{-k} \leq \lambda_{-k+1} \leq \cdots \leq \lambda_{-1} \leq \lambda_0 = a$$
$$b = \lambda_{g+1} \leq \lambda_{g+2} \leq \cdots \leq \lambda_{g+k+1} \leq \lambda_{g+k+2}$$

Na těchto uzlech lze pak zkonstruovat B-splajny $B_{-k}^{k+1}(x), \ldots, B_{-1}^{k+1}(x)$ a $B_{g-k+1}^{k+1}(x), \ldots, B_g^{k+1}(x)$. Celá báze prostoru $S_k^{\Delta\lambda}$ bude tvořena B-splajny $B_{-k}^{k+1}(x), \ldots, B_g^{k+1}(x)$. Každý splajn $s_k \in S_k^{\Delta\lambda}$ lze pak vyjádřit jako lineární kombinaci těchto B-splajnů. Tedy

$$s_k(x) = \sum_{i=-k}^{g} b_i B_i^{k+1}(x),$$

což lze maticově zapsat jako

$$s_k(x) = \boldsymbol{C}_{k+1}(x)\boldsymbol{b}$$

Každý splajn je pak jednoznačně určen vektorem B-splajnových koeficientů $\boldsymbol{b} = (b_{-k}, b_{-k+1}, \ldots, b_g)^T$ a takzvanou kolokační maticí, která pro bod $x$ vypadá jako $\boldsymbol{C}_{k+1}(x) = (B_{-k}^{k+1}(x), B_{-k+1}^{k+1}(x), \ldots, B_{-1}^{k+1}(x))$.

**Poznámka 1.5.** Pro vektor $\boldsymbol{x} = (x_1, x_2, \ldots, x_n)^T$ kolokační matice vypadá jako

$$\boldsymbol{C}_{k+1}(\boldsymbol{x}) = \begin{pmatrix} B_{-k}^{k+1}(x_1) & \ldots & B_g^{k+1}(x_1) \\ \vdots & \ddots & \vdots \\ B_{-k}^{k+1}(x_n) & \ldots & B_g^{k+1}(x_n) \end{pmatrix}$$

**Poznámka 1.6.** Uzlům na rozšířené síti uzlů, pro které platí $\lambda_i = \lambda_j$, $i \neq j$, se říká násobné uzly.

**Věta 1.3.** *Nechť $B_{-k}^{k+1}(x), \ldots, B_g^{k+1}(x)$ jsou B-splajny tvořící bázi prostoru $S_k^{\Delta\lambda}$. Poté platí následující rovnosti:*

$$\sum_{i=-k}^{g} B_i^{k+1}(x) = 1, \ \forall x \in [a, b],$$

*respektive*

$$\sum_{i=j-k}^{j} B_i^{k+1}(x) = 1, \ \forall x \in [\lambda_j, \lambda_{j+1}].$$

**Věta 1.4.** *Nechť $B_{-k}^{k+1}(x), \ldots, B_g^{k+1}(x)$ jsou B-splajny tvořící bázi prostoru $S_k^{\Delta\lambda}$ a $s_k \in S_k^{\Delta\lambda}$. Poté $l$-tou derivaci splajnu $s_k$, kde $l \in \{1, \ldots, k-1\}$, lze zapsat jako*

$$s_k^{(l)}(x) = \prod_{t=1}^{l}(k+1-t) \sum_{-(k-l)}^{g} b_i^{(l)} B_i^{k+1-l}, \ \text{přičemž}$$

$$b_i^{(m)} = \begin{cases} b_i & pro\ m = 0 \\ \dfrac{b_i^{(m-1)} - b_{i-1}^{(m-1)}}{\lambda_{i+k+1-m} - \lambda_i} & pro\ m > 0 \end{cases}$$

### 1.2.2. Interpolace

Pro danou síť uzlů $(\Delta\lambda)$, dané $k$ a dané body $(x_j, y_j)$, $a \leq x_j \leq b,\ j = 1, \ldots, n$, chceme nalézt $s_k(x) \in S_k^{\Delta\lambda}[a, b]$ tak, aby platilo

$$s_k(x_j) = y_j,\ j = 1, \ldots, n.$$

Jinými slovy, chceme nalézt splajn na $(\Delta\lambda)$, který by procházel body $(x_j, y_j)$. Jak již víme, takový splajn $s_k(x)$ lze zapsat jako lineární kombinaci B-splajnů a je určen jednoznačně vektorem $\boldsymbol{b}$. Budeme se tedy snažit nalézt takový vektor $\boldsymbol{b}$, který bude řešit soustavu

$$\boldsymbol{C}_{k+1}(\boldsymbol{x})\boldsymbol{b} = \boldsymbol{y},$$

kde $\boldsymbol{x} = (x_1, x_2, \ldots, x_n)^T,\ \boldsymbol{y} = (y_1, y_2, \ldots, y_n)^T$ a $\boldsymbol{C}_{k+1}(\boldsymbol{x})$ je kolokační matice.

- Jestliže je matice $\boldsymbol{C}_{k+1}(\boldsymbol{x})$ regulární, existuje právě jedno řešení soustavy, které lze najít jako

$$\boldsymbol{b} = \Big[\boldsymbol{C}_{k+1}(\boldsymbol{x})\Big]^{-1}\boldsymbol{y}$$

- Jestliže je $\boldsymbol{C}_{k+1}(\boldsymbol{x})$ plné řádkové hodnosti, pak tato soustava má nekonečně mnoho řešení. V tomto případě nás může zajímat řešení, které navíc minimalizuje funkcionál

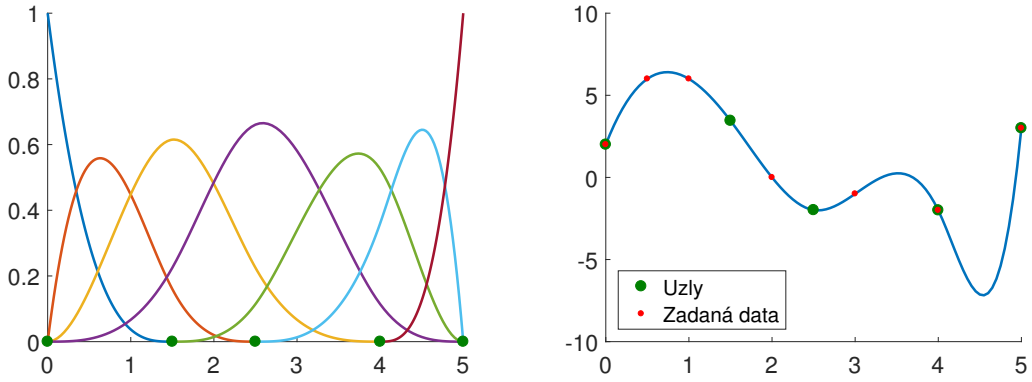
$$J_l(s_k) = \int_a^b \left(s_k^{(l)}(x)\right)^2 dx = \int_a^b \left(\Big(\sum_{i=-k}^{g} b_i B_i^{k+1}(x)\Big)^{(l)}\right)^2 dx,\ l \in \{0, 1, \ldots, k-1\}$$

  Splajnu, který interpoluje zadaná data a minimalizuje funkcionál $J_l(s_k)$, se říká optimální interpolační splajn.

- Jestliže je $\boldsymbol{C}_{k+1}(\boldsymbol{x})$ plné sloupcové hodnosti, pak tato soustava nemá řešení. Lze však najít vektor $\boldsymbol{b}^*$, který minimalizuje normu rezidua

$$\Big\|\boldsymbol{C}_{k+1}(\boldsymbol{x})\boldsymbol{b} - \boldsymbol{y}\Big\|$$

  Takový vektor lze získat užitím Moore-Penroseovy inverze.




(a) Použité B-splajny (b) Polynomiální interpolační splajn

Obrázek 1.2: Příklad interpolace

**Poznámka 1.7.** Moore-Penroseovými inverzemi jsem se zabýval v bakalářské práci *Zobecněná Moore-Penroseova inverze a její výpočet.*

**Příklad 1.1.** Nechť máme v $\mathbb{R}^2$ zadaná data {(0, 2), (0,5, 6), (1, 6), (2, 0), (3, -1), (4, -2), (5, 3)} a chceme nalézt splajn stupně 3, který by danými body procházel.

Jako síť uzlů si zvolíme 5 bodů {0, 1,5, 2,5, 4, 5 }. Tuto síť v MATLABu rozšíříme násobnými krajními uzly pomocí příkazu `augknt`. Dále sestrojíme B-splajny na rozšířené síti uzlů příkazem `spmak` a vytvoříme kolokační matici pomocí `spcol`. Jelikož je kolokační matice regulární, existuje právě jeden interpolační splajn a z poznatků podkapitoly 1.2.2 ho můžeme nalézt. Na obrázku 1.2 je znázorněn výsledek.

### 1.2.3. Metoda nejmenších čtverců

Pro danou síť uzlů ($\Delta\lambda$), dané $k$, dané body $(x_j, y_j)$, $a \leq x_j \leq b$, a váhové koeficienty $w_j > 0$, kde $j = 1, \ldots, n$, chceme nalézt $s_k(x) \in S_k^{\Delta\lambda}[a, b]$ minimalizující výraz

$$\sum_{j=1}^{n} w_j \Big(y_j - s_k(x_j)\Big)^2$$

Jinými slovy, chceme nalézt takový splajn, pro který platí, že součet čtverců odchylek $s_k(x_j)$ od $y_j$ pro všechna $j$ je minimální.

Splajn $s_k(x)$ z prostoru $S_k^{\Delta\lambda}[a,b]$ lze zapsat jako lineární kombinaci B-splajnů, tudíž úlohu můžeme přepsat na hledání takového vektoru $\boldsymbol{b}$, který minimalizuje výraz

$$\delta(\boldsymbol{b}) = \sum_{j=1}^{n} w_j \Big(y_j - \sum_{i=-k}^{g} b_i B_i^{k+1}(x_j)\Big)^2,$$

což lze zapsat maticově jako

$$\begin{aligned}\delta(\boldsymbol{b}) &= \Big(\boldsymbol{y} - \boldsymbol{C}_{k+1}(\boldsymbol{x})\boldsymbol{b}\Big)^T \boldsymbol{W}\Big(\boldsymbol{y} - \boldsymbol{C}_{k+1}(\boldsymbol{x})\boldsymbol{b}\Big) \\ &= \boldsymbol{b}^T\boldsymbol{C}_{k+1}^T(\boldsymbol{x})\boldsymbol{W}\boldsymbol{C}_{k+1}(\boldsymbol{x})\boldsymbol{b} - 2\boldsymbol{y}\boldsymbol{W}\boldsymbol{C}_{k+1}(\boldsymbol{x})\boldsymbol{b} + \boldsymbol{y}^T\boldsymbol{W}\boldsymbol{y},\end{aligned}$$

kde $\boldsymbol{x} = (x_1,\ldots,x_n)^T$, $\boldsymbol{y} = (y_1,\ldots,y_n)^T$, $\boldsymbol{b} = (b_{-k},\ldots,b_g)^T$, $\boldsymbol{C}_{k+1}(\boldsymbol{x})$ je kolokační matice a $\boldsymbol{W}$ je diagonální matice typu $n \times n$, která má váhové koeficienty $w_j$, $j = 1,\ldots,n$ na diagonále.

Funkce $\delta(\boldsymbol{b})$ je kvadratickou funkcí proměnné $\boldsymbol{b}$. Z numerických metod víme, že pokud hessián této funkce bude pozitivně definitní, pak bude existovat právě jedno její minimum. Hessián funkce $\delta(\boldsymbol{b})$ je

$$\frac{\partial^2\delta(\boldsymbol{b})}{\partial\boldsymbol{b}^T\partial\boldsymbol{b}} = \boldsymbol{C}_{k+1}^T(\boldsymbol{x})\boldsymbol{W}\boldsymbol{C}_{k+1}(\boldsymbol{x})$$

Čtvercová matice $\boldsymbol{C}_{k+1}^T(\boldsymbol{x})\boldsymbol{W}\boldsymbol{C}_{k+1}(\boldsymbol{x})$ typu $(g+k+1)\times(g+k+1)$ je pozitivně definitní právě tehdy, když $\boldsymbol{C}_{k+1}(x)$ je plné sloupcové hodnosti. K určení, zdali matice je plné sloupcové hodnosti, nám pomůže následující věta 1.5 převzatá z [5].

**Věta 1.5.** *(Schoenberg-Whitney) Nechť je dána rozšířené síť uzlů $(\Delta\lambda)$ a vektor $\boldsymbol{x} \in \mathbb{R}^n$. Pak pro kolokační matice $\boldsymbol{C}_{k+1}(\boldsymbol{x})$ B-splajnů $B_i^{k+1}(x)$, $i = -k,\ldots,g$ platí následující tvrzení:*

1. *$\boldsymbol{C}_{k+1}(\boldsymbol{x})$ je plné sloupcové hodnosti právě tehdy, když $n > g+k+1$ a když existuje množina bodů $\{\mu_{-k},\ldots,\mu_g\} \subset \{x_1,\ldots x_n\}$, přičemž platí:*

(a) $\mu_i < \mu_i + 1,\ i = -k, \ldots, g-1$

(b) $\lambda_i < \mu_i < \lambda_{i+k+1},\ i = -k, \ldots, g$

2. $\boldsymbol{C}_{k+1}(\boldsymbol{x})$ *je regulární právě tehdy, když* $n = g + k + 1$ *a když platí:*

   (a) $\lambda_{i-k-1} < x_i < \lambda_i,\ i = 1, \ldots, n$

3. $\boldsymbol{C}_{k+1}(\boldsymbol{x})$ *je plné řádkové hodnosti právě tehdy, když* $n < g + k + 1$ *a když existuje množina bodů* $\{\mu_1, \ldots, \mu_n\} \subset \{\lambda_{-k}, \ldots \lambda_g\}$*, přičemž platí:*

   (a) $\mu_i < \mu_i + 1,\ i = 1, \ldots, n-1$

   (b) $\mu_i < x_i < \mu_{i+k+1},\ i = 1, \ldots, n$

Věta nám jinými slovy říká, že aby byla matice $\boldsymbol{C}_{k+1}(\boldsymbol{x})$ plné sloupcové hodnosti, tak se v každém nosiči B-splajnů musí nacházet alespoň jeden z bodů $x_i$ původních dat.

**Příklad 1.2.** Nechť je dána síť uzlů ($\Delta\lambda$) a data $\boldsymbol{x} \in \mathbb{R}^8$ a $\boldsymbol{y} \in \mathbb{R}^8$. Úkolem je zjistit, zdali jsou kolokační matice $\boldsymbol{C}_{k+1}(\boldsymbol{x})$ či $\boldsymbol{C}_{k+1}(\boldsymbol{y})$ plné sloupcové hodnosti aniž bychom jejich hodnost počítali numericky.

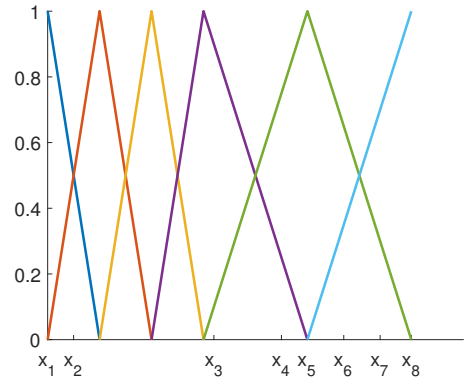
Jestliže si vykreslíme B-splajny na rozšířené síti uzlů a body $x_i$ či $y_j$ do jednoho grafu, pak lze řešení vyčíst rovnou z obrázků. Viz obrázky 1.3a a 1.3b.

Předpokládejme, že $\boldsymbol{C}_{k+1}(\boldsymbol{x})$ je plné sloupcové hodnosti. Pak hessián funkce $\delta(\boldsymbol{b})$ je pozitivně definitní a tak pro tuto funkci existuje právě jedno minimum. Nejjednodušší způsob, jak takové minimum najít, je funkci $\delta(\boldsymbol{b})$ zderivovat podle hledané proměnné a položit ji rovnou nule. V daném případě tedy celý výraz zderivujeme podle vektoru $\boldsymbol{b}$.

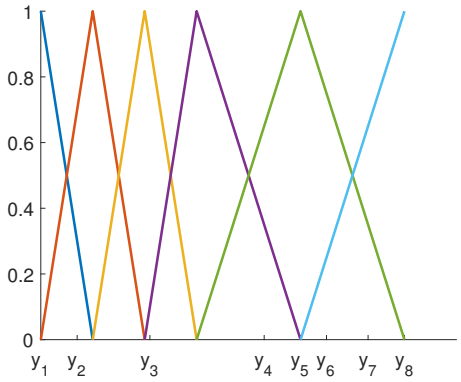
$$0 = \frac{\partial \delta(\boldsymbol{b})}{\partial \boldsymbol{b}} = -2\boldsymbol{C}_{k+1}^T(\boldsymbol{x})\boldsymbol{W}\boldsymbol{y} + 2\boldsymbol{C}_{k+1}^T(\boldsymbol{x})\boldsymbol{W}\boldsymbol{C}_{k+1}(\boldsymbol{x})\boldsymbol{b}$$

$$\boldsymbol{C}_{k+1}^T(\boldsymbol{x})\boldsymbol{W}\boldsymbol{C}_{k+1}(\boldsymbol{x})\boldsymbol{b} = \boldsymbol{C}_{k+1}^T(\boldsymbol{x})\boldsymbol{W}\boldsymbol{y}$$

Na levé straně rovnice se nachází matice $\boldsymbol{C}_{k+1}^T(\boldsymbol{x})\boldsymbol{W}\boldsymbol{C}_{k+1}(\boldsymbol{x})$, kterou si pro jednoduchost označíme jako $\boldsymbol{B}$. Na pravé straně rovnice máme po roznásobení vektor




(a) $\boldsymbol{C}_{k+1}(\boldsymbol{x})$ není plné sloupcové hodnosti




(b) $\boldsymbol{C}_{k+1}(\boldsymbol{y})$ je plné sloupcové hodnosti

Obrázek 1.3: Příklad dat, jejichž kolokační matice mají nebo nemají plné sloupcové hodnosti

velikosti $n$, který si označíme například $\boldsymbol{v}$. Tedy

$$\underbrace{\boldsymbol{C}_{k+1}^T(\boldsymbol{x})\boldsymbol{W}\boldsymbol{C}_{k+1}(\boldsymbol{x})}_{\boldsymbol{B}}\boldsymbol{b} = \underbrace{\boldsymbol{C}_{k+1}^T(\boldsymbol{x})\boldsymbol{W}\boldsymbol{y}}_{\boldsymbol{v}}$$

$$\boldsymbol{B}\boldsymbol{b} = \boldsymbol{v}$$

Danou soustavu nazýváme normální soustava rovnic. Pokud je matice $\boldsymbol{B}$ regulární, pak k ní existuje i matice inverzní a řešení normální soustavy lze pak určit jednoznačně. Takové řešení vypadá jako

$$\boldsymbol{b} = \boldsymbol{B}^{-1}\boldsymbol{v}$$

Za předpokladu plné sloupcové hodnosti matice $\boldsymbol{C}_{k+1}(\boldsymbol{x})$ platí, že je matice $\boldsymbol{B}$ pozitivně definitní a tudíž je i regulární. K určení jednoznačného řešení soustavy nám tedy stačí pouze plná hodnost matice $\boldsymbol{C}_{k+1}(\boldsymbol{x})$.

Jestliže bude vektor $\boldsymbol{b}^*$ řešením normální soustavy, potom hledaný splajn, který aproximuje daná data ve smyslu MNČ, bude vypadat jako

$$s_k^*(x) = \sum_{i=-k}^{g} b_i^* B_i^{k+1}(x)$$

**Poznámka 1.8.** Matici $\boldsymbol{B}$ a vektor $\boldsymbol{v}$ lze zapsat pomocí skalárních součinů jako

$$\boldsymbol{B} = \begin{pmatrix} \left\langle B_{-k}^{k+1}, B_{-k}^{k+1} \right\rangle & \dots & \left\langle B_{g}^{k+1}, B_{-k}^{k+1} \right\rangle \\ \vdots & \ddots & \vdots \\ \left\langle B_{-k}^{k+1}, B_{g}^{k+1} \right\rangle & \dots & \left\langle B_{g}^{k+1}, B_{g}^{k+1} \right\rangle \end{pmatrix}, \boldsymbol{v} = \begin{pmatrix} \left\langle \boldsymbol{y}, B_{-k}^{k+1} \right\rangle \\ \vdots \\ \left\langle \boldsymbol{y}, B_{g}^{k+1} \right\rangle \end{pmatrix},$$

přičemž si skalární součin v případě aproximace diskrétních dat pomocí metody nejmenších čtverců definujeme jako tzv. diskrétní skalární součiny. Diskrétní skalární součin mezi dvěma B-splajny vypadá jako

$$\left\langle B_{r}^{k+1}, B_{s}^{k+1} \right\rangle = \sum_{j=1}^{n} w_j B_{r}^{k+1}(x_j) B_{s}^{k+1}(x_j),$$

zatímco diskrétní skalární součin mezi vektorem a B-splajnem vypadá jako

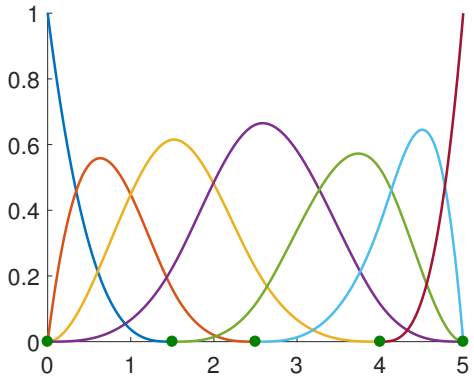
$$\left\langle \boldsymbol{y}, B_{s}^{k+1} \right\rangle = \sum_{j=1}^{n} w_j y_j B_{s}^{k+1}(x_j),$$

V kapitole jsme se zabývali diskrétními daty, nicméně pokud bychom chtěli aproximovat data spojitá, stačilo by si dodefinovat spojitý skalární součin.

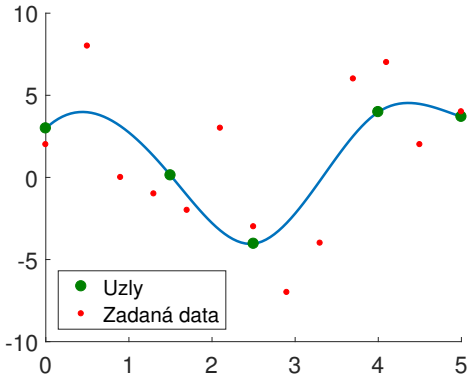
**Poznámka 1.9.** Metoda nejmenších čtverců (zkráceně MNČ) se používá hlavně v případech, kdy máme o hodně vyšší počet daných bodů $x_j$ jako počet uzlů sítě. V případě, kdy $n \leq g + k + 1$, je vhodnější použít interpolaci.

**Příklad 1.3.** Nechť máme v $\mathbb{R}^2$ zadaná data {(0, 2), (0,5, 8), (0,9, 0), (1,3, -1), (1,7, -2), (2,1,3), (2,5, -3), (2,9, -7), (3,3, -4), (3,7, 6), (4,1, 7), (4,5, 2), (5, 4)} a chceme nalézt splajn stupně 3, který by daná data aproximoval ve smyslu MNČ.

Jako síť uzlů si zvolíme 5 bodů {0, 1,5, 2,5, 4, 5 }. Tuto síť v MATLABu rozšíříme násobnými krajními uzly pomocí příkazu `augknt`. Dále sestrojíme B-splajny na rozšířené síti uzlů příkazem `spmak` a vytvoříme kolokační matici pomocí `spcol`. Jelikož je kolokační matice plné sloupcové hodnosti, existuje právě jeden splajn stupně 3, který minimalizuje odchylky čtverců od dat. Z poznatků podkapitoly 1.2.3 pak můžeme nalézt aproximující splajn. Na obrázku 1.4 je znázorněn výsledek.




(a) Použité B-splajny




(b) Polynomiální splajn aproximující data ve smyslu MNČ

Obrázek 1.4: Příklad MNČ

### 1.2.4. Vyhlazování

Pro danou síť uzlů ($\Delta\lambda$), dané $k$, dané body $(x_j, y_j)$, $a \leq x_j \leq b$, váhové koeficienty $w_j > 0$, kde $j = 1, \ldots, n$, a vyhlazující parametr $\alpha \in (0, 1)$ chceme nalézt $s_k(x) \in S_k^{\Delta\lambda}[a, b]$ minimalizující

$$J_l^\alpha(s_k) = (1-\alpha)\int_a^b \left(s_k^{(l)}(x)\right)^2 dx + \alpha \sum_{j=1}^{n} w_j \Big(y_j - s_k(x_j)\Big)^2,$$

kde $l \in \{0, 1, \ldots, k-1\}$. Jedná se o kompromis mezi optimální interpolaci dat a metodou nejmenších čtverců. Pro $\alpha \to 0$ se $s_k(x)$ blíží aproximaci dat, zatímco pro $\alpha \to 1$ se $s_k(x)$ blíží splajnu ve smyslu MNČ.

Splajn $s_k(x) \in S_k^{\Delta\lambda}[a, b]$ lze zapsat jako lineární kombinaci B-splajnů, tudíž úlohu můžeme přepsat na hledání vektoru $\boldsymbol{b}$. Označíme-li si navíc

$$I_1(\boldsymbol{b}) = \int_a^b \left(\frac{\partial^l}{\partial x^l} \sum_{i=-k}^{g} b_i B_i^{k+1}(x)\right)\left(\frac{\partial^l}{\partial x^l} \sum_{j=-k}^{g} b_j B_j^{k+1}(x)\right) dx$$

$$I_2(\boldsymbol{b}) = \sum_{j=1}^{n} w_j \left(y_j - \sum_{i=-k}^{g} b_i B_i^{k+1}(x_j)\right)^2$$

pak původní úlohu můžeme přepsat jako hledání minima výrazu

$$J_l^\alpha(\boldsymbol{b}) = (1-\alpha)I_1(\boldsymbol{b}) + \alpha I_2(\boldsymbol{b})$$

Výraz $I_2(\boldsymbol{b})$ již víme jak rozepsat z předchozí podkapitoly 1.2.3. Budeme se tedy zabývat výrazem $I_1(\boldsymbol{b})$.

$$\begin{aligned} I_1(\boldsymbol{b}) &= \int_a^b \left(\frac{\partial^l}{\partial x^l}\sum_{i=-k}^{g} b_i B_i^{k+1}(x)\right)\left(\frac{\partial^l}{\partial x^l}\sum_{j=-k}^{g} b_j B_j^{k+1}(x)\right)dx = \\ &= \int_a^b \left(\sum_{i=-k}^{g} b_i \frac{\partial^l}{\partial x^l} B_i^{k+1}(x)\right)\left(\sum_{j=-k}^{g} b_j \frac{\partial^l}{\partial x^l} B_j^{k+1}(x)\right)dx = \\ &= \left(\prod_{t=1}^{l}(k+1-t)\right)^2 \int_a^b \left(\sum_{i=-k+l}^{g} b_i^{(l)} B_i^{k+1-l}(x)\right)\left(\sum_{j=-k+l}^{g} b_j^{(l)} B_j^{k+1-l}(x)\right)dx, \end{aligned}$$

kde jsme využili věty 1.4.

Definujeme čtvercovou matici $\boldsymbol{M}_{kl} := \Big(m_{ij}\Big)_{i,j=-k+l,\ldots,g}$ typu $(g+k+1-l)\times(g+k+1-l)$, kde

$$m_{ij} = \int_a^b B_i^{k+1-l}(x) B_j^{k+1-l}(x)dx$$

Tato matice je pásová a pozitivně semidefinitní, protože $B_i^{k+1-l}(x) \geq 0$ a B-splajny $B_{-k+l}^{k+1-l}(x), \ldots, B_g^{k+1-l}(x)$ jsou bázové splajny s konečnými nosiči. $I_1$ můžeme poté maticově přepsat jako

$$I_1(\boldsymbol{b}^{(l)}) = \left(\prod_{t=1}^{l}(k+1-t)\right)^2 \left(\boldsymbol{b}^{(l)}\right)^T \boldsymbol{M}_{kl}\boldsymbol{b}^{(l)}$$

Dále je si možné výraz více zjednodušit, když si definujeme matice

$$\boldsymbol{D}_l = (k+1-l)diag\left\{\frac{1}{\lambda_{i+k+1-l}-\lambda_i},\ i=-(k-l),\dots,g\right\}$$

$$\boldsymbol{L}_l = \begin{pmatrix} -1 & 1 & 0 & 0 & \dots & 0 & 0 \\ 0 & -1 & 1 & 0 & \dots & 0 & 0 \\ 0 & 0 & -1 & 1 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \dots & 1 & 0 \\ 0 & 0 & 0 & 0 & \dots & -1 & 1 \end{pmatrix},$$

kde $\boldsymbol{D}_l$ je diagonální matice typu $(g+k+1-l)\times(g+k+1-l)$ a $\boldsymbol{L}_l$ je typu $(g+k+1-l)\times(g+k+2-l)$. $\boldsymbol{b}^{(l)}$ můžeme poté psát jako

$$\boldsymbol{b}^{(l)} = \boldsymbol{D}_l\boldsymbol{L}_l\boldsymbol{b}^{(l-1)} = \boldsymbol{D}_l\boldsymbol{L}_l\boldsymbol{D}_{l-1}\boldsymbol{L}_{l-1}\boldsymbol{b}^{(l-2)} = \dots = \boldsymbol{D}_l\boldsymbol{L}_l\ \cdots\ \boldsymbol{D}_2\boldsymbol{L}_2\boldsymbol{D}_1\boldsymbol{L}_1\boldsymbol{b}$$

Jestliže si označíme

$$\boldsymbol{S}_l = \boldsymbol{D}_l\boldsymbol{L}_l\boldsymbol{D}_{l-1}\boldsymbol{L}_{l-1}\ \cdots\ \boldsymbol{D}_2\boldsymbol{L}_2\boldsymbol{D}_1\boldsymbol{L}_1,$$

pak

$$I_1 = \Big(\prod_{t=1}^{l}(k+1-t)\Big)^2\Big(\boldsymbol{b}^{(l)}\Big)^T\boldsymbol{M}_{kl}\boldsymbol{b}^{(l)} = \boldsymbol{b}^T\boldsymbol{S}_l^T\boldsymbol{M}_{kl}\boldsymbol{S}_l\boldsymbol{b} = \boldsymbol{b}^T\boldsymbol{N}_{kl}\boldsymbol{b},$$

kde $\boldsymbol{N}_{kl} = \boldsymbol{S}_l^T\boldsymbol{M}_{kl}\boldsymbol{S}_l$. Původní funkci, kterou chceme minimalizovat, lze pak přepsat jako

$$\begin{aligned} J_l^\alpha(\boldsymbol{b}) &= (1-\alpha)\boldsymbol{b}^T\boldsymbol{N}_{kl}\boldsymbol{b} + \alpha\Big(\boldsymbol{y}-\boldsymbol{C}_{k+1}(\boldsymbol{x})\boldsymbol{b}\Big)^T\boldsymbol{W}\Big(\boldsymbol{y}-\boldsymbol{C}_{k+1}(\boldsymbol{x})\boldsymbol{b}\Big) = \\ &= \boldsymbol{b}^T\Big((1-\alpha)\boldsymbol{N}_{kl} + \alpha\boldsymbol{C}_{k+1}^T(\boldsymbol{x})\boldsymbol{W}\boldsymbol{C}_{k+1}(\boldsymbol{x})\Big)\boldsymbol{b} - 2\alpha\boldsymbol{b}^T\boldsymbol{C}_{k+1}^T(\boldsymbol{x})\boldsymbol{W}\boldsymbol{y} + \\ &+ \alpha\boldsymbol{y}^T\boldsymbol{W}\boldsymbol{y} \end{aligned}$$

Jelikož je $J_l^\alpha(\boldsymbol{b})$ opět kvadratickou funkcí proměnné $\boldsymbol{b}$, tak platí, že pokud hessián téhle funkce je pozitivně definitní, pak existuje právě jedno její minimum. Hessián

funkce $J_l^\alpha(\boldsymbol{b})$ je

$$\frac{\partial^2 J_l^\alpha(\boldsymbol{b})}{\partial \boldsymbol{b}^T \partial \boldsymbol{b}} = (1-\alpha)\boldsymbol{N}_{kl} + \alpha \boldsymbol{C}_{k+1}^T(\boldsymbol{x})\boldsymbol{W}\boldsymbol{C}_{k+1}(\boldsymbol{x})$$

Jelikož je $M_{kl}$ pozitivně semidefinitní, pak i $N_{kl}$ je pozitivně semidefinitní. Pozitivní definitnost celého výrazu bude tedy záviset na hodnosti matice $\boldsymbol{C}_{k+1}(\boldsymbol{x})$. Pokud bude matice plné sloupcové hodnosti, tak hessián bude pozitivně definitní.

Při hledání minima postupujeme podobně jako u MNČ. Funkci $J_l^\alpha(\boldsymbol{b})$ zderivujeme podle hledané proměnné a položíme ji rovnou nule.

$$\begin{aligned} 0 = \frac{\partial J_l^\alpha(\boldsymbol{b})}{\partial \boldsymbol{b}^T} &= 2\Big((1-\alpha)\boldsymbol{N}_{kl} + \alpha \boldsymbol{C}_{k+1}^T(\boldsymbol{x})\boldsymbol{W}\boldsymbol{C}_{k+1}(\boldsymbol{x})\Big)\boldsymbol{b} - 2\alpha \boldsymbol{C}_{k+1}^T(\boldsymbol{x})\boldsymbol{W}\boldsymbol{y} \\ \Big((1-\alpha)\boldsymbol{N}_{kl} + \alpha \boldsymbol{C}_{k+1}^T(\boldsymbol{x})\boldsymbol{W}\boldsymbol{C}_{k+1}(\boldsymbol{x})\Big)\boldsymbol{b} &= \alpha \boldsymbol{C}_{k+1}^T(\boldsymbol{x})\boldsymbol{W}\boldsymbol{y} \end{aligned}$$

Na levé straně rovnice je čtvercová matice $(1-\alpha)\boldsymbol{N}_{kl} + \alpha \boldsymbol{C}_{k+1}^T(\boldsymbol{x})\boldsymbol{W}\boldsymbol{C}_{k+1}(\boldsymbol{x})$, kterou si pro jednoduchost označíme jako $\boldsymbol{A}$. Na pravé straně rovnice máme po roznásobení vektor, který si označíme například $\boldsymbol{u}$. Tedy

$$\underbrace{\Big((1-\alpha)\boldsymbol{N}_{kl} + \alpha \boldsymbol{C}_{k+1}^T(\boldsymbol{x})\boldsymbol{W}\boldsymbol{C}_{k+1}(\boldsymbol{x})\Big)}_{\boldsymbol{A}}\boldsymbol{b} = \underbrace{\alpha \boldsymbol{C}_{k+1}^T(\boldsymbol{x})\boldsymbol{W}\boldsymbol{y}}_{\boldsymbol{u}}$$

$$\boldsymbol{A}\boldsymbol{b} = \boldsymbol{u}$$

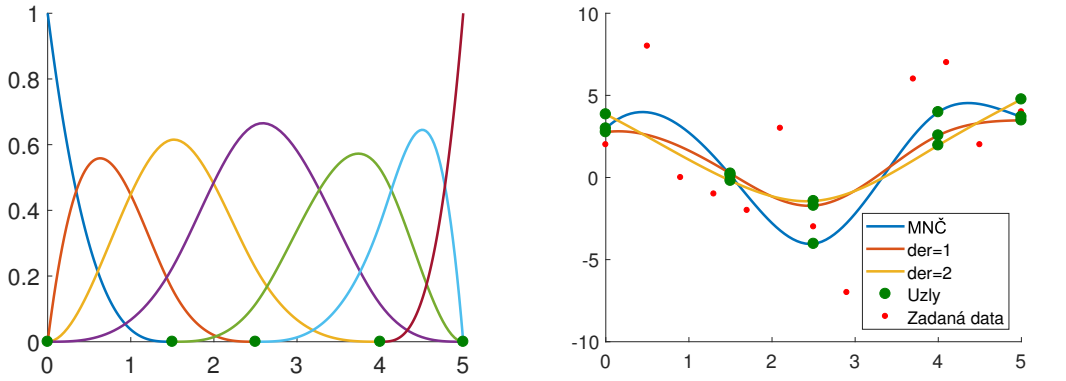
Jestliže bude matice $\boldsymbol{A}$ regulární, pak bude existovat právě jedno řešení soustavy. Toto řešení by vypadalo jako

$$\boldsymbol{b}^* = \boldsymbol{A}^{-1}\boldsymbol{u}$$

Jestliže bude $\boldsymbol{C}_{k+1}(\boldsymbol{x})$ plné sloupcové hodnosti, pak $\boldsymbol{A}$ bude pozitivně definitní, čili regulární, a řešení soustavy bude určeno jednoznačně.

**Příklad 1.4.** Nechť máme v $\mathbb{R}^2$ zadaná data {(0, 2), (0,5, 8), (0,9, 0), (1,3, -1), (1,7, -2), (2,1,3), (2,5, -3), (2,9, -7), (3,3, -4), (3,7, 6), (4,1, 7), (4,5, 2), (5, 4)} a chceme je aproximovat splajnem stupně 3.

Stejná data jsme již aproximovali ve smyslu MNČ v příkladu 1.3. Data tedy nyní aproximujeme pomocí vyhlazování a srovnáme s MNČ. Jako síť uzlů si opět zvolíme 5 bodů $\{0,\ 1{,}5,\ 2{,}5,\ 4,\ 5\}$ a derivace pro srovnání zvolíme 1 a 2. Jelikož je kolokační matice plné sloupcové hodnosti, existuje právě jeden splajn stupně 3, který minimalizuje $J_l^\alpha$. Z poznatků podkapitoly 1.2.4 pak můžeme nalézt aproximující splajn. Pomocí kódu `vyhl.m`, jenž byl poskytnut paní docentkou Machalovou a který lze nalézt v příloze, vypočítáme. Na obrázku 1.5 je znázorněn výsledek.




(a) B-splajny

(b) Srovnání MNČ a vyhlazování pro $l = 1$ a $l = 2$

Obrázek 1.5: Aproximace dat pomocí splajnu stupně $k = 3$

# 2. Ortogonální B-splajny

Konstruovat splajny jako lineární kombinaci nezávislých B-splajnů tvořící bázi daného prostoru nám přináší řadu výhod. Jednou z nich je například vlastnost lokálního nosiče, která nám říká, že jednotlivé B-splajny jsou nenulové pouze na určitých konečných intervalech. Pracujeme-li tedy na užším intervalu, tak stačí brát v potaz pouze ty B-splajny, které jsou na daném intervalu nenulové. Taková báze tvořená B-splajny však obecně není ortogonální. Pokud by byla, tak by se nám práce s ní zjednodušila v případech, kdy bychom využívali spojité skalární součiny. Jelikož jakoukoliv bázi lze zortogonalizovat, je možné uvažovat nad metodami ortogonalizace báze tvořené B-splajny. Různé metody samozřejmě vedou k různým výsledkům. V této kapitole si uvedeme několik takových metod a nakonec jejich výsledky srovnáme.

B-splajny stupně 0 jsou již přirozeně ortogonální, což je dáno disjunktní povahou jejich nosičů. Budeme se proto zajímat pouze o B-splajny stupně vyššího než 0.

Všechny informace této kapitoly byly čerpány z [4].

## 2.1. Gramova-Schmidtova metoda

Gramova-Schmidtova ortogonalizační metoda je jedna z nejznámějších metod ortogonalizace bází. Metoda spočívá v postupné ortogonalizaci prvků (v našem případě B-splajnů) vzhledem k těm předcházejícím. Abychom zortogonalizovali všechny B-splajny, bude tedy vhodné začít v jednom z krajních B-splajnů. V závislosti na tom, který z krajních B-splajnů zvolíme, můžeme metodu nazvat Gramova-Schmidtova metoda zleva doprava či Gramova-Schmidtova metoda zprava doleva. Použití těchto dvou alternativ dosáhneme odlišných výsledků.

Pro úplnost si uvedeme obecnou Gramovu-Schmidtovu metodu zleva doprava. Budeme tedy chtít zortogonalizovat nezávislé funkce $h_1, \ldots, h_n$ z nějakého Hilbertova prostoru $H$. Výsledné ortogonální prvky si označíme $e_1, \ldots, e_n$.

**Algoritmus 2.1.** *Nejprve určíme* $e_1$*:*

$$e_1 = \frac{h_1}{\|h_1\|}$$

*Pro* $i = 1, \ldots, n-1$ *postupně vypočítáme:*

$$\alpha_i = -\beta_{i+1}\langle e_i, h_{i+1}\rangle$$

$$\beta_{i+1} = \frac{1}{\sqrt{\|h_{i+1}\|^2 - \langle e_i, h_{i+1}\rangle^2}}$$

$$e_{i+1} = \alpha_i e_i + \beta_{i+1} h_{i+1}$$

Daný algoritmus lze využít i pro naše účely dosazením B-splajnů za prvky $h_1, \ldots, h_n$, nicméně je nutné nejprve definovat jejich skalární součin. Konkrétně si v případě B-splajnů skalární součin definujeme jako spojitý skalární součin.

$$\langle B_i^{k+1}, B_j^{k+1}\rangle = \int_a^b B_i^{k+1}(x) B_j^{k+1}(x)\ dx$$

**Poznámka 2.1.** Z důvodu vlastnosti lokálního nosiče si stačí skalární součin definovat jako integraci součinu B-splajnů přes interval $[\lambda_i, \lambda_{j+k+1}]$. Tento interval lze ještě více zúžit na průnik intervalů $[\lambda_i, \lambda_{i+k+1}]$ a $[\lambda_j, \lambda_{j+k+1}]$. Mimo průnik těchto intervalů je totiž funkční hodnota alespoň jednoho z daných B-splajnů nulová, tudíž jejich součin se též bude rovnat nule.

**Kód 2.1.** Pro výpočet skalárního součinu dvou B-splajnů jsem v MATLABu vytvořil kód `skalar.m`. Kód je k nalezení v příloze diplomové práce.

**Kód 2.2.** Pro výpočet matice skalárních součinů B-splajnů jsem v MATLABu vytvořil kód `skalarmatice.m`. Kód je k nalezení v příloze diplomové práce.

**Kód 2.3.** Pro ortogonalizaci B-splajnů pomocí Gramovy-Schmidtovy metody zleva doprava jsem v MATLABu vytvořil kódy `gramsch.m` a `gsorto.m`. Kódy jsou k nalezení v příloze diplomové práce.

**Poznámka 2.2.** Přímou interpretací algoritmu 2.1 pro B-splajny je vytvořený kód `gramsch.m`. Použitím pouze tohoto kódu k ortogonalizaci B-splajnů stupně

většího jak 1 sice získáme B-splajny, jež jsou ortogonální s jejich sousedy, nicméně s přímo nesousedícími B-splajny již ortogonální obecně nejsou (respektive jejich skalární součin se pouze blíží nule s tolerancí $10^{-2}$). Proto jsem vytvořil kód `gsorto.m` jež ve svém textu kódu `gramsch.m` využívá. Funguje tak, že po ortogonalizaci všech B-splajnů pomocí funkce `gramsch.m` pak ještě jednotlivě zortogonalizujeme nesousedící dvojice právě zortogonalizovaných B-splajnů pomocí `gramsch.m`. Výsledek je pak sice přesnější, ale zato se nám zvýší výpočetní náročnost. B-splajny stupně 1 takové problémy s neortogonalitou však nemají, tudíž je pro ně výhodnější využít kódu `gramsch.m`.

**Poznámka 2.3.** Kód Gramovy-Schmidtovy metody zprava doleva je totožný s kódem Gramovy-Schmidtovy metody zleva doprava. Jediný rozdíl je způsob zadání B-splajnů na vstupu, kde abychom dostali Gramovu-Schmidtovu metodu zprava doleva, tak místo B-splajnů seřazených zleva doprava zadáme B-splajny seřazené zprava doleva. B-splajn první v pořadí ve formě struct bude tedy B-splajn nejvíce vpravo. Jestliže již máme B-splajny ve formě struct seřazené zleva doprava, lze je v MATLABu seřadit zprava doleva například pomocí funkce `flip`.

**Příklad 2.1.** Nechť máme zadanou síť uzlů $\{0, 1, 2, 3, 4, 5, 6, 7\}$ a na ní definované B-splajny stupně 1: $B_0^2(x), \ldots, B_5^2(x)$. Úkolem je tyto B-splajny vůči sobě zortogonalizovat.

K řešení využijeme MATLAB a vytvořený kód `gsorto.m`. Uzly zapíšeme jako vektor do proměnné `knots` a pomocí funkce `spmak` vytvoříme B-splajny. Ty si zapíšeme do proměnné `s` ve formě struct. Ortogonalizační funkci zavoláme pomocí `orto=gsorto(s)`. Výsledky a srovnání se symetrickou Gramovou-Schmidtovou metodou lze najít v podkapitole 2.2 na obrázku 2.1.

Příklad vypracovaný s pomocí MATLABu lze nalézt v příloze pod názvem `priklad_gs.m`

## 2.2. Symetrická metoda

Nevýhodou jednostranné Gramovy-Schmidtovy metody je asymetrie B-splajnů i pro ekvidistantní uzly a ztráta vlastnosti konečného nosiče. S každým krokem algoritmu se intervaly nosičů rozšiřují. Proto byla navržena takzvaná oboustranná metoda. Princip oboustranné metody je B-splajny jednostranně dvakrát ortogonalizovat kolem zvoleného centrálního bodu $x_c$. Tento bod je obecně uzel. Ten můžeme volit několika způsoby, ale nejvýhodnější je zvolit prostřední uzel celého intervalu (nebo jeden z prostředních uzlů v případě sudého počtu uzlů), na kterém konstruujeme B-splajny. Jestliže je síť uzlů ekvidistantní, je možné jako centrální bod zvolit právě prostřední bod celého intervalu, který ani uzlem být nemusí.

Konkrétně můžeme použít Gramovu-Schmidtovu metodu zleva doprava na B-splajny vlevo od centrálního bodu $x_c$ a Gramovu-Schmidtovu metodu zprava doleva na B-splajny vpravo od $x_c$, přičemž přeskočíme B-splajny, jež mají $x_c$ ve svém nosiči. Výsledkem budou dvě skupiny B-splajnů. Skupinu B-splajnů vlevo od centrálního bodu si pro jednoduchost označíme LB-splajny a skupinu vpravo si označíme RB-splajny.

Samozřejmě se ještě budeme muset vypořádat s B-splajny, které jsme přeskočili. Označíme si je například $s_1, \ldots, s_r$. Po těch budeme chtít, aby po zortogonalizování byly pro ekvidistantní síť uzlů symetrické kolem centrálního bodu. Po B-splajnech na neekvidistantní síti uzlů vlastnost symetrie požadovat nebudeme.

Existuje více přístupů, jak takové ortogonalizace prostředních B-splajnů dosáhnout. Jedním z nich je s pomocí následující věty.

**Věta 2.1.** *Nechť $S$ je symetrický operátor na lineárním prostoru a nechť $x$ a $y$ jsou lineárně nezávislé funkce z nějakého Hilbertova prostoru, pro něž platí*

*$y = Sx$. Definujme:*

$$\widetilde{x} = \left(\frac{1}{\sqrt{1+\langle x,y\rangle}} + \frac{1}{\sqrt{1-\langle x,y\rangle}}\right)\frac{x}{2} + \left(\frac{1}{\sqrt{1+\langle x,y\rangle}} - \frac{1}{\sqrt{1-\langle x,y\rangle}}\right)\frac{y}{2}$$

$$\widetilde{y} = \left(\frac{1}{\sqrt{1+\langle x,y\rangle}} - \frac{1}{\sqrt{1-\langle x,y\rangle}}\right)\frac{x}{2} + \left(\frac{1}{\sqrt{1+\langle x,y\rangle}} + \frac{1}{\sqrt{1-\langle x,y\rangle}}\right)\frac{y}{2}$$

*Pak funkce $\widetilde{x}$ a $\widetilde{y}$ jsou ortonormální a symetrické. Tedy $\langle \widetilde{x}, \widetilde{y}\rangle = 0$ a $S\widetilde{x} = \widetilde{y}$.*

**Kód 2.4.** Jako interpretaci věty 2.1 jsem v MATLABu vytvořil kód `symlemma.m`. Kód je k nalezení v příloze diplomové práce.

Než větu 2.1 využijeme, rozdělíme si $s_1, \ldots, s_r$ do dvojic, přičemž spárujeme první a poslední B-splajn, druhý a předposlední, atd. První pár $(s_1, s_r)$ zortogonalizujeme vůči již zortogonalizovaným B-splajnům, čímž dostaneme dvojici $(x_1, x_r)$. Tu poté symetricky zortonormalizujeme pomocí Věty 2.1 a přidáme do odpovídajících skupin zortogonalizovaných B-splajnů. První z dvojice tedy přiřadíme mezi LB-splajny a druhý z dvojice mezi RB-splajny. Takto postupně pokračujeme s následujícími dvojicemi $(s_i, s_{r-i+1})$, $i = 2, \ldots, \left\lfloor \frac{r}{2} \right\rfloor$, až dokud nám nezbyde žádný nebo pouze jeden B-splajn. Ve druhém případě zbývající B-splajn pouze zortogonalizujeme vůči všem již zortogonalizovaným B-splajnům.

Kombinací Gramovy-Schmidtovy metody se symetrizací párů dostaneme symetrickou Gramovu-Schmidtovu metodu. Její postup lze shrnout v následující větě.

**Věta 2.2.** *Nechť $\boldsymbol{x} = (x_i)_{i=1}^{2n}$ je posloupnost lineárně nezávislých funkcí. Definujme $\boldsymbol{x}^R$ a $\boldsymbol{x}^L$ jako*

$$\boldsymbol{x}^L = (x_1^L, x_{2n}^L, x_2^L, x_{2n-1}^L, \ldots, x_n^L, x_{n+1}^L) = gs(x_1, x_{2n}, x_2, x_{2n-1}, \ldots, x_n, x_{n+1})$$

$$\boldsymbol{x}^R = (x_{2n}^R, x_1^R, x_{2n-1}^R, x_2^R, \ldots, x_{n+1}^R, x_n^R) = gs(x_{2n}, x_1, x_{2n-1}, x_2, \ldots, x_{n+1}, x_n)$$

*kde $gs(\ldots)$ značí aplikaci Gramovy-Schmidtovy metody na dané funkce. Dále definujme posloupnost funkcí $\boldsymbol{y} = (y_i)_{i=1}^{2n}$ tak, že každá dvojice $(y_i, y_{2n-i+1})$ vznikne*

*zortonormalizováním dvojic* $(x_i^L, x_{2n-i+1})$ *pomocí Věty 2.1. Potom* $\boldsymbol{y}$ *je posloupnost symetrických ortogonálních funkcí.*

**Poznámka 2.4.** Věta 2.2 platí pouze pro posloupnost funkcí, jejíž počet je sudý. Pro lichý počet funkcí je potřeba větu upravit, což nám ukazuje následující věta 2.3.

**Věta 2.3.** *Nechť* $\boldsymbol{x} = (x_i)_{i=1}^{2n+1}$ *je posloupnost lineárně nezávislých funkcí. Definujme* $\boldsymbol{x}^R$ *a* $\boldsymbol{x}^L$ *jako*

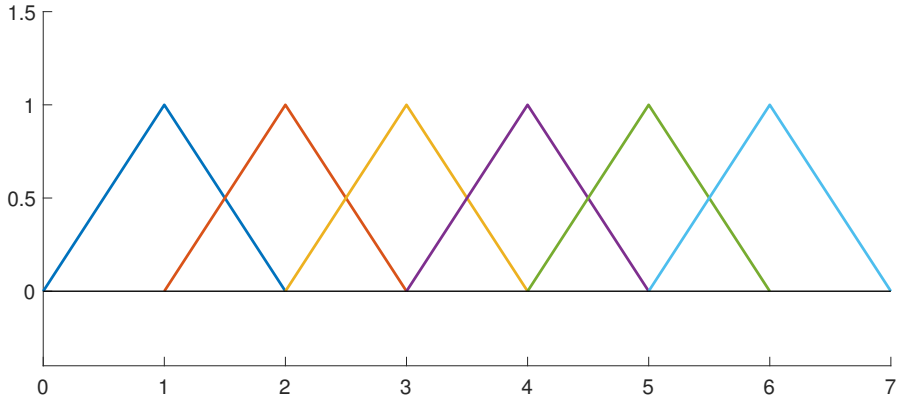
$$\boldsymbol{x}^L = (x_1^L, x_{2n}^L, x_2^L, x_{2n-1}^L, \ldots, x_n^L, x_{n+2}^L, x_{n+1}^L) = gs(x_1, x_{2n}, \ldots, x_n, x_{n+2}, x_{n+1})$$
$$\boldsymbol{x}^R = (x_{2n}^R, x_1^R, x_{2n-1}^R, x_2^R, \ldots, x_{n+1}^R, x_n^R) = gs(x_{2n}, x_1, x_{2n-1}, x_2, \ldots, x_{n+1}, x_n)$$

*kde* $gs(\ldots)$ *značí aplikaci Gramovy-Schmidtovy metody na dané funkce. Dále definujme posloupnost funkcí* $\boldsymbol{y} = (y_i)_{i=1}^{2n}$ *tak, že každá dvojice* $(y_i, y_{2n-i+1})$ *vznikne zortonormalizováním dvojic* $(x_i^L, x_{2n-i+1})$ *pomocí Věty 2.1. Prostřední funkci definujme jako* $y_{k+1} = x_{k+1}^L$. *Potom* $\boldsymbol{y}$ *je posloupnost symetrických ortogonálních funkcí.*

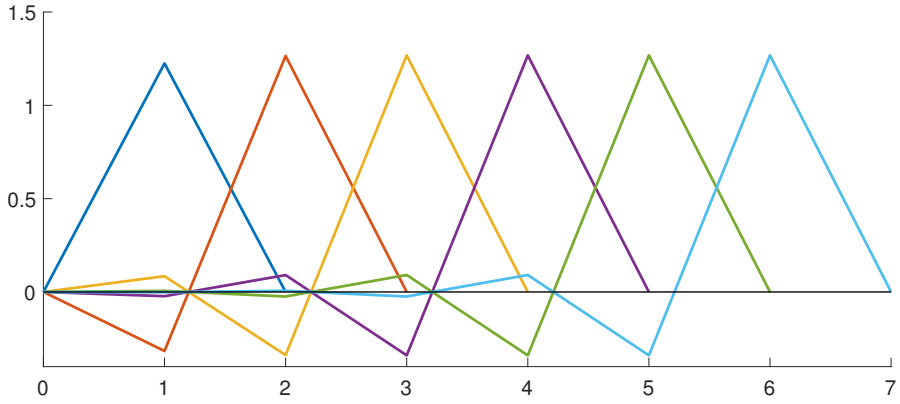
**Kód 2.5.** Pro ortogonalizaci B-splajnů pomocí symetrické Gramovy-Schmidtovy metody jsem v MATLABu vytvořil kód `gssymcor.m`. Kód funguje pro oba případy lichého či sudého počtu B-splajnů na vstupu. Kód je k nalezení v příloze diplomové práce.

**Příklad 2.2.** Nechť máme zadanou síť uzlů $\{0, 1, 2, 3, 4, 5, 6, 7\}$ a na ní definované B-splajny stupně 1: $B_0^2(x), \ldots, B_5^2(x)$. Úkolem je tyto B-splajny vůči sobě zortogonalizovat.

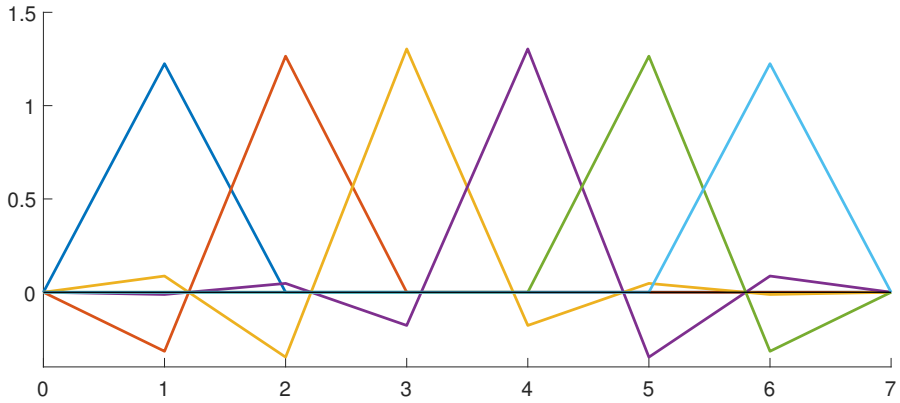
K řešení využijeme MATLAB a vytvořený kód `gssymcor.m`. Uzly zapíšeme jako vektor do proměnné `knots` a pomocí funkce `spmak` vytvoříme B-splajny. Ty si zapíšeme do proměnné `s` ve formě struct. Ortogonalizační funkci zavoláme pomocí `symorto=gssymcor(s)`. Do `symorto` se nám pak ve formě struct zapíší výsledné ortogonální B-splajny. Výsledky a srovnání s obecnou Gramovou-Schmidtovou metodou lze najít na obrázku 2.1.




(a) B-splajny stupně 1




(b) Zortogonalizované B-splajny po použití Gramovy-Schmidtovy metody




(c) Zortogonalizované B-splajny po použití symetrické Gramovy-Schmidtovy metody

Obrázek 2.1: Srovnání obecné a symetrické Gramovy-Schmidtovy metody

Jak lze vidět, nosiče B-splajnů zortogonalizovaných pomocí Gramovy-Schmidtovy metody zleva doprava se postupně rozšiřují, čím více se blížíme k poslednímu B-splajnu. Nosiče zortogonalizovaných B-splajnů pomocí symetrické metody se naopak rozšiřují čím více se blížíme ke středu intervalu, od středu ke kraji se naopak zužují. V průměru jsou však intervaly nosičů užší po použití symetrické Gramovy-Schmidtovy metody. Jelikož je síť uzlů ekvidistantní, tak jsou navíc zortogonalizované B-splajny pomocí symetrické Gramovy-Schmidtovy metody symetrické vzhledem k prostřednímu bodu intervalu.

Příklad vypracovaný s pomocí MATLABu lze nalézt v příloze pod názvem `priklad_gs.m`.

# 3. Splinety

Termín splinet poprvé zazněl v [4]. Jedná se o skupinu B-splajnů zortogonalizovanou určitým způsobem, kde je hlavní důraz kladen na to, aby se po jejich zortogonalizování strukturou co nejvíce podobaly původním B-splajnům.

Cílem symetrické ortogonalizace bylo co nejvíce zredukovat velikosti intervalů nosičů jednotlivých B-splajnů a zachovat případnou symetrii. Chtěli bychom, aby tyto vlastnosti měly i výsledné B-splajny ve splinetech.

Konstrukce splinetů bývá složitější čím vyššího stupně dosahují původní B-splajny. V této kapitole si proto nejprve předvedeme konstrukci na B-splajnech stupně 1 a až poté se podíváme na B-splajny obecného stupně $k$.

Všechny informace této kapitoly byly čerpány z [4].

**Poznámka 3.1.** Pro jednoduchost budeme výsledné zortogonalizované B-splajny ve splinetu značit jako OB-splajny.

## 3.1. Splinety stupně 1

Konstrukci splinetů stupně 1 lze nejjednodušeji provést na síti uzlů $(\Delta\lambda)$, pro kterou platí, že má $g = 2^N - 1$ vnitřních uzlů, kde $N \in \mathbb{N}$. Předpokládáme navíc, že žádný z uzlů není násobný. Tudíž

$$(\Delta\lambda) : \lambda_0 < \lambda_1 < \cdots < \lambda_{2^N-1} < \lambda_{2^N}$$

Takovému případu budeme říkat úplný dyadický a interval $[\lambda_0, \lambda_{2^N}]$ nazveme dyadický interval.

### 3.1.1. Dyadický případ

Nechť máme síť uzlů $(\Delta\lambda)$, jejíž počáteční a koncový bod tvoří dyadický interval. V triviálním případě $N = 1$ se na síti nachází pouze jeden B-splajn. Ten stačí pouze znormalizovat, čímž dostaneme žádaný splinet. Dále budeme tedy předpokládat, že $N > 1$.

Daná síť uzlů nám přirozeně rozděluje interval $[a, b]$ na podintervaly. Jelikož nosiče původních B-splajnů a OB-splajnů jsou obecně rozdílné, budeme chtít předem najít takové intervaly, které budou nosiči výsledných ortogonálních B-splajnů. Nazveme je nosiče úrovně $s$, kde $s \in \{0, 1, \ldots, N-1\}$. Nejmenší nosiče budou úrovně 0. Ty se jednotlivě skládají ze dvou sousedních intervalů, jejichž počáteční a koncové body jsou sousední uzly, počínaje intervaly nejvíce vlevo. Nutno podotknout, že každé dva nosiče stejné úrovně jsou disjunktní. První nosič úrovně 0 se tedy bude skládat z prvního a druhého intervalu sítě, druhý nosič ze třetího a čtvrtého intervalu, atd. Takto můžeme pokračovat až po nosič s pořadovým číslem $2^{N-1}$, který se bude skládat z předposledního a posledního intervalu. Tyto nosiče úrovně 0 označíme $I_{r,0}$, kde $r = 1, \ldots, 2^{n-1}$. Konkrétně $I_{r,0} = (\lambda_{2r-2}, \lambda_{2r}]$, přičemž $\lambda_{2r-1}$ nazveme centrální uzel, což je společný uzel sjednocených intervalů.

Nosiče obecné úrovně $s$ si označíme $I_{r,s}$. Levý krajní uzel nosiče si označíme $\lambda^L_{r,s}$ a pravý krajní uzel si označíme $\lambda^R_{r,s}$. Důležitou roli bude hrát i prostřední uzel intervalu, který si označíme $\lambda^C_{r,s}$. Tudíž $I_{r,s} = (\lambda^L_{r,s}, \lambda^R_{r,s}]$, kde $r = 1, \ldots, 2^{N-s-1}$. Pro $s = 0$ tedy bude platit $\lambda^L_{r,0} = \lambda_{2r-2}$, $\lambda^C_{r,0} = \lambda_{2r-1}$ a $\lambda^R_{r,0} = \lambda_{2r}$.

Pro nosiče vyšší úrovně postupujeme rekurentně. Nechť tedy máme definované nosiče obecné úrovně $m$. Nosiče úrovně $m+1$ dostaneme sjednocením dvou sousedních nosičů úrovně $m$ počínaje levými krajními nosiči. Nosiče budeme sjednocovat podobným způsobem, jako jsme sjednocovali intervaly na vytvoření nosičů úrovně 0. Sjednocení prvního a druhého nosiče úrovně $m$ bude tvořit první nosič úrovně $m+1$, sjednocení třetího a čtvrtého nosiče úrovně $m$ bude tvořit druhý nosič úrovně $m+1$, atd. Výsledné nosiče úrovně $m+1$ tedy budou opět disjunktní. Takto můžeme pokračovat pro všechna $s$, až dokud nedosáhneme posledního nosiče úrovně $N-1$, který bude pokrývat celý dyadický interval. Obecně pak platí $\lambda^L_{r,s} = \lambda_{(2r-2)2^s}$, $\lambda^C_{r,s} = \lambda_{(2r-1)2^s}$ a $\lambda^R_{r,s} = \lambda_{(2r)2^s}$.

Jakmile máme nalezené nosiče pro výsledné OB-splajny, budeme chtít každému takovému nosiči přiřadit právě jeden původní B-splajn. Toho dosáhneme porovnáním prostředních uzlů jednotlivých nosičů OB-splajnů s prostředními uzly

B-splajnů. Jestliže se takový uzel intervalu $I_{r,s}$ s prostředním uzlem nějakého B-splajnu shoduje, daný B-splajn k intervalu přiřadíme a označíme ho $B_{r,s}$.

Ortonormalizace takto označených B-splajnů probíhá následovně. Začneme B-splajny na nejmenších nosičích, což budou B-splajny na nosičích úrovně 0. Konkrétně se tedy bude jednat o B-splajny $B_{r,0}$, kde $r = 1, \ldots, 2^{N-1}$. Tyto B-splajny pouze jednotlivě znormalizujeme a označíme je jako výsledné OB-splajny $OB_{r,0}$.

B-splajny na vyšších úrovních $s \geq 1$ (přičemž budeme po úrovních postupovat vzestupně) pak postupně zortogonalizujeme vůči všem již zortogonalizovaným OB-splajnům, jejichž nosiče se celé nachází v intervalu $I_{r,s}$. Z důvodu dyadické struktury sítě je však stačí zortogonalizovat vůči těm OB-splajnům, jejichž nosiče mají s nosiči právě ortogonalizovaných B-splajnů neprázdný průnik. Na každé předchozí úrovni jsou takové OB-splajny dva. Obecně tedy stačí B-splajn $B_{r,s}$, kde $s \geq 1$, zortogonalizovat pouze vůči $2^s$ OB-splajnům.

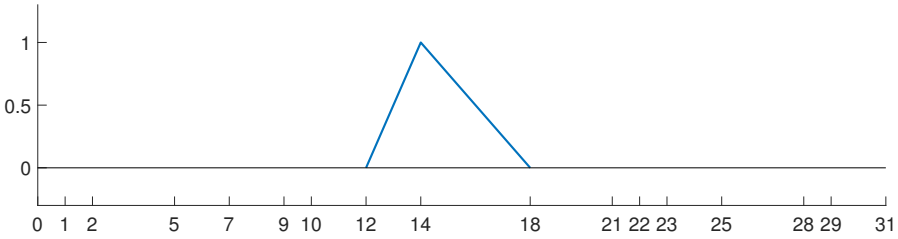
**Kód 3.1.** Pro ortogonalizaci B-splajnů stupně 1 na dyadickém intervalu pomocí splinetů jsem v MATLABu vytvořil kód `splinet1.m`. Kód je k nalezení v příloze diplomové práce.

**Příklad 3.1.** Nechť máme zadanou neekvidistantní dyadickou síť uzlů, jež má $2^4 - 1$ vnitřních uzlů. Naším úkolem je na této síti vytvořit ortogonální B-splajny stupně 1.

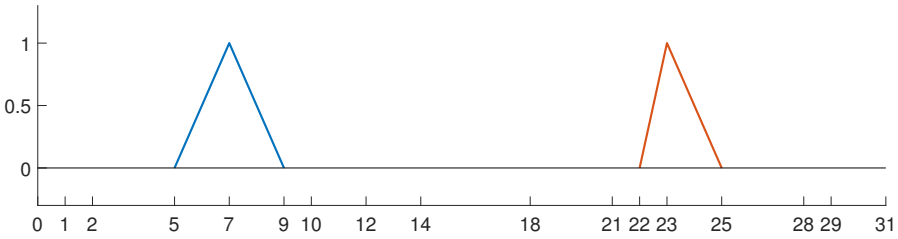
K řešení využijeme MATLAB a vytvořený kód `splinet1.m`. Uzly zapíšeme jako vektor do proměnné `knots` a funkci zavoláme pomocí `O1=splinet1(knots)`. Do `O1` se nám pak ve formě struct zapíší výsledné OB-splajny.

Původní B-splajny rozdělené do úrovní lze vidět na obrázku 3.1. Výsledné ortogonální OB-splajny jsou pak na obrázku 3.2.

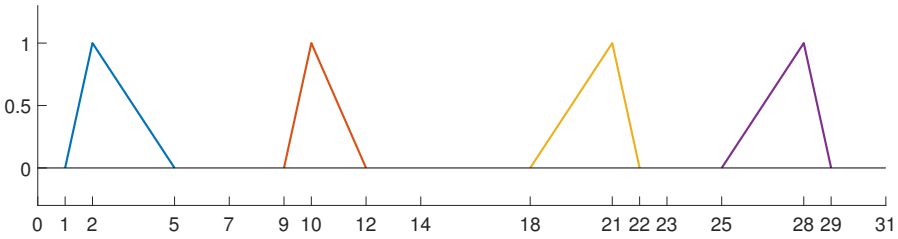
Příklad vypracovaný s pomocí MATLABu lze nalézt v příloze pod názvem `priklad_daydic_1.m`.



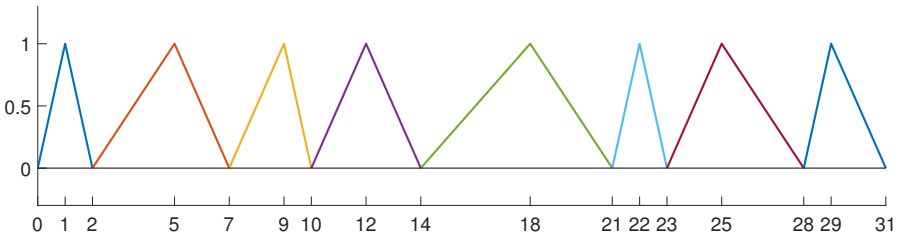

B-splajny na úrovni 3




B-splajny na úrovni 2

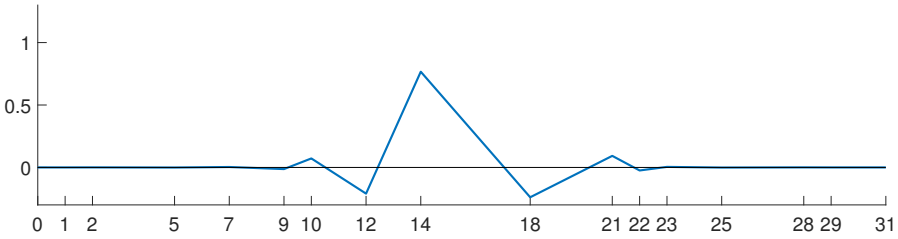



B-splajny na úrovni 1

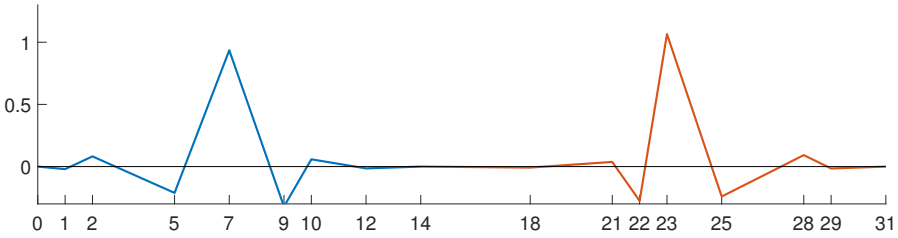



B-splajny na úrovni 0

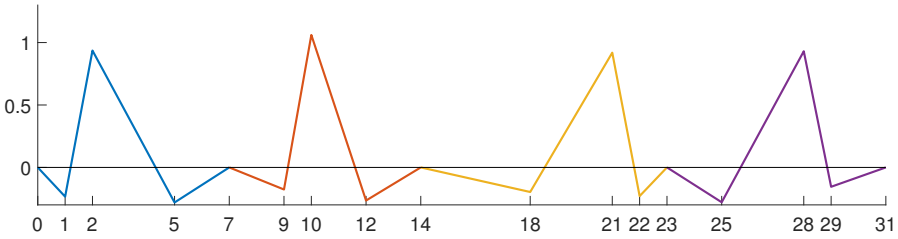
Obrázek 3.1: Rozdělení B-splajnů do úrovní



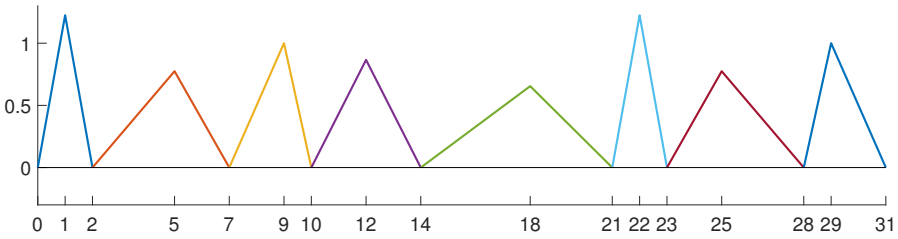

OB-splajny na úrovni 3




OB-splajny na úrovni 2




OB-splajny na úrovni 1




OB-splajny na úrovni 0

Obrázek 3.2: Ortogonální B-splajny - dyadický případ

### 3.1.2. Obecný případ

Jestliže se počet vnitřních uzlů $g$ v síti nerovná $2^N - 1$ pro žádné $N \in \mathbb{N}$, pak se jedná o obecný případ. Můžeme říci, že $2^N \leq g < 2^{N+1} - 1$, kde $N \in \mathbb{N}$. Pro $g = 2^{N+1} - 1$ bychom opět dostali dyadický případ.

Provedení ortogonalizace obecného případu lze rozdělit do čtyř kroků:

1. Rozdělíme původní síť uzlů na co největší dyadické sousední disjunktní intervaly tak, aby byly symetrické vzhledem k prostřednímu uzlu (nebo vzhledem k prostřednímu bodu intervalu mezi prostředními dvěma uzly v případě sudého počtu uzlů) a aby se jejich velikosti postupně zužovali čím více bychom se blížili ke středu. Největší takové intervaly tedy pokrývají krajní uzly. Takto pokračujeme, dokud nám nezbyde prostřední interval, který obsahuje pouze 2 až 4 uzly.

2. Na jednotlivých intervalech získaných v předchozím kroku zkonstruujeme dyadické splinety pomocí postupu uvedeného v podkapitole 3.1.1.

3. Mezi každými dvěma sousedními dyadickými splinety se nachází takzvané nepravidelné B-splajny, jejichž nosiče jsou součástí dvou vedlejších dyadických intervalů. Tyto B-splajny vůči těmto splinetům zortogonalizujeme. Je nutné však podotknout, že je stačí zortogonalizovat vůči těm OB-splajnům ve splinetech, které mají neprázdné průniky nosičů s právě ortogonalizovanými nepravidelnými B-splajny. Dále je zortogonalizujeme vůči všem předchozím nepravidelným OB-splajnům. Opět je však stačí zortogonalizovat vůči těm nepravidelným OB-splajnům, jejichž průniky nosičů s nosiči právě ortogonalizaných B-splajnů jsou neprázdné. Stačí je tedy zortogonalizovat vůči posledně zortogonalizovanému nepravidelnému OB-splajnu vlevo či vpravo.

4. Po provedení prvních tří kroků nám zbývá zortogonalizovat 1 až 4 prostředních B-splajnů. V závislosti na tom, kolik jich zbývá, provedeme jeden z následujících kroků:

(a) Pokud zbývá pouze jeden B-splajn, tak ho zortogonalizujeme vůči posledně vytvořeným splinetům vlevo a vpravo a vůči posledně zortogonalizovaným nepravidelným OB-splajnům vlevo a vpravo. Pokud žádné takové splinety či nepravidelné OB-splajny neexistují, B-splajn pouze znormalizujeme.

(b) Pokud zbývají dva B-splajny, tak levý B-splajn zortogonalizujeme vůči posledně vytvořenému splinetu vlevo a posledně zortogonalizovanému OB-splajnu vlevo a pravý B-splajn vůči posledně vytvořenému splinetu vpravo a posledně zortogonalizovanému nepravidelnému OB-splajnu vpravo. Dále tyto dva zbývající B-splajny ještě symetricky zortogonalizujeme.

(c) Pokud zbývají tři B-splajny, tak levý B-splajn zortogonalizujeme vůči posledně vytvořenému splinetu vlevo a posledně zortogonalizovanému nepravidelnému OB-splajnu vlevo a pravý vůči posledně vytvořenému splinetu vpravo a posledně zortogonalizovanému nepravidelnému OB-splajnu vpravo. Prostřední B-splajn pak zortogonalizujeme vůči těmto dvěma právě zortogonalizovaným B-splajnům.

(d) Pokud zbývají čtyři B-splajny, tak B-splajn nejvíce vlevo zortogonalizujeme vůči posledně vytvořenému splinetu vlevo a posledně zortogonalizovanému nepravidelnému OB-splajnu vlevo a B-splajn nejvíce vpravo vůči posledně vytvořenému splinetu vpravo a posledně zortogonalizovanému nepravidelnému OB-splajnu vpravo. Zbydou nám dva prostřední B-splajny. Levý z nich zortogonalizujeme vůči právě zortogonalizovanému B-splajnu nejvíce vlevo a pravý vůči právě zortogonalizovanému B-splajnu nejvíce vpravo. Tyto poslední dva B-splajny navíc ještě symetricky zortogonalizujeme.

**Poznámka 3.2.** Jestliže má původní síť uzlů pouze 4 uzly, tak B-splajny na dané síti stačí pouze symetricky zortogonalizovat.

**Kód 3.2.** Pro ortogonalizaci B-splajnů stupně 1 na nedyadickém intervalu po-

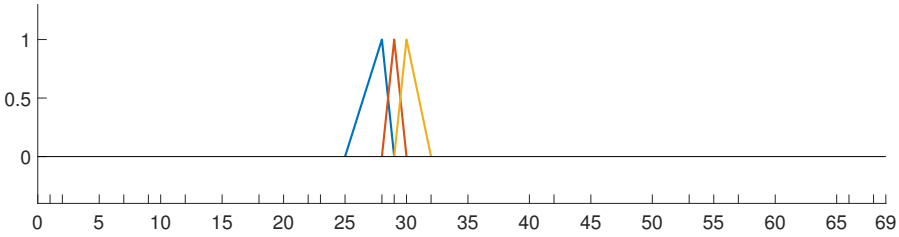
mocí splinetů jsem v MATLABu vytvořil kód `splinet2.m`. Kód je k nalezení v příloze diplomové práce.

**Příklad 3.2.** Nechť máme zadanou neekvidistantní nedyadickou síť uzlů, jež má $2^4+13$ vnitřních uzlů. Naším úkolem je na této síti vytvořit ortogonální B-splajny stupně 1.

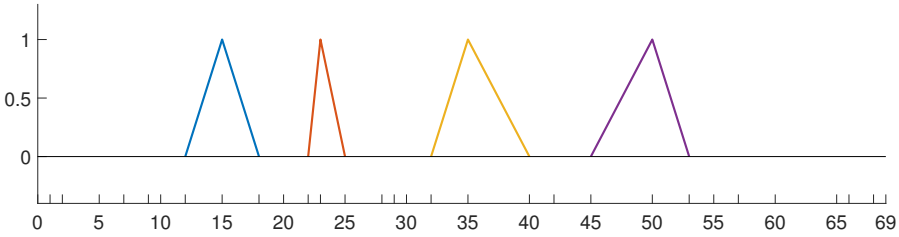
K řešení využijeme MATLAB a vytvořený kód `splinet2.m`. Uzly zapíšeme jako vektor do proměnné `knots` a funkci zavoláme pomocí `O2=splinet2(knots)`. Do `O2` se nám pak ve formě struct zapíší výsledné OB-splajny.

Původní B-splajny rozdělené do úrovní lze vidět na obrázku 3.3. Výsledné ortogonální OB-splajny jsou pak na obrázku 3.4.

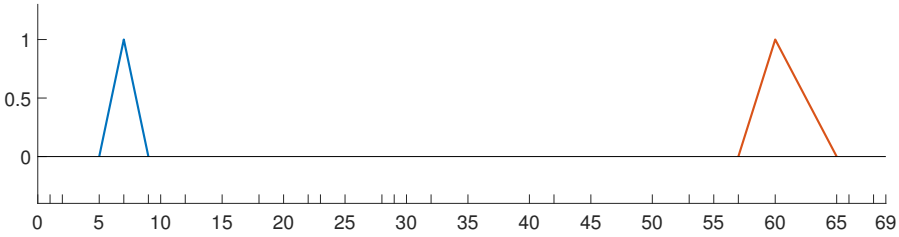
Příklad vypracovaný s pomocí MATLABu lze nalézt v příloze pod názvem `priklad_nd_1.m`.



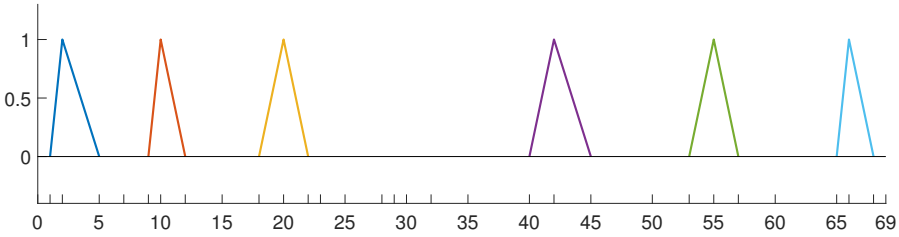

Prostřední B-splajny



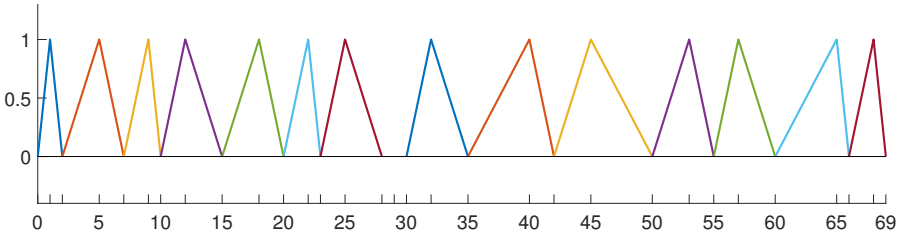

Nepravidelné B-splajny mezi splinety




B-splajny na úrovni 2

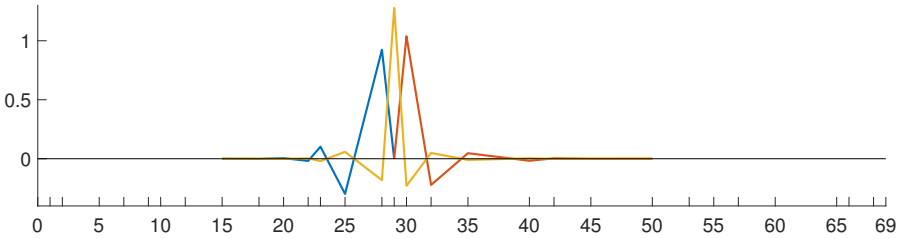



B-splajny na úrovni 1

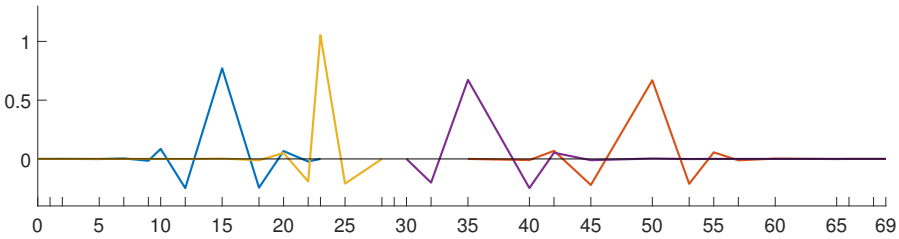



B-splajny na úrovni 0

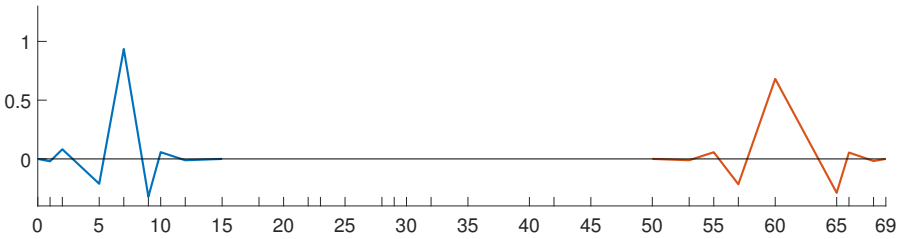
Obrázek 3.3: Zortogonalizované B-splajny stupně 1 - obecný případ



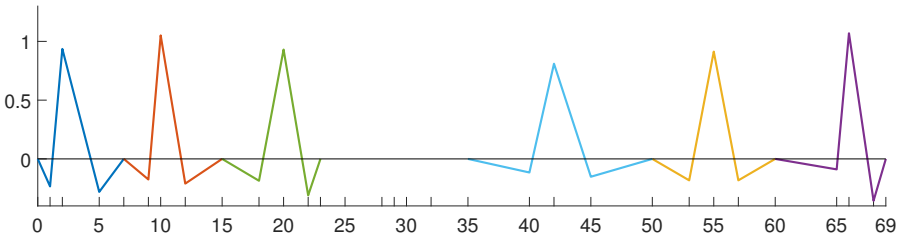

Prostřední OB-splajny



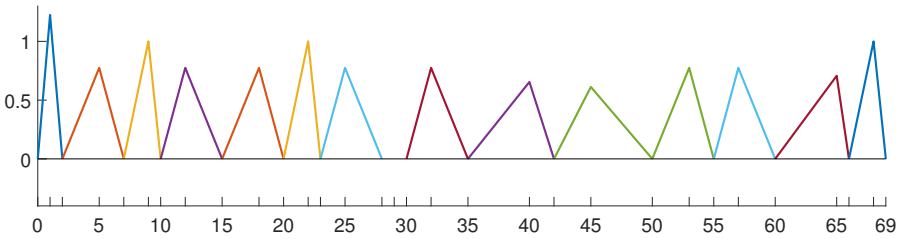

Nepravidelné OB-splajny mezi splinety




OB-splajny na úrovni 2




OB-splajny na úrovni 1




OB-splajny na úrovni 0

Obrázek 3.4: Zortogonalizované B-splajny stupně 1 - obecný případ

## 3.2. Splinety stupně k

Princip tvoření splinetů stupně $k$ spočívá v tom, že si původní B-splajny roztřídíme do několika skupin, s kterými budeme pracovat jako s jedním celkem. Jako nosiče splajnů v jedné skupině tedy budeme brát sjednocení všech nosičů v daném celku. Po rozdělení do celků budeme postupovat podobně jako u splinetů stupně 1.

Konstrukci splinetů stupně $k$ lze nejjednodušeji provést na síti uzlů $(\Delta\lambda)$, pro kterou platí, že má $g = k2^N - 1$ vnitřních uzlů, kde $N \in \mathbb{N}$. Předpokládáme navíc, že žádný z uzlů není násobný. Tudíž

$$(\Delta\lambda) : \lambda_0 < \lambda_1 < \cdots < \lambda_{k2^N-1} < \lambda_{k2^N}$$

Takovému případu budeme říkat úplný dyadický a interval $[\lambda_0, \lambda_{k2^N}]$ nazveme dyadický interval.

### 3.2.1. Dyadický případ

Nejprve původní B-splajny rozdělíme do již zmíněných celků, přičemž každý celek bude obsahovat právě $k$ sousedních B-splajnů. Každý takto rozdělený B-splajn musí patřit právě do jednoho celku. Abychom rozdělili všechny B-splajny do celků, začneme $k$ krajními B-splajny. Celků dohromady bude $2^N - 1$.

Nosiče výsledných ortogonálních celků budou opět rozděleny do úrovní. Nejmenší takové nosiče budou úrovně 0. Jestliže celky očíslujeme zprava doleva od jedničky po $2^N - 1$, tak právě nosiče celků s lichým číslem budou nosiče úrovně 0. Tyto nosiče označíme jako intervaly $I_{r,0}$, kde $r = 1, \ldots, 2^{N-1}$. Konkrétně $I_{r,0} = (\lambda_{2k(r-1)}, \lambda_{2kr}]$, přičemž $\lambda_{2kr-k}$ je centrální uzel.

Nosiče úrovně $s \geq 1$, $s \in \mathbb{N}$ dostaneme jako sjednocení dvou sousedních nosičů úrovně $s - 1$, počínaje levými krajními nosiči. Tedy první nosič úrovně $s$ bude sjednocením prvního a druhého nosiče úrovně $s - 1$, druhý nosič úrovně $s$ bude sjednocením třetího a čtvrtého nosiče úrovně $s - 1$, atd. Konkrétně $I_{r,s} = (\lambda_{2k(r-1)2^s}, \lambda_{(2kr)2^s}]$, kde $r = 1, \ldots, 2^N - s - 1$. Nutno podotknout, že každé

dva nosiče stejné úrovně budou disjunktní. Takto budeme pokračovat až dokud nedostaneme poslední nosič úrovně $N-1$, který bude pokrývat celý dyadický interval.

Jakmile máme nosiče pro výsledné ortogonální celky, budeme chtít každému původnímu celku přiřadit právě jeden takový nosič. Toho dosáhneme porovnáním prostředních uzlů jednotlivých nosičů pro ortogonální celky s prostředními uzly celků. Jestliže se prostřední uzel intervalu $I_{r,s}$ s prostředním uzlem nějakého celku shoduje, B-splajny v daném celku postupně označíme $B_{t,s}$, kde $t = kr - k + \ 1, \ldots, kr$.

Ortogonalizace probíhá následovně. Nejprve si vezmeme celky na nejmenších nosičích, což budou celky na nosičích úrovně 0. Konkrétně se tedy bude jednat o B-splajny $B_{t,0}$. B-splajny v jednotlivých takových celcích symetricky zortogonalizujeme pomocí symetrické Gramovy-Schmidtovy metody a označíme jako OB-splajny $OB_{t,0}$.

B-splajny na vyšších úrovních $s \geq 1$ v jednom celku pak postupně zortogonalizujeme vůči všem OB-splajnům, jejichž nosiče úrovní se celé nachází v intervalu $I_{r,s}$, a symetricky je zortogonalizujeme vůči samy sobě. Z důvodu dyadické struktury sítě však stačí B-splajny ortogonalizovat vůči těm OB-splajnům, jejichž nosiče mají s nosiči právě ortogonalizovaného celku neprázdný průnik. Na každé předchozí úrovni je takových OB-splajnů $2k$. Obecně tedy stačí B-splajny na vyšších úrovních zortogonalizovat pouze vůči $(2k)^s$ OB-splajnům.

**Kód 3.3.** Pro ortogonalizaci B-splajnů obecného stupně $k$ na dyadickém intervalu pomocí splinetů jsem v MATLABu vytvořil kód `splinet_k_dyadic.m`. Kód je k nalezení v příloze diplomové práce.

**Příklad 3.3.** Nechť máme zadanou ekvidistantní dyadickou síť uzlů, jež má $(3 \times 2^4 - 1)$ vnitřních uzlů. Naším úkolem je na této síti vytvořit ortogonální B-splajny stupně 3.

K řešení využijeme MATLAB a vytvořený kód `splinet_k_dyadic.m`. Uzly zapíšeme jako vektor do proměnné `knots` a funkci zavoláme pomocí

`OKD=splinet_k_dyadic(knots,3)`. Do `OKD` se nám pak ve formě struct zapíší výsledné OB-splajny.

Původní B-splajny rozdělené do úrovní lze vidět na obrázku 3.5. Výsledné ortogonální OB-splajny jsou pak na obrázku 3.6.

Příklad vypracovaný s pomocí MATLABu lze nalézt v příloze pod názvem `priklad_dyadic_k.m`.

### 3.2.2. Obecný případ

Jestliže se počet vnitřních uzlů $g$ v síti nerovná $k2^N - 1$ pro žádné $N \in \mathbb{N}$, pak se jedná o obecný případ. Můžeme říct, že $k2^N \leq g < k2^{N+1} - 1$ pro $N \in \mathbb{N}$. Pro $g = 2^{N+1} - 1$ bychom opět měli dyadický případ.

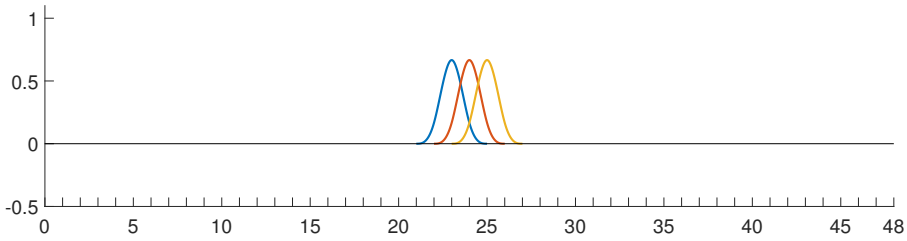
Provedení ortogonalizace obecného případu lze rozdělit do čtyř kroků:

1. Rozdělíme původní síť uzlů na co největší dyadické sousední disjunktní intervaly tak, aby byly symetrické vzhledem k prostřednímu uzlu (nebo vzhledem k prostřednímu bodu intervalu mezi prostředními dvěma uzly v případě sudého počtu uzlů) a aby se jejich velikosti postupně zmenšovali čím více bychom se blížili ke středu. Krajní body největších takových intervalů tedy jsou krajní uzly. Takto pokračujeme, dokud nám nezbyde prostřední interval, který obsahuje pouze 2 až $4k$ uzlů.

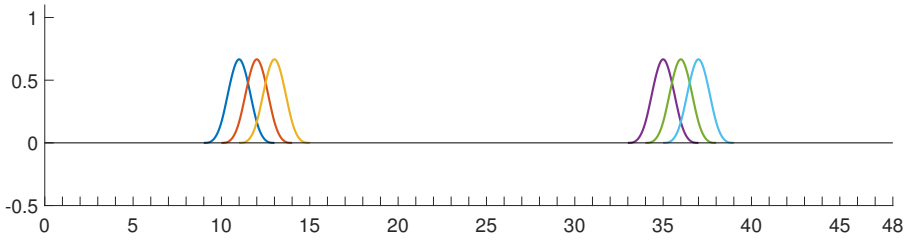
2. Na jednotlivých intervalech získaných v předchozím kroku zkonstruujeme dyadické splinety pomocí postupu uvedeného v podkapitole 3.2.1.

3. Mezi každými dvěma dyadickými splinety pak ještě zbývají takzvané nepravidelné B-splajny, jejichž nosiče jsou součástí dvou vedlejších splinetů. Tyto B-splajny budeme chtít zortogonalizovat vůči těmto splinetů, vůči předchozím nepravidelným OB-splajnům a symetricky vůči samy sobě, což provedeme následovně:

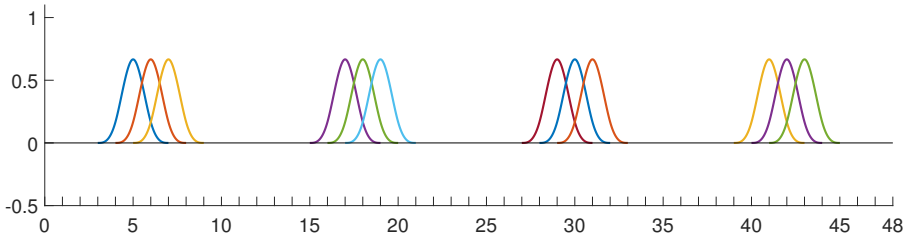
   (a) Chceme nepravidelné B-splajny zortogonalizovat vůči vedlejším splinetům. Je nutné však podotknout, že je opět stačí zortogonalizovat



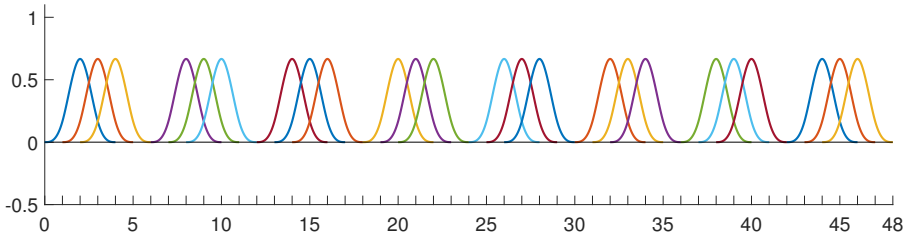

B-splajny na úrovni 3




B-splajny na úrovni 2

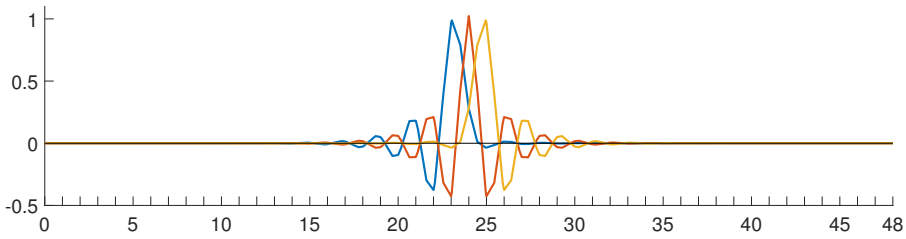



B-splajny na úrovni 1

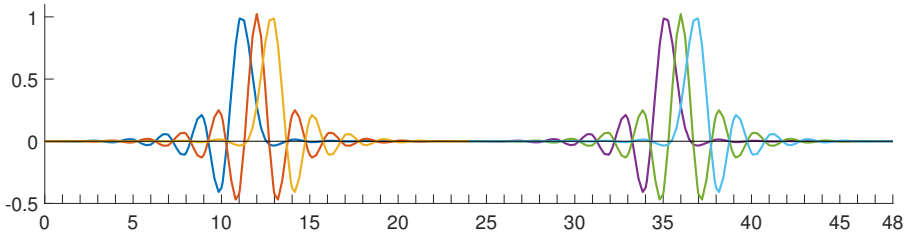



B-splajny na úrovni 0

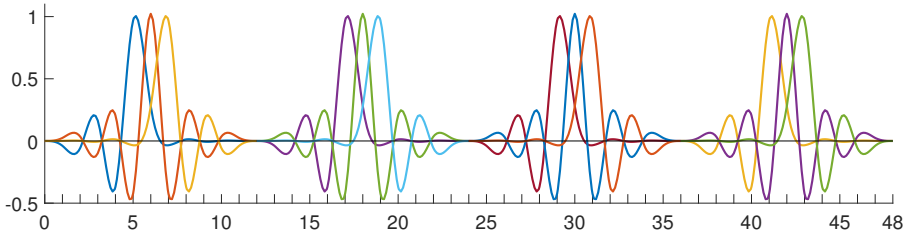
Obrázek 3.5: Rozdělení B-splajnů stupně 3 - dyadický případ



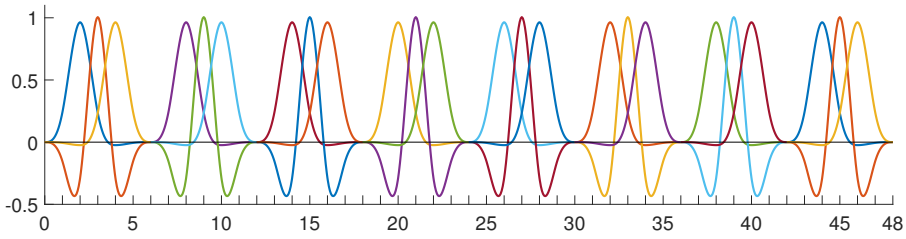

OB-splajny na úrovni 3




OB-splajny na úrovni 2




OB-splajny na úrovni 1




OB-splajny na úrovni 0

Obrázek 3.6: Zortogonalizované B-splajny stupně 3 - dyadický případ

vůči těm OB-splajnům ve splinetech, které mají neprázdné průniky nosičů s právě ortogonalizovanými nepravidelnými B-splajny.

(b) Chceme nepravidelné B-splajny zortogonalizovat vůči všem předchozím nepravidelným OB-splajnům. Opět je však stačí zortogonalizovat vůči těm nepravidelným OB-splajnům, jejichž průniky nosičů s nosiči právě ortogonalizovaných B-splajnů jsou neprázdné. Stačí je tedy zortogonalizovat vůči $k$ posledně zortogonalizovaným nepravidelným OB-splajnům vlevo či vpravo.

(c) Chceme nepravidelné B-splajny zortogonalizovat vůči samy sobě. Toho dosáhneme pomocí symetrické ortogonalizace.

4. Po provedení prvních tří kroků nám ještě zbývá zortogonalizovat $k$ až $5k-1$ prostředních B-splajnů. $k$ prostředních B-splajnů nejvíce vlevo zortogonalizujeme vůči poslednímu splinetu vlevo a vůči $k$ posledně zortogonalizovaným nepravidelným OB-splajnům vlevo. Obdobně to proběhne i s $k$ B-splajny nejvíce vpravo, tj. zortogonalizujeme je vůči poslednímu splinetu vpravo a vůči $k$ posledně zortogonalizovaným nepravidelným OB-splajnům vpravo. V závislosti na tom, kolik nám zbývalo zortogonalizovat B-splajnů na začátku kroku, provedeme jeden z následujících kroků:

   (a) Pokud zbývalo $k$ až $2k$ prostředních B-splajnů, tak je ještě symetricky zortogonalizujeme vůči samy sobě.

   (b) Pokud zbývalo $2k+1$ až $3k-1$ prostředních B-splajnů, tak právě zortogonalizované B-splajny ještě symetricky zortogonalizujeme vůči samy sobě. Zbývající ještě nezortogonalizované B-splajny zortogonalizujeme vůči těm právě zortogonalizovaným a posléze i symetricky vůči samy sobě.

   (c) Pokud zbývalo $3k$ až $5k-1$ B-splajnů, tak ještě nezortogonalizované B-splajny zortogonalizujeme vůči těm právě zortogonalizovaným a posléze i symetricky vůči samy sobě.

**Poznámka 3.3.** Jestliže má původní síť uzlů pouze $2k+1$ až $4k$ uzlů, tak při ortogonalizaci můžeme rovnou přeskočit na krok 4a či 4b.

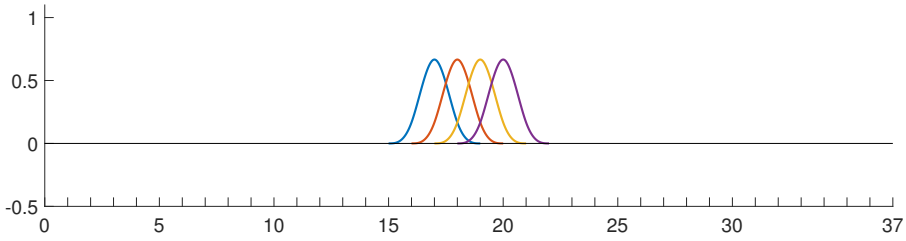
**Kód 3.4.** Pro ortogonalizaci B-splajnů obecného stupně $k$ na nedyadickém intervalu pomocí splinetů jsem v MATLABu vytvořil kód `splinet_k_general.m`. Kód je k nalezení v příloze diplomové práce.

**Příklad 3.4.** Nechť máme zadanou ekvidistantní nedyadickou síť uzlů, jež má $2^5+4$ vnitřních uzlů. Naším úkolem je na této síti vytvořit ortogonální B-splajny stupně 3.

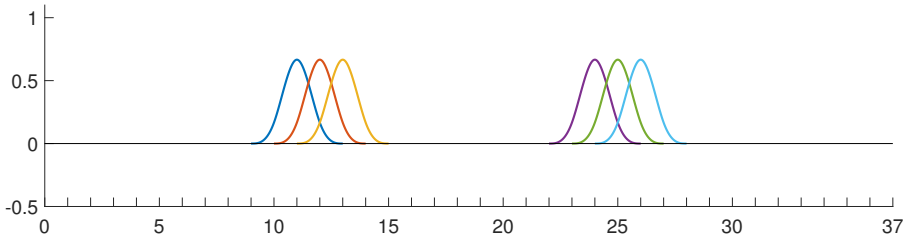
K řešení využijeme MATLAB a vytvořený kód `splinet_k_general.m`. Uzly zapíšeme jako vektor do proměnné `knots` a funkci zavoláme pomocí `OB=splinet_k_general(knots,3)`. Do OB se nám pak ve formě struct zapíší výsledné OB-splajny.

Původní B-splajny rozdělené do úrovní lze vidět na obrázku 3.7. Výsledné ortogonální OB-splajny jsou pak na obrázku 3.8.

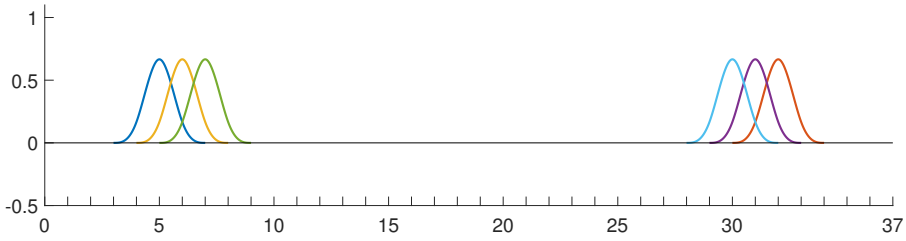
Příklad vypracovaný s pomocí MATLABu lze nalézt v příloze pod názvem `priklad_nd_k.m`.



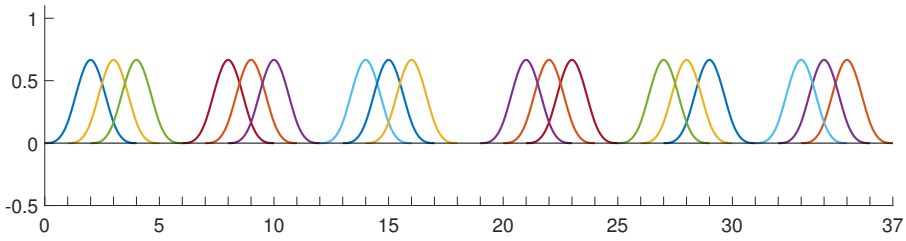

Prostřední B-splajny




Nepravidelné B-splajny mezi splinety

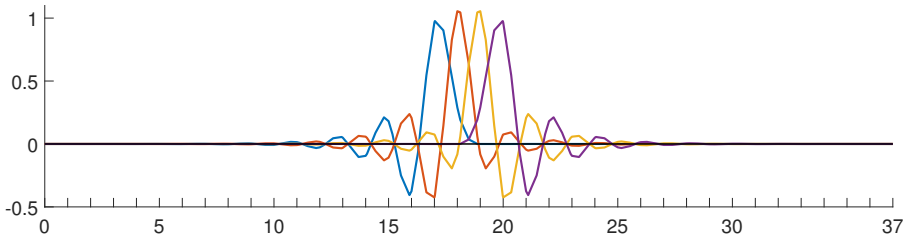



B-splajny na úrovni 1

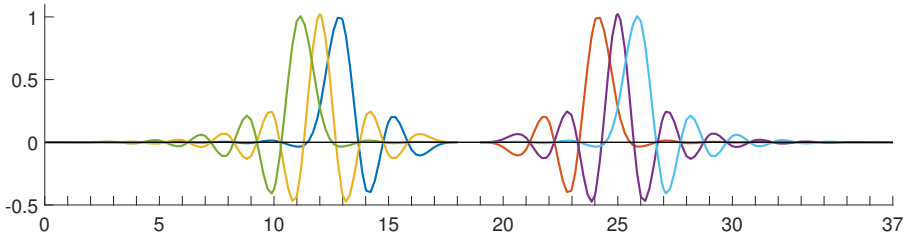



B-splajny na úrovni 0

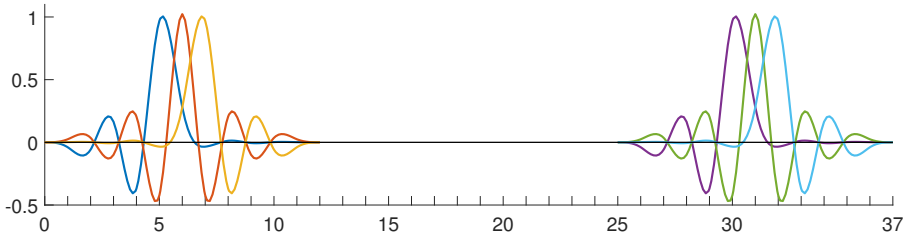
Obrázek 3.7: B-splajny stupně 3 - obecný případ



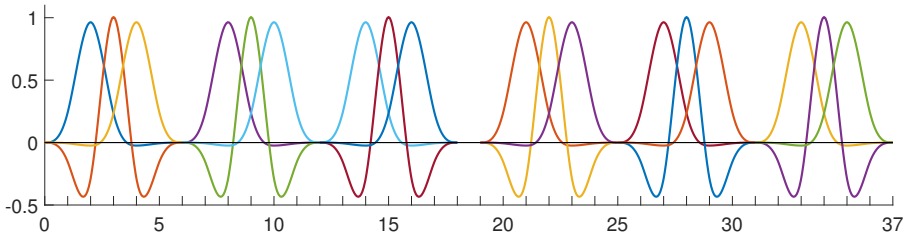

Prostřední OB-splajny




Nepravidelné OB-splajny mezi splinety




OB-splajny na úrovni 1




OB-splajny na úrovni 0

Obrázek 3.8: Zortogonalizované B-splajny stupně 3 - obecný případ

# 4. Praktická část

V kapitole 1 jsme si ukázali, že jsou B-splajny užitečné při hledání polynomických splajnů, které by interpolovaly či aproximovaly zadaná data. V dané kapitole jsme však po B-splajnech vzájemnou ortogonalitu nepožadovali. Je přirozené se tedy zeptat, zdali by se ortogonální B-splajny také daly využít k aproximaci dat a jestli by nám práce s ortogonálními B-splajny nezjednodušila výpočet.

Logicky by nám ortogonální B-splajny měly pomoci v problémech, kde můžeme jejich ortogonality využít. Ortogonalita se nám tedy může hodit například v případech, kdy pracujeme se spojitými skalárními součiny. Matice takových skalárních součinů B-splajnů by pak byla diagonální jednotková, čímž by se nám měly zjednodušit výpočty v soustavách, kde ji využíváme.

Jako data k ilustraci užití ortogonálních B-splajnů jsem si zvolil volně dostupná data z [8]. Jedná se o denní přehled osob nakažených onemocněním COVID-19. Konkrétně jsem si vybral denní počty nově nakažených osob za rok 2021. Celkem se tedy jedná o 365 hodnot, které jsou reprezentovány dvojicemi $(x_i, y_i)$, $i = 1, \ldots, 365$. Vektor $\boldsymbol{x}$ postupně nabývá hodnot od 1 do 365, čímž reprezentuje jednotlivé dny roku. Pod číslem 1 si tedy představíme den 1. 1. 2021, pod číslem 2 den 2. 1. 2021, atd. Pod vektorem $\boldsymbol{y}$ se pak nachází počty nově nakažených osob v odpovídající dny. Užitá data jsou součástí přílohy pod názvem `covid_prehled.csv`.

## 4.1. Interpolace

Nejprve se podíváme na interpolaci. Data budeme chtít proložit splajnem stupně 3. Aby byl interpolační splajn jediný, tak kolokační matice musí být regulární, čehož dle věty 1.5 docílíme tak, když báze prostoru bude složena právě z 365 B-splajnů a v každém B-splajnu se bude nacházet alespoň jeden bod $x_i$. Síť uzlů, která tomuto požadavku vyhovuje, je například síť skládající se právě ze všech bodů vektoru $\boldsymbol{x}$. Abychom dostali celou bázi prostoru, tak síť bude ještě nutné rozšířit o $2k$ uzlů navíc. Jako pomocné uzly zvolíme například

$\{-3, -2, -1, 366, 367, 368\}$. Záměrně jsme nevolili násobné uzly, protože ortogonalizovat B-splajny pomocí splinetů lze pouze na síti uzlů s jednoduchými uzly.

Pomocí kódu `splinet_k_general.m` na rozšířené síti vytvoříme ortogonální bázi B-splajnů, sestrojíme kolokační matici a z poznatků kapitoly 1.2.2 vypočítáme vektor $\boldsymbol{b}$, díky němuž bude interpolační splajn určen jednoznačně. Výsledný splajn vypočítáme a znázorníme. Výsledek lze vidět na obrázku 4.1. Jelikož se mimo interval $[1, 365]$ žádná další data nenachází, tak je splajn vykreslen pouze na tomto intervalu. Jak splajn vypadá na celé rozšířené síti uzlů už pro nás není relevantní.

Jak lze z obrázku vypozorovat, pro vyšší počet dat se splajnová funkce chová poněkud chaoticky a volba jiného stupně splajnu $k$ by nám v tomto ohledu nepomohla. Je proto vhodnější využít jednu z metod aproximace s menším počtem uzlů.

Nutno podotknout, že se vlastnost ortogonality B-splajnů ve výpočtu nevyužila, nicméně jak jsme si právě ověřili, interpolační splajn se dá zjistit i s pomocí ortogonálních B-splajnů.

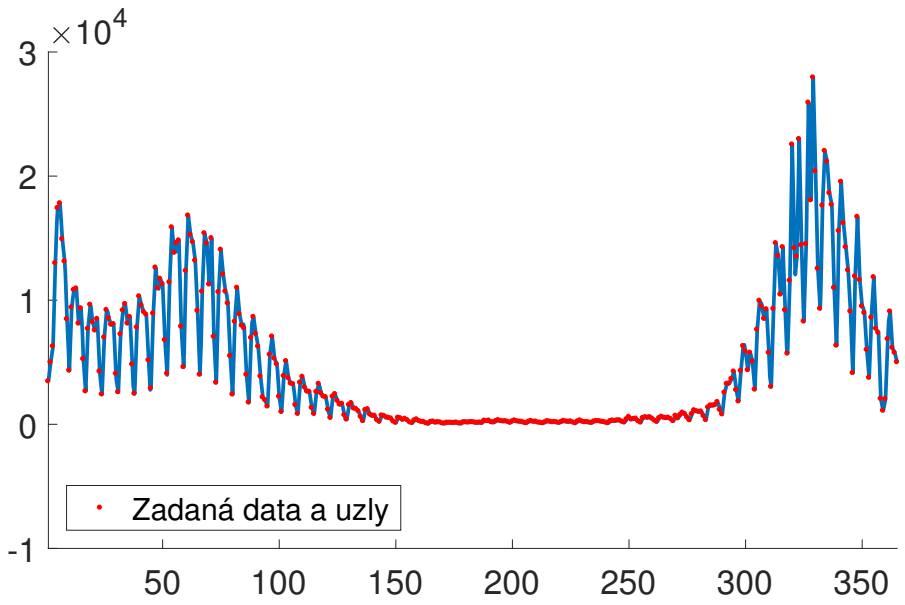
**Poznámka 4.1.** Splajn z obrázků 4.1 byl vykreslen pomocí MATLABu a jeho výpočet se nachází v příloze pod názvem `priklad_inter.m`.

## 4.2. Metoda nejmenších čtverců

Nyní se pokusíme data aproximovat ve smyslu nejmenších čtverců.

Jak již víme z podkapitoly 1.2.3, splajn, pro nějž je hodnota výrazu $\delta(\boldsymbol{b})$ minimální, je určen jednoznačně právě tehdy, když je kolokační matice plné sloupcové hodnosti. Dle věty 1.5 bude tedy vhodné zvolit takovou síť uzlů, ve které se mezi každými dvěma sousedními uzly bude nacházet alespoň jeden bod z vektoru $\boldsymbol{x}$.

Nejprve si pomocí kódu `splinet_k_general.m` vytvoříme bázové ortogonální B-splajny a zkonstruujeme kolokační matici. Jakmile máme kolokační matici plné sloupcové hodnosti, můžeme najít jednoznačné řešení normální soustavy rovnic:




Obrázek 4.1: Interpolace dat

$$\underbrace{\boldsymbol{C}_{k+1}^{T}(\boldsymbol{x})\boldsymbol{W}\boldsymbol{C}_{k+1}(\boldsymbol{x})}_{\boldsymbol{B}}\,\boldsymbol{b} = \underbrace{\boldsymbol{C}_{k+1}^{T}(\boldsymbol{x})\boldsymbol{W}\boldsymbol{y}}_{\boldsymbol{v}}$$

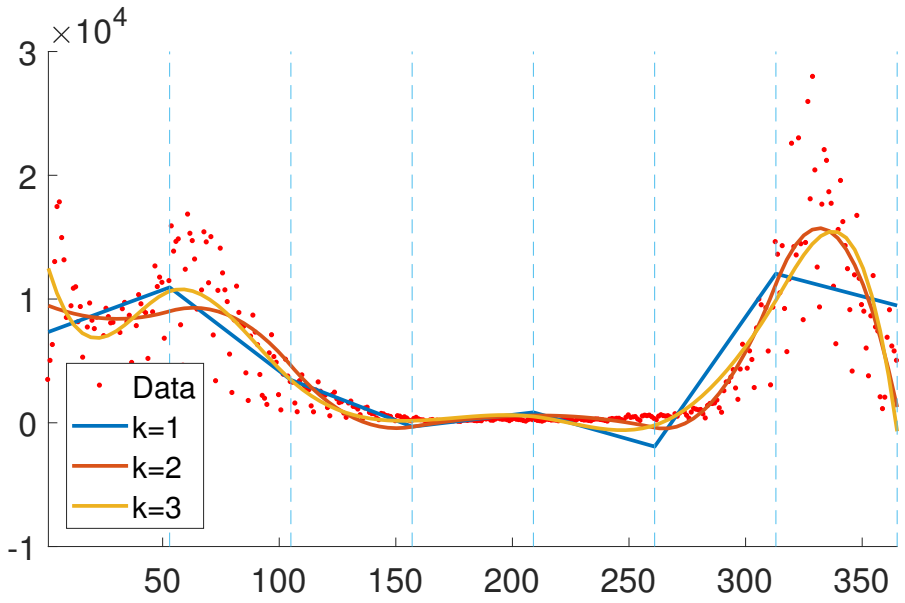
$$\boldsymbol{B}\boldsymbol{b} = \boldsymbol{v}$$

jako

$$\boldsymbol{b} = \boldsymbol{B}^{-1}\boldsymbol{v}$$

Dle poznámky 1.8 lze matici $\boldsymbol{B}$ a vektor $\boldsymbol{v}$ zapsat pomocí skalárních součinů. Jestliže bychom měli zadaná data spojitá a namísto váhových koeficientů bychom měli spojitou váhovou funkci, tak by nám díky ortogonalitě B-splajnů matice skalárních součinů $\boldsymbol{B}$ vycházela jako jednotková diagonální, čímž by se nám zjednodušil výpočet. U konkrétních diskrétních dat, které využíváme, se však ortogonalita neuplatní.

Na rozdíl od interpolace již má větší smysl se podívat, jaký má vliv volba sítě uzlů či volba stupně výsledného aproximačního splajnu. Navíc ještě musíme navolit váhové koeficienty, což může také ovlivnit výsledek. V následujících případech




Obrázek 4.2: Metoda nejmenších čtverců pro 8 uzlů

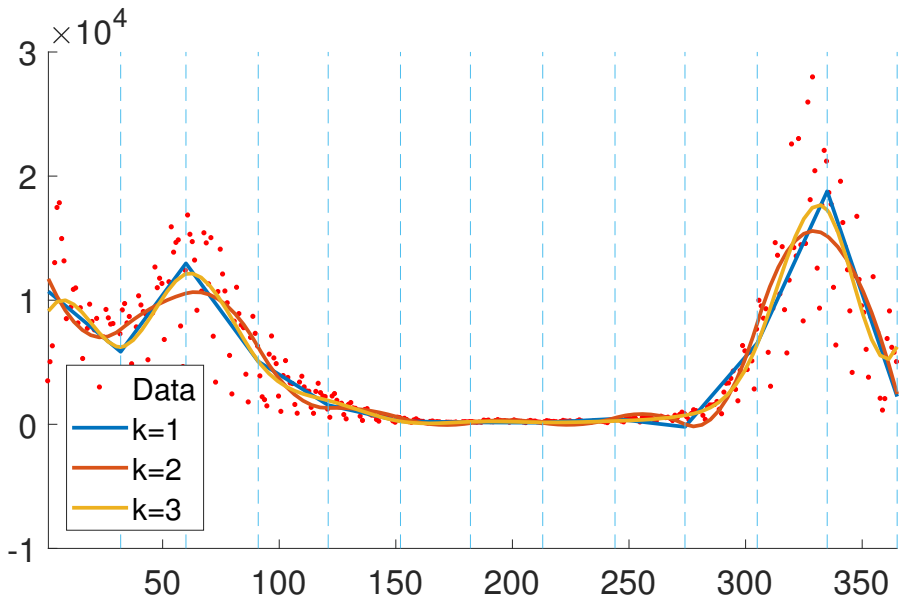
je však jako vektor váhových koeficientů volen jednotkový vektor.

V následujících několika bodech se pokusíme sestrojit splajny na síti uzlů s 8, 13, 30 a 50 uzly.

- Nejprve se podíváme na aproximaci pomocí metody nejmenších čtverců na ekvidistantní síti 8 uzlů. Jako stupně splajnů si postupně navolíme $1, 2$ a 3 a síť uzlů rozšíříme o $2k$ uzlů, přičemž jako pomocné uzly opět volíme pouze jednoduché uzly, abychom B-splajny mohli zortogonalizovat pomocí splinetů. Výsledné splajny vykreslíme do jednoho grafu, což lze vidět na obrázku 4.2. Umístění uzlů je tentokrát znázorněno svislými čárkovanými přímkami. Datové body byly zobrazeny až za dané funkce, aby byly funkce co nejlépe vidět.

  Již pro síť s pouze 8 uzly jsme v případě splajnů stupně 2 a 3 dostali poměrně dobrou aproximaci dat. Přesto se podíváme, jestli se nám situace nezlepší na síti s více uzly.

- Vzhledem k tomu, jaké máme data, tak není špatný nápad si zvolit takovou síť uzlů, kde sousední uzly tvoří intervaly, které odpovídají právě jednomu

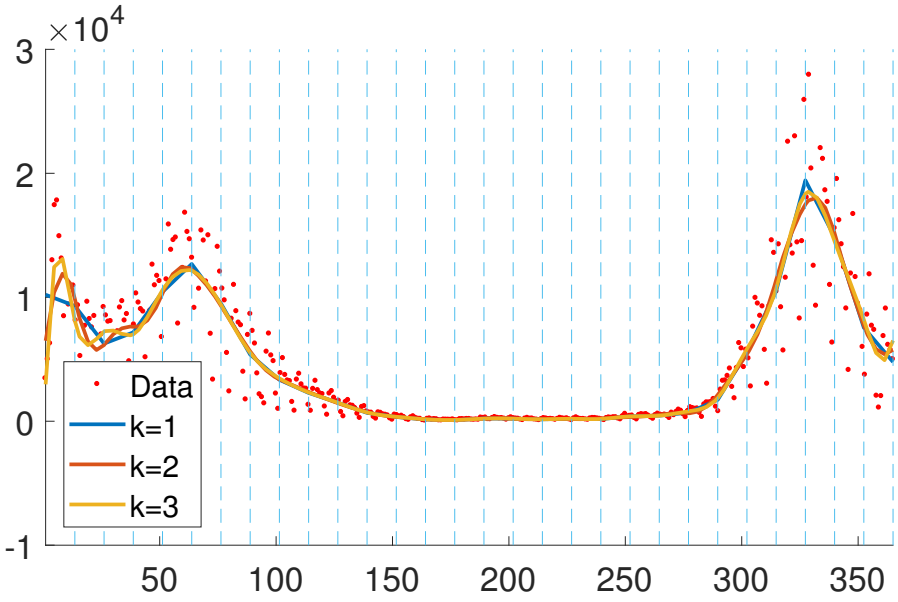



Obrázek 4.3: MNČ pro 13 uzlů

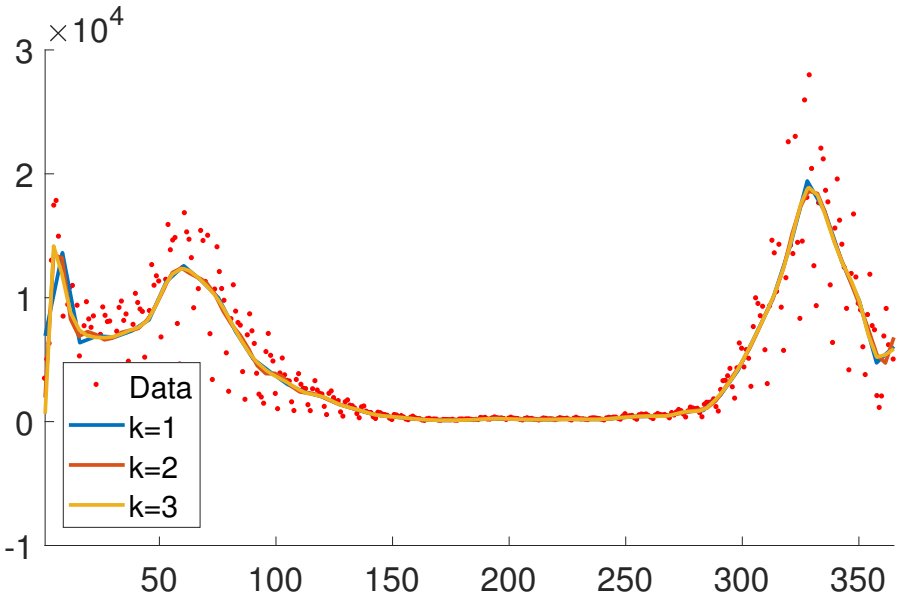
kalendářnímu měsíci v datech. Jedná se tedy o neekvidistantní síť uzlů s 13 uzly, přičemž prvním dvanácti uzlům se přiřadí body z dat odpovídající prvním dnům v měsících, zatímco poslednímu uzlu přiřadíme bod odpovídající poslednímu dnu roku. Opět jako stupně splajnů postupně navolíme $k = 1, 2, 3$ a vykreslíme do jednoho grafu. Graf je znázorněn na obrázku 4.3.

Jak lze z obrázku vypozorovat, nyní i splajn stupně 1 poměrně pěkně aproximuje daná data. Z takového grafu lze navíc zřetelně vyčíst, v kterých měsících byl počet nově nakažených osob nejvyšší či nejnižší.

- Zkusíme opět počet uzlů navýšit a jako síť zvolíme 30 ekvidistantních uzlů. Splajny stupňů $1, 2$ a $3$ nakreslíme do jednoho grafu, jenž je znázorněn na obrázku 4.4. Splajny jsou v takovém případě již hodně podobné a už není nutné hledat lepší aproximující splajny. Pro zajímavost jsou však na obrázku 4.5 vykresleny splajny na síti s 50 ekvidistantními uzly. Svislé přímky znázorňující polohu uzlů již není nutné vykreslovat, jelikož jsou uzly příliš blízko u sebe. Splajny na takové síti uzlů už téměř splývají.




Obrázek 4.4: MNČ pro 30 uzlů




Obrázek 4.5: MNČ pro 50 uzlů

**Poznámka 4.2.** Splajny z obrázků 4.2–4.5 byly vykresleny pomocí MATLABu a jejich výpočet se nachází v příloze pod názvem `priklad_mnc.m`.

## 4.3. Vyhlazování

Nyní se budeme snažit daná data aproximovat pomocí vyhlazování. Pro připomenutí hledáme vektor $\boldsymbol{b}$, jenž minimalizuje výraz

$$J_l^\alpha(\boldsymbol{b}) = (1-\alpha)\underbrace{\int_a^b \left(\frac{\partial^l}{\partial x^l}\sum_{i=-k}^{g} b_i B_i^{k+1}(x)\right)\left(\frac{\partial^l}{\partial x^l}\sum_{j=-k}^{g} b_j B_j^{k+1}(x)\right)dx}_{I_1(\boldsymbol{b})} +$$

$$+\alpha\underbrace{\sum_{j=1}^{n} w_j\left(y_j - \sum_{i=-k}^{g} b_i B_i^{k+1}(x_j)\right)^2}_{I_2(\boldsymbol{b})}$$

$$J_l^\alpha(\boldsymbol{b}) = (1-\alpha)I_1(\boldsymbol{b}) + \alpha I_2(\boldsymbol{b})$$

V podkapitole 1.2.4 jsme si ukázali, že lze pomocí věty 1.4 výraz $I_1(\boldsymbol{b})$ upravit na součin jednodušších matic. Toho tentokrát nevyužijeme, protože bychom museli přepočítat všechny koeficienty. Navíc bychom museli zkonstruovat ortogonální B-splajn nižšího stupně, čímž by se nám jenom zvýšil čas výpočtu. Derivaci B-splajnů ve výrazu $I_1(\boldsymbol{b})$ budeme namísto toho počítat rovnou v MATLABu pomocí funkce `fnder`.

Definujeme čtvercovou matici $\boldsymbol{H}_{kl} := \left(h_{ij}\right)_{i,j=-k,\ldots,g}$ typu $(g+k+1)\times(g+k+1)$, kde

$$h_{ij} = \int_a^b \frac{\partial^l}{\partial x^l} B_i^{k+1}(x)\frac{\partial^l}{\partial x^l} B_j^{k+1}(x)dx,$$

což vlastně bude matice skalárních součinů $l$-tých derivací B-splajnů. K výpočtu matice $\boldsymbol{H}_{kl}$ můžeme tudíž použít kód `skalarmatice.m`.

Výraz $I_1(\boldsymbol{b})$ pak bude vypadat jako

$$I_1(\boldsymbol{b}) = \boldsymbol{b}^T\boldsymbol{H}_{kl}\boldsymbol{b}$$

Celkem tedy můžeme $J_l^\alpha(\boldsymbol{b})$ zapsat jako

$$J_l^\alpha(\boldsymbol{b}) = (1-\alpha)\boldsymbol{b}^T\boldsymbol{H}_{kl}\boldsymbol{b} + \alpha\Big(\boldsymbol{y} - \boldsymbol{C}_{k+1}(\boldsymbol{x})\boldsymbol{b}\Big)^T\boldsymbol{W}\Big(\boldsymbol{y} - \boldsymbol{C}_{k+1}(\boldsymbol{x})\boldsymbol{b}\Big)$$

Nyní budeme postupovat obdobně jako v podkapitole 1.2.4. Vypočítáme si hessián funkce $J_l^\alpha(\boldsymbol{b})$ a zjistíme, kdy je pozitivně definitní. Jelikož je matice $\boldsymbol{H}_{kl}$ pozitivně semidefinitní, tak pozitivní definitnost hessiánu bude opět záviset na plné sloupcové hodnosti kolokační matice.

Pokud máme kolokační matici plné sloupcové hodnosti, tak existuje jediné minimum funkce $J_l^\alpha(\boldsymbol{b})$, které můžeme hledat jako řešení soustavy

$$\underbrace{\Big((1-\alpha)\boldsymbol{H}_{kl} + \alpha\boldsymbol{C}_{k+1}^T(\boldsymbol{x})\boldsymbol{W}\boldsymbol{C}_{k+1}(\boldsymbol{x})\Big)}_{\boldsymbol{A}}\boldsymbol{b} = \underbrace{\alpha\boldsymbol{C}_{k+1}^T(\boldsymbol{x})\boldsymbol{W}\boldsymbol{y}}_{\boldsymbol{u}}$$

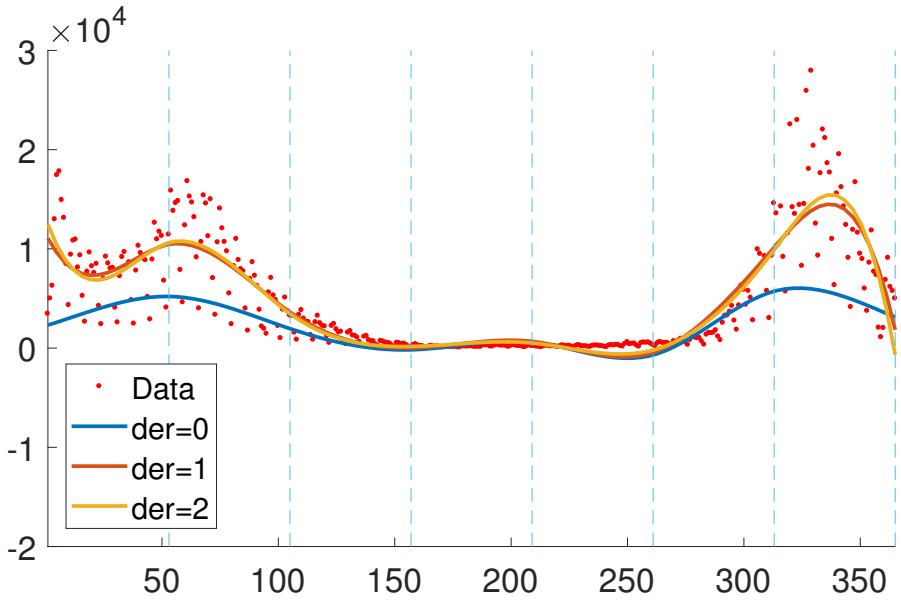
$$\boldsymbol{A}\boldsymbol{b} = \boldsymbol{u}$$

Jestliže bude matice $\boldsymbol{A}$ regulární, pak bude existovat právě jedno řešení soustavy. Pozitivní definitnost a tím i regularita matice $\boldsymbol{A}$ plyne z plné sloupcové hodnosti kolokační matice $\boldsymbol{C}_{k+1}(\boldsymbol{x})$.

Co se týče konstrukce splajnu, nejprve pomocí kódu `splinet_k_general.m` vytvoříme bázové ortogonální B-splajny a zkonstruujeme kolokační matici a matici $\boldsymbol{H}_{kl}$.

V následujících několika bodech se pokusíme sestrojit splajny na síti uzlů s 8, 13 a 30 uzly. Jako vektor váhových koeficientů bude volen jednotkový vektor.

- Stejně jako u metody nejmenších čtverců se nejprve podíváme na aproximaci vyhlazováním na ekvidistantní síti 8 uzlů. Budeme aproximovat splajnem stupně $k = 3$ a jako derivaci postupně navolíme $l = 0, 1, 2$ a $\alpha = 0.5$. Síť uzlů rozšíříme o $2k$ uzlů, přičemž jako pomocné uzly opět volíme pouze jednoduché uzly, abychom B-splajny mohli zortogonalizovat pomocí splinetů. Výsledné splajny vykreslíme do jednoho grafu, což lze vidět na obrázku 4.6. Řád derivace v legendě grafu jsem pojmenoval jako `der`, protože $l$ v MATLABu vypadalo spíše jako $I$.



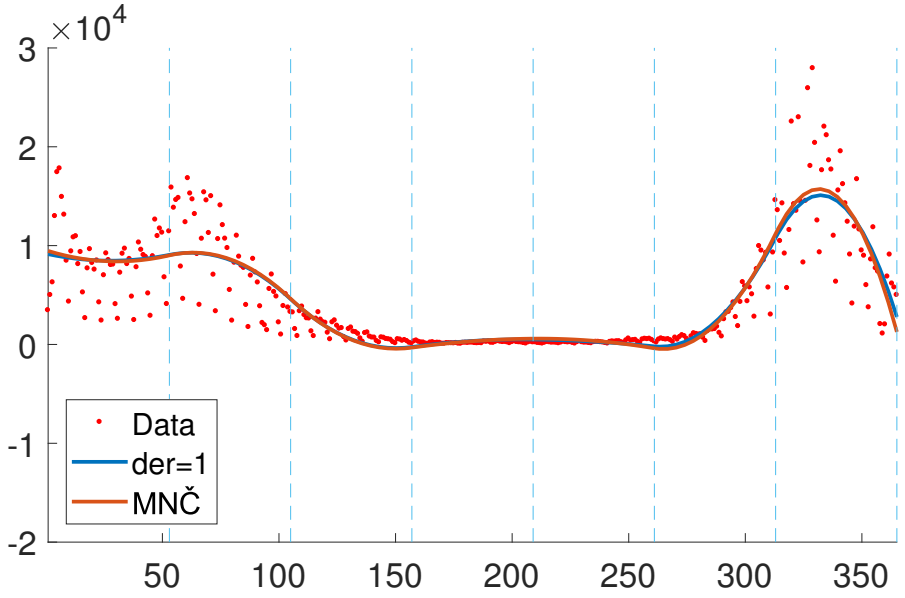

Obrázek 4.6: Vyhlazování pro 8 uzlů, kde $k = 3$

Jak lze z obrázku vypozorovat, splajn vyhlazování, pro nějž je $l = 0$, je pro volbu aproximace nevhodný. Nutno podotknout, že se ve výpočtu takového splajnu přímo využije ortogonalita B-splajnů.

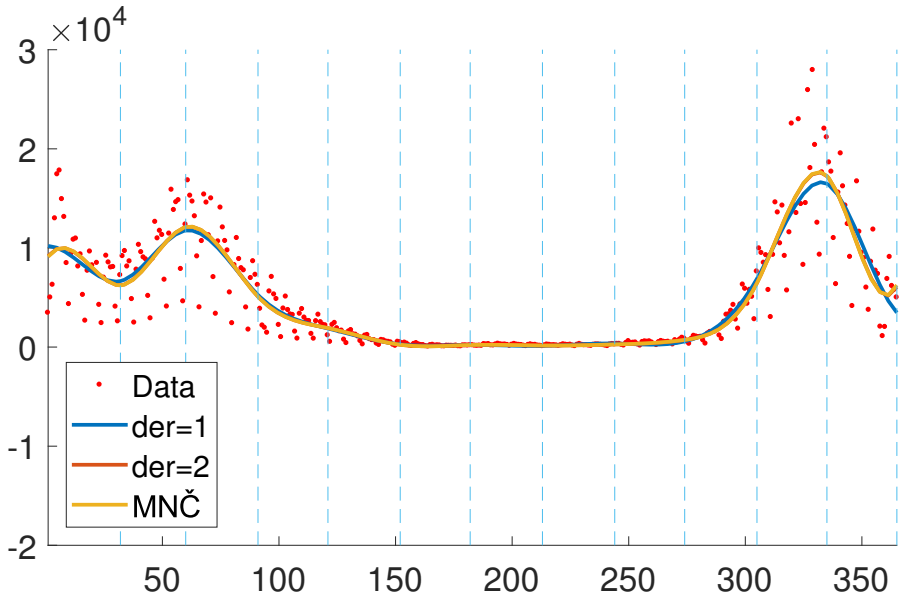
- Na stejné síti uzlů zkusíme data aproximovat pomocí vyhlazujícího splajnu, jenž bude mít stupeň $k = 2$ a jako řád derivace bude zvolena $l = 1$. Navíc jako $\alpha$ zvolíme 0.1, tudíž by se výsledný splajn měl blížit dostatečně hladké aproximaci. Srovnání vyhlazovacího splajnu a splajnu vytvořeného pomocí MNČ lze vidět na obrázku 4.7.

  I přes volbu poměrně malého $\alpha$ je aproximace pomocí metody nejmenších čtverců a vyhlazování téměř totožné. Je to pravděpodobně dáno volbou poměrně pěkných dat.

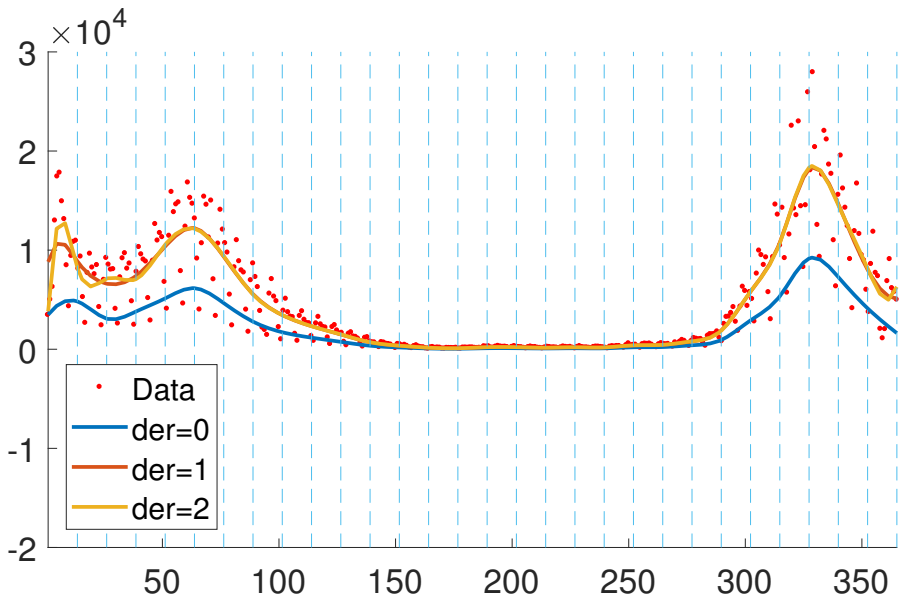
- Na neekvidistantní síti 13 uzlů, jejíž intervaly mezi sousedními uzly odpovídají kalendářním měsícům v roce, budeme chtít aproximovat daná data pomocí vyhlazujících splajnů stupně $k = 3$, přičemž postupně navolíme $l = 1, 2$ a $\alpha = 0.1$. Splajny navíc srovnáme s aproximačním splajnem sestrojeným pomocí metody nejmenších čtverců. Graf je znázorněn na obrázku




Obrázek 4.7: Srovnání MNČ a vyhlazování pro 8 uzlů, kde $k = 2$




Obrázek 4.8: Vyhlazování pro 13 uzlů, kde $k = 3$




Obrázek 4.9: Vyhlazování pro 30 uzlů, kde $k = 3$

4.8.

Jak lze z obrázku vypozorovat, splajny jsou téměř identické, přičemž splajn aproximovaný ve smyslu MNČ dokonce jeden ze splajnů překrývá.

- Jako síť ještě zvolíme 30 ekvidistantních uzlů. Splajny budeme konstruovat jako vyhlazující splajny stupně $k = 3$, kde postupně využijeme řády derivací $l = 0, 1, 2$. Výsledek nakreslíme do jednoho grafu, jenž je znázorněn na obrázku 4.9. Vyhlazovací splajny se pro $l = 1$ a $l = 2$ liší pouze na počátečních intervalech.

**Poznámka 4.3.** Splajny z obrázků 4.6–4.9 byly vykresleny pomocí MATLABu a jejich výpočet se nachází v příloze pod názvem `priklad_vyhl.m`. Jedná se o upravený kód `vyhl.m`, jenž byl poskytnut paní docentkou Machalovou a který lze také nalézt v příloze. U obrázků, kde jsme vyhlazování srovnávali s metodou nejmenších čtverců, jsme ještě navíc využili `priklad_mnc.m`

## 4.4. Shrnutí

Snažili jsme se proložit data [8] splajnem pomocí interpolace, metody nejmenších čtverců a vyhlazování, přičemž jsme jako bázové funkce používali ortogonální B-splajny. Jak jsme se přesvědčili, interpolace se k proložení většího množství dat nehodila. Aproximací pomocí metody nejmenších čtverců a pomocí vyhlazování jsme docházeli k dobrým, ale podobným výsledkům. K dostatečně dobré aproximaci daných dat by nám však pravděpodobně stačila aproximace pomocí metody nejmenších čtverců na síti o 13 uzlech. Obecně je však vhodnější použít vyhlazování, jelikož se metoda snaží nejen aproximovat daná data, ale i eliminovat náhlé změny funkčních hodnot aproximujícího splajnu.

K aproximaci jsme používali ortogonální B-splajny, nicméně vzhledem k tomu, že jsme měli diskrétní data, tak jsme vlastnost ortogonality často nevyužili. Jak již bylo zmíněno, v případě spojitých dat by tomu bylo jinak. Alespoň jsme si však ukázali, že metody fungují i pro ortogonální B-splajny.

# Závěr

Když mě paní docentka Machalová položila dotaz, zdali bych neměl zájem vypracovat diplomovou práci na téma ortogonálních B-splajnů, tak jsem ještě neměl tušení, co to vůbec B-splajn je. Naštěstí se během mého studia otevřel předmět, jež se splajny a B-splajny zabýval, takže jsem základní teorii mohl vstřebat na přednáškách, což mi výrazně usnadnilo psaní. Tyto základní poznatky jsem se snažil shrnout v první přípravné kapitole. Velkou oporou k psaní první kapitoly mi také byla skripta [3], z nichž jsem čerpal většinu základních definic. Tato skripta mi navíc byla sympatická tím, že byla jediná skripta o splajnech, které jsem našel psaná česky.

Cílem této diplomové práce bylo čtenáře seznámit s ortogonálními B-splajny a říct si o jejich využití. Problematika ortogonálních B-splajnů už bylo poněkud specifičtější téma, jež bylo zapotřebí si pořádně nastudovat sám. K tomu mi hlavně posloužil článek [4]. Představením ortogonálních B-splajnů a jejich metodami výpočtu jsem se zabýval ve druhé kapitole. Z metod se jednalo o Gramovu-Schmidtovu ortogonalizační metodu a její symetrickou alternativu. Kapitolu jsem navíc doplnil o příklady a kódy jednotlivých metod, jež byly vytvořeny v softwaru MATLAB.

V třetí kapitole se mluvilo o hlavním tématu již zmíněného článku a to o splinetech. V kapitole jsem zmínil, že se jednalo o poměrně nový způsob ortogonalizace, jenž byl určitým způsobem výhodnější, než předchozí metody. V závislosti na volbě sítě uzlů a stupně B-splajnů na dané síti byly v kapitole popsány celkem 4 alternativy výpočtů splinetů. Všechny tyto alternativy jsem navíc doplnil o příklady a kódy vytvořené v MATLABu.

Ve čtvrté a závěrečné kapitole jsem se podíval na využití ortogonálních B-splajnů v aproximaci dat. Konkrétně se jednalo o proložení dat pomocí interpolace, metody nejmenších čtverců a vyhlazování. K předvedení daných metod jsem využil MATLAB a volně dostupná data [8]. Jelikož se jednalo o poměrně pěkná data, tak nám k dobré aproximaci dat stačilo pouze menší počet uzlů v síti.

# Literatura


[1] De Boor, C.: *A practical guide to splines*, volume 27. Springer-Verlag New York, 1978.

[2] Dierckx, P.: *Curve and Surface Fitting with splines*. Oxford University Press, New York, 1993.

[3] Kobza, J.: *Splajny*. Vydavatelství Univerzity Palackého v Olomouci, 1993.

[4] Liu, X., Nassar, H., Podgórski, K.: *Splinets - efficient orthonormalization of the B-splines*. arXiv:1910.07341 [math.ST], 2019.

[5] Machalová, J.: *Optimal Interpolating And Optimal Smoothing Spline*. Journal of Electrical Engineering, Vol. 53, No. 12/s, 2002, 79-82.

[6] Machalová, J.: *Optimal Inerpolatory Splines Using B-spline Representation*. Acta Univ. Palacki. Olomuc., Fac. rer. nat., Mathematica **41** (2002), 105-118.

[7] Schumaker, L.: *Spline Functions: Basic Theory*. Cambridge University Press, 2007.

[8] Ústav zdravotnických informací a statistiky ČR: *COVID-19: Přehled osob s nově prokázanou nákazou* [online]. Ministerstvo zdravotnictví České republiky, 2021. Datum aktualizace: 12. 1. 2022, 4:54 [cit. 12. 1. 2022]. Dostupné z: https://opendata.mzcr.cz/dataset/covid-19-prehled-osob-s-nove-prokazanou-nakazou.

# Seznam příloh

## Kódy

- `vyhl.m` – vyhlazování dat (kód byl vytvořen paní docentkou Machalovou)
- `matice_fcionalu.m` – matice skalárních součinů potřebná k vyhlazování (kód byl vytvořen paní docentkou Machalovou)
- `matder.m` – vytváření pomocných matic k vyhlazování (kód byl vytvořen paní docentkou Machalovou)
- `gramsch.m` – algoritmus Gramovy-Schmidtovy metody
- `gsorto.m` – přesnější Gramova-Schmidtova metoda (viz poznámku 2.2)
- `skalar.m` – spojitý skalární součin mezi dvěma B-splajny
- `skalarmatice.m` – vypsání skalární matice B-splajnů
- `symlemma.m` – symetrická ortonormalizace dvou normalizovaných B-splajnů
- `gssymcor.m` – symetrická Gramova-Schmidtova metoda
- `splinet1.m` – tvoření splinetů stupně 1, dyadický případ
- `splinet2.m` – tvoření splinetů stupně 1, obecný případ
- `splinet_k_dyadic.m` – tvoření splinetů obecného stupně $k$, dyadický případ
- `splinet_k_general.m` – tvoření splinetů stupně $k$, obecný případ

## Příklady vypočítané v MATLABu

- `priklad_gs.m` – vypracované příklady 2.1 a 2.2
- `priklad_daydic_1.m` – vypracovaný příklad 3.1
- `priklad_nd_1.m` – vypracovaný příklad 3.2
- `priklad_dyadic_k.m` – vypracovaný příklad 3.3
- `priklad_nd_k.m` – vypracovaný příklad 3.4
- `priklad_inter.m` – interpolace dat `covid_prehled.csv`
- `priklad_mnc.m` – aproximace dat `covid_prehled.csv` pomocí metody nejmenších čtverců

- `priklad_vyhl.m` – aproximace dat `covid_prehled.csv` pomocí vyhlazování (jedná se o upravený kód `vyhl.m`, jenž byl poskytnut paní docentkou Machalovou)

## Data

- `covid_prehled.csv` – data o počtu nakažených osob onemocněním COVID-19 (převzato z [8])