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

DIPLOMOVÁ PRÁCE

Vyhlazující splajny v R

Vedoucí diplomové práce: **RNDr. Jitka Machalová, Ph.D.**
Vypracovala: **Bc. Zuzana Bělašková**
Studijní program: N1103 Aplikovaná matematika
Studijní obor: Aplikace matematiky v ekonomii
Forma studia: Prezenční
Rok odevzdání: 2015

BIBLIOGRAFICKÁ IDENTIFIKACE

Autor: Zuzana Bělašková

Název práce: Vyhlazující splajny v R

Typ práce: Diplomová práce

Pracoviště: Katedra matematické analýzy a aplikací matematiky

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

Rok obhajoby práce: 2015

Abstrakt: Splajn představuje speciální funkci, která je po částech polynom nižšího stupně. Tyto splajny se používají k aproximaci dat a v dnešní době existuje více možností, jak se dají zkonstruovat. Tato práce je zaměřena na B-splajnovou reprezentaci, kde k sestavení splajnu používáme bázové funkce. Kromě nezbytné teorie si v této práci představíme také použití splajnů v rámci interpolace, metody nejmenších čtverců a vyhlazování. Všechny metody jsou doplněny o kódy vytvořené v softwaru R. Jak lze jednotlivé splajny zkonstruovat a jaké musí být splněny podmínky je obsahem této práce.

Klíčová slova: Bázový splajn, báze, síť uzlů, interpolace, metoda nejmenších čtverců, vyhlazování splajnů.

Počet stran: 86

Počet příloh: 1

Jazyk: český

BIBLIOGRAPHICAL IDENTIFICATION

Author: Zuzana Bělašková

Title: Smoothing spline in R

Type of thesis: Master's

Department:
Department of Mathematical Analysis and Application of Mathematics

Supervisor: RNDr. Jitka Machalová, Ph.D.

The year of presentation: 2015

Abstract: Splines are special piecewise polynomial functions, which we are using for approximation. Nowadays, there are more ways how can be splines created. This thesis is focused on B-spline representation, which created splines by using basis functions. We introduce the necessary theory and demonstrate the using of basis splines in three approximation methods - interpolation, least squares method and smoothing. All methods are supplemented by codes created in the software R. The thesis deals with question, how to create the smoothing spline and which conditions must be fulfilled.

Key words: Basis spline, sequence of knots, interpolation, least squares method, smoothing spline.

Number of pages: 86

Number of appendices: 1

Language: Czech

Prohlášení

Prohlašuji, že jsem diplomovou práci vypracovala samostatně pod vedením paní RNDr. Jitky Machalové, Ph.D. s použitím uvedené literatury.

V Olomouci dne ....

Podpis

Obsah

Poděkování

Ráda bych poděkovala vedoucí mé diplomové práce, paní RNDr. Jitce Machalové, Ph.D., za cenné rady, věcné připomínky a také za čas, který mi na konzultacích věnovala.

Úvod

Cílem mé diplomové práce je seznámit čtenáře s problematikou vyhlazujících splajnů a ukázat jejich použití v softwaru R. Pod pojmem „splajn“ si představíme speciální funkci, která je po částech polynomem. První zmínka o splajnech se datuje na počátek 20. století. Je ovšem nutné říci, že v této době byla podoba splajnů zcela jiná než dnes. Nejednalo se totiž o funkci, ale o jakási pružná pravítka, která procházela předem danými body a vytvářela tak hladké plochy. V této době byly splajny využívány zejména v lodním a leteckém průmyslu. Velký „boom“ zažily splajny až s příchodem prvních počítačů. Zpracovávat totiž velké množství čísel a rozsáhlé soustavy rovnic ručně bylo poněkud obtížné a počítače nám tuto práci velmi usnadnily. Pro práci se splajny se nabízí využít některý z matematických softwarů. Já ve své práci používám software R, ve kterém není problematika splajnů až tak rozšířená. Jeho nespornou výhodou je fakt, že se jedná o volně šířitelný software, je tedy dostupný opravdu pro každého.

Jelikož teorie splajnů je jen velmi okrajově zahrnuta v mém navazujícím magisterském studiu, byla jsem odkázána výhradně na literaturu. I přes to, že tato teorie je značně rozsáhlá, existuje jen velmi málo česky psaných publikací na toto téma. Jednou z nich jsou skripta „Splajny“ od J. Kobzy [2], ze kterých jsem při psaní mé diplomové práce čerpala zejména základní pojmy z teorie splajnů. Pro třetí a čtvrtou kapitolu mé práce jsem dále přibrala článek „Optimal interpolating and optimal smoothing spline“ od J. Machalové [3].

Jak již bylo naznačeno, cílem této práce je seznámit se s teorií splajnů, zejména s jejich B-splajnovou reprezentací a vytvořit vyhlazující splajny v softwaru R. První úvodní kapitola se bude věnovat základním pojmům a vlastnostem bázových splajnů. Zároveň zde bude vysvětlen rozdíl mezi pp-reprezentací a B-splajnovou reprezentací splajnů, včetně výhod, jenž B-splajnová reprezentace nabízí. Po nastudování základní teorie se v kapitolách dvě a tři podrobněji podíváme na interpolaci a metodu nejmenších čtverců. Kromě nezbytné teorie zde budou zařazeny také mnou vytvořené funkce v softwaru R, které čtenáři poskytnou pohled na to, jak tyto metody aproximace fungují v praxi. Následující čtvrtá kapitola

se věnuje hlavnímu cíli mé práce - vyhlazujícím splajnům. Čtenář v této kapitole zjistí, že se jedná o jakýsi kompromis mezi interpolací a metodou nejmenších čtverců. Nezbytná teorie bude opět doplněna o vytvořenou funkci v R, která poslouží jako ukázka pro použití vyhlazujících splajnů na konkrétních hodnotách.

Jak již bylo naznačeno, k výpočtům a vykreslování grafů splajnů budu používat matematický software R. Tento program obsahuje balíček příkazů `splines`, který je určený pro práci se splajny. Nicméně, jak ukazuje kapitola 5.1, jeho využití je dosti omezené. I z toho důvodu jsem se snažila všechny probírané metody naprogramovat sama a vytvořit tak univerzálnější nástroj pro práci se splajny. V balíčku `splines` také úplně chybí příkazy pro vykreslení grafu splajnu. Této problematice se bude věnovat poslední kapitola 5.2, která obsahuje mnou vytvořené funkce pro různé způsoby vykreslování požadovaných grafů splajnu.

1. Bázové splajny (B-splajny)

Jako první použil slovo „spline“ ve spojitosti s hladkou po částech polynomiální aproximací matematik I. J. Schoenberg, který je také považován za zakladatele teorie splajnů. Nicméně již dříve tento pojem využívali konstruktéři, např. aby docílili dostatečně hladkého tvaru trupu lodí a později také letadel. Pro získání požadované hladké křivky používali speciální pružné pravítko (anglicky „spline“), které bylo upevněno v určitých bodech a díky své pružnosti dosáhlo požadovaného tvaru.
V dnešní době existuje celá řada splajnů. Tato práce je zaměřena na *polynomické splajny*, konkrétně na jejich *B-splajnovou reprezentaci*. U polynomických splajnů existuje více možností, jak se dají zkonstruovat. Pokud je splajn tvořen polynomy, tak hovoříme o *po částech polynomické reprezentaci* (neboli pp-reprezentaci). Další způsob je ten, kdy k sestavení splajnu používáme bázové funkce, tzv. *B-splajny* a v této práci se jimi budeme podrobněji zabývat.

1.1. Základní pojmy teorie splajnů

Pro začátek je nutné si uvést některé základní pojmy. Následující přístup souvisí s pp-reprezentací, kde se k sestavení polynomického splajnu využívají polynomy.

Definice 1.1. Na síti uzlů $(\Delta x) : a = x_0 < x_1 < \ldots < x_n < x_{n+1} = b$ budeme polynomickým splajnem stupně m s defektem d (m a d jsou celá nezáporná čísla) rozumět funkci $S_{md}(x)$ s následujícími vlastnostmi:

1. $S_{md}(x)$ je na každém intervalu $\langle x_i, x_{i+1}\rangle, i = 0, 1, \ldots, n$ polynomem stupně nejvýše m.

2. $S_{md} \in C^{m-d}\langle a, b\rangle$, tj. v uzlech x_i má $S_{md}(x)$ spojité derivace do řádu $m-d$, číslo d se nazývá defektem splajnu.

Symbolem $S_{md}(\Delta x)$ budeme označovat lineární prostor splajnů $S_{md}(x)$ na síti uzlů (Δx).

Uvažujme splajn $S_{md}(x)$ z Definice 1.1. Víme tedy, jak vypadá síť uzlů, na které je definován, známe jeho stupeň m i defekt d. Dimenzi prostoru $S_{md}(\Delta x)$ určíme pomocí vztahu

$$\dim S_{md}(\Delta x) = (m+1)(n+1) - n(m-d+1) = m + nd + 1,$$

kde $(m+1)(n+1)$ představuje počet neznámých koeficientů polynomů, tj. máme $(m+1)$ koeficientů na $(n+1)$ intervalech, které určují splajn $S_{md}(x)$. Druhá část výrazu $n(m-d+1)$ vyjadřuje, že v n vnitřních uzlech splajnu má být splněno $(m-d+1)$ podmínek spojitosti, čímž dojde k eliminaci některých neznámých.

Poznámka 1.1. Uvažujme splajn $S_{31}(x)$, který je definovaný na síti uzlů $(\Delta x) = \{x_0, x_1, x_2\}$. Potom $a = x_0, b = x_2$ a x_1 je vnitřní uzel náležící intervalu (a, b). Splajn $S_{31}(x)$ je určený dvěma polynomy:

$$P_0(x) = a_0x^3 + b_0x^2 + c_0x + d_0 \quad \text{pro} \quad x \in \langle x_0, x_1 \rangle ,$$

$$P_1(x) = a_1x^3 + b_1x^2 + c_1x + d_1 \quad \text{pro} \quad x \in \langle x_1, x_2 \rangle ,$$

tj. stupeň splajnu je $m = 3$ a defekt d budeme uvažovat roven jedné. Pak má tento splajn, dle Definice 1.1, spojité derivace do řádu $m-d$ v uzlu x_1. V našem případě tedy do druhého řádu včetně, což obvykle značíme symbolem $S_{31}(x) \in C^2 \langle a, b \rangle$. Z toho vyplývá, že musí platit následující podmínky spojitosti:

$$P_0(x_1) = P_1(x_1),$$

$$P_0'(x_1) = P_1'(x_1),$$

$$P_0''(x_1) = P_1''(x_1).$$

Je vidět, že pro splajn $S_{31}(x)$ dostaneme 3 podmínky spojitosti, což odpovídá vztahu $m - d + 1 = 3 - 1 + 1 = 3$. Splajn $S_{31}(x)$ je určený dvěma polynomy,

které dohromady obsahují 8 neznámých koeficientů a_i, b_i, c_i, d_i, pro $i = 0, 1$. Jak již bylo řečeno, x_1 je jediný vnitřní bod uvažovaného intervalu (a, b). Dimenzi prostoru splajnů určíme tak, že od počtu neznámých koeficientů odečteme počet podmínek spojitosti vynásobený počtem jeho vnitřních uzlů. V našem případě máme tedy

$$\dim S_{31}(\Delta x) = 8 - 3 = 5.$$

1.2. Definice a základní vlastnosti B-splajnů

Od po částech polynomické reprezentace splajnů nyní přejdeme k novějšímu způsobu sestrojení polynomických splajnů, k tzv. B-splajnové reprezentaci. Nevýhoda pp-reprezentace spočívá v tom, že u splajnů vyššího stupně dostáváme při hledání neznámých koeficientů rozsáhlé soustavy lineárních rovnic. Například máme-li splajn, který je na intervalu $\langle x_i, x_{i+1}\rangle$ vyjádřený jako polynom třetího stupně, tj.

$$P_i(x) = a_i x^3 + b_i x^2 + c_i x + d_i,$$

je potřeba určit 4 neznámé koeficienty a_i, b_i, c_i a d_i pro $i = 0, \ldots, n$. S rostoucím stupněm splajnu počet těchto koeficientů narůstá a jejich výpočet se komplikuje. Díky tomu, že u B-splajnové reprezentace používáme k sestavení polynomického splajnu B-splajny, se těmto potížím vyhneme.

Definice 1.2. Nechť $(\Delta y) = \{y_i; i = 1, \ldots, N\}$ je neklesající posloupnost uzlů splajnu. Ke každému uzlu y_i, který má na síti v pravo ještě k sousedních uzlů, $y_i < y_{i+k}$, definujeme B-splajn k-tého řádu (stupně $k - 1$) vztahem

$$B_i^k(x) = (y_{i+k} - y_i)(t - x)_+^{k-1}[y_i, y_{i+1}, \ldots, y_{i+k}].$$

Každý B-splajn definovaný tímto způsobem a také každá jeho lineární kombinace je po částech polynomickou funkcí.

Poznámka 1.2. V definici B-splajnu je poměrná diference k-tého řádu funkce $(t - x)_+^{k-1}$ argumentu t v závislosti na parametru x. Tato funkce je definována

předpisem

$$(t-x)_+^{k-1} = \begin{cases} (t-x)^{k-1} & \text{pro} \quad t > x, \\ 0 & \text{pro} \quad t \leq x. \end{cases}$$

Nyní si uvedeme **základní vlastnosti bázových splajnů**, které si označíme **(v1)** až **(v3)**, jenž jsou převzaté z [2].

(v1) B-splajn $B_i^k(x)$ je na každém intervalu $\langle y_j, y_{j+1}\rangle$ pro $j = 1, \ldots, N-1$ polynomem stupně nejvýše $k-1$. Pro jednoduché uzly y_j, tj. pro uzly ve kterých je defekt $d = 1$, platí

$$B_i^k(x) \in C^{k-2}\langle y_1, y_{N-1}\rangle, \quad \text{tj.} \quad B_i^k(x) \in S_{k-1,1}(\Delta y).$$

Poznámka 1.3. Prostor $S_{k-1,1}(\Delta y)$ uváděný ve vlastnosti **(v1)** je totožný s prostorem $S_{md}(\Delta x)$, který jsme si zavedli v Definici 1.1. Uvažujeme tedy polynom stupně $m = k-1$ s defektem $d = 1$. Jediné co je rozdílné, je síť uzlů, na které je uvažovaný B-splajn definovaný.

(v2) Nosičem $B_i^k(x)$ je interval $\langle y_i, y_{i+k}\rangle$; tj. $B_i^k(x) = 0$ pro $x \notin \langle y_i, y_{i+k}\rangle$.

Poznámka 1.4. Důsledkem předešlé vlastnosti je fakt, že na intervalu $\langle y_j, y_{j+1}\rangle$ jsou nenulové pouze bázové splajny $B_i^k(x)$, pro $i = j-k+1, j-k+2, \ldots, j$ (celkem tedy k bázových splajnů řádu k).

(v3) Pro $x \in \langle y_j, y_{j+1}\rangle$, pro $j \in \{k-1, \ldots, n+1-k\}$ platí

$$\sum_i B_i^k(x) = 1.$$

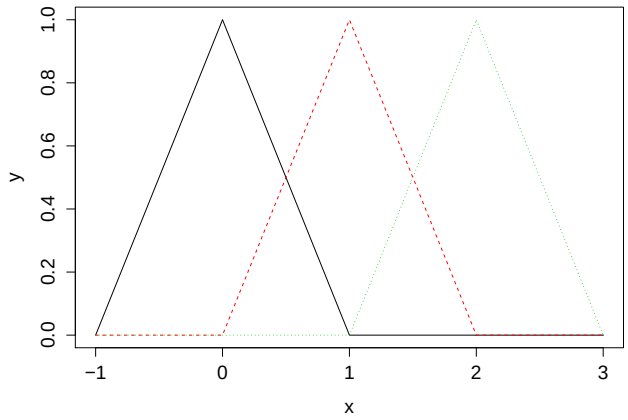
B-splajny tvoří tzv. rozklad jednotky.

Poznámka 1.5. Poslední uváděná vlastnost **(v3)** je dobře patrná z následujícího obrázku. Uvažujme síť uzlů $(\Delta y) = \{-1, 0, 1, 2, 3\}$ a řád B-splajnu $k = 2$. V tomto případě jsme na síti uzlů (Δy) schopni sestavit tři lineární B-splajny $B_i^2(x)$, pro

$i = 1, 2, 3$. Vidíme tedy, že součet funkčních hodnot B-splajnů je roven jedné i mimo uzly sítě (Δy). Například pro bod 1.5 platí, že

$$\sum_{i=1}^{3} B_i^2(1.5) = 0 + \frac{1}{2} + \frac{1}{2} = 1.$$

Mimo interval $\langle 0, 2\rangle$ již tato vlastnost neplatí. Je patrné, že např. pro $\forall x \in \langle -1, 0\rangle$ nebude platit, že $\sum_{i=1}^{3} B_i^2(x) = 1$.



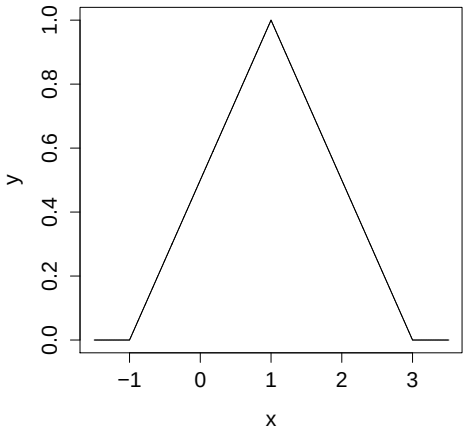
Obrázek 1: Poznámka 1.4

Názvy bázových splajnů se často odvozují od stupně polynomu. Pojmem **konstantní B-splajn** rozumíme bázový splajn $B_i^1(x)$, tj. B-splajn nultého stupně. Pojem **lineární B-splajn** značí B-splajn prvního stupně $B_i^2(x)$ a například pojem **kvadratický B-splajn** označuje B-splajn druhého stupně $B_i^3(x)$. Stejným způsobem se dají odvodit také názvy pro B-splajny vyšších stupňů.

Poznámka 1.6. Pro větší názornost zde uvedeme grafy některých B-splajnů vytvořených v softwaru R (podrobný popis generování těchto grafů, je uvedený v kapitole 5.2).

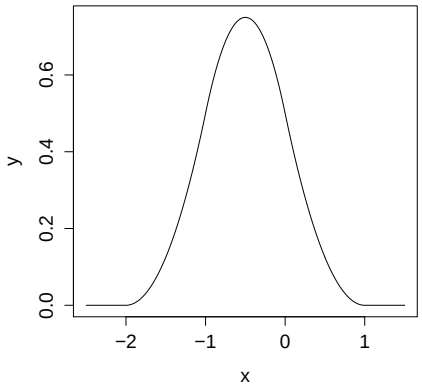
Obrázek 2 představuje graf lineárního B-splajnu $B_1^2(x)$, který je definovaný na síti uzlů $(\Delta y) = \{-1, 1, 3\}$. Na této síti uzlů můžeme kromě tohoto B-splajnu

znázornit také dva konstantní B-splajny $B_1^1(x)$ a $B_2^1(x)$. My zde však uvažujeme bázový splajn nejvyššího možného řádu, který lze na této síti sestrojit, tj. $B_1^2(x)$.



Obrázek 2: Lineární B-splajn $B_1^2(x)$

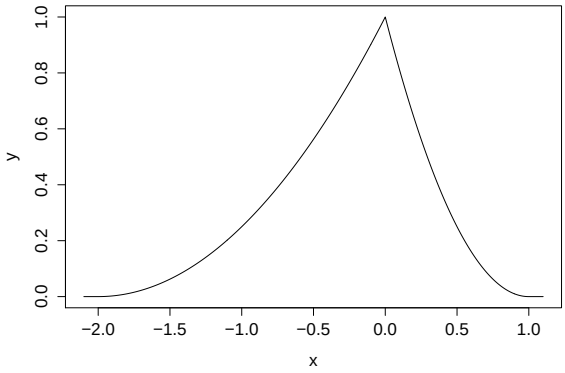
Obrázek 3 znázorňuje kvadratický B-splajn $B_1^3(x)$. Tento B-splajn je definovaný na síti uzlů $(\Delta y) = \{-2, -1, 0, 1\}$ a jedná se opět o bázový splajn nejvyššího možného řádu, který lze na této síti uzlů vytvořit.



Obrázek 3: Kvadratický B-splajn $B_1^3(x)$

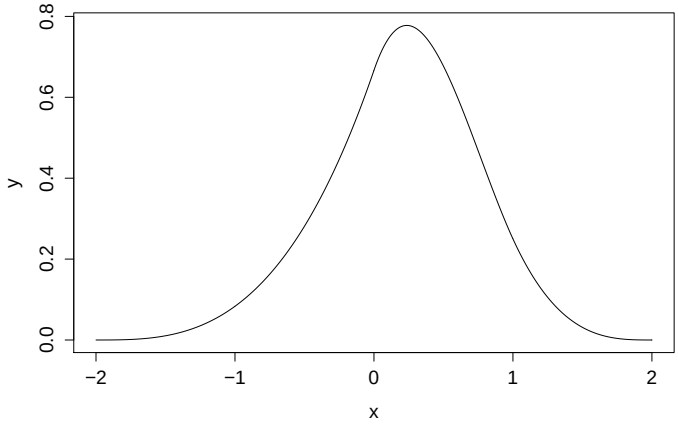
U B-splajnů $B_1^2(x)$ i $B_1^3(x)$, které jsme si v rámci této poznámky představili, jsme uvažovali defekt $d = 1$. Pro zajímavost se nyní podíváme, jak bude vypadat graf kvadratického B-splajnu $B_1^3(x)$, zvolíme-li defekt $d = 2$. Budeme tedy uvažo-

vat násobné uzly. Tato násobnost sníží stupeň diferencovatelnosti v daném uzlu a ovlivní v něm spojitost. Na Obr. 4 je znázorněn kvadratický B-splajn definovaný na síti uzlů $(\Delta y) = \{-2, 0, 0, 1\}$, tj. uvažujeme jeden dvojnásobný uzel.



Obrázek 4: Kvadratický B-splajn $B_1^3(x)$

Z grafu B-splajnu je patrné, že násobnost uzlu narušila spojitost v derivaci a v daném uzlu zůstala spojitá pouze funkční hodnota. Budeme-li ovšem uvažovat kubický B-splajn $B_1^4(x)$ definovaný na síti uzlů $(\Delta y) = \{-2, 0, 0, 1, 2\}$ s defektem $d = 2$, sníží se nám sice hladkost v násobném uzlu, ale zůstane v něm spojitá derivace do řádu $(4 - 1) - 2$, tedy do prvního řádu.



Obrázek 5: Kubický B-splajn $B_1^4(x)$

Nyní se opět vraťme k lineárnímu prostoru splajnů $S_{md}(\Delta x)$, který jsme si

zavedli v kapitole věnované pp-reprezentaci. Uvažujme tedy splajn $S_{md}(x)$, který je definovaný na síti uzlů $(\Delta x) = \{x_0, \ldots, x_n, x_{n+1}\}$. V návaznosti na Poznámku 1.1 je zřejmé, že umíme určit dimenzi takto zavedeného prostoru $S_{md}(\Delta x)$. Pro připomenutí, dimenzi určíme ze vztahu

$$\dim S_{md}(\Delta x) = m + nd + 1.$$

Budeme tedy uvažovat prostor $S_{md}(\Delta x)$ jen s tím rozdílem, že nyní použijeme k vyjádření splajnů jejich B-splajnovou reprezentaci. Ve [2] je na straně 109 uvedeno, že posloupnost B-splajnů $\left\{B_i^k(x)\right\}$ tvoří **bázi** a jednotlivé B-splajny jsou lineárně nezávislé. Každý B-splajn a také jejich libovolná lineární kombinace, kterou budeme označovat $s_{k-1}(x)$, je po částech polynom stupně nejvýše $k-1$. Je zřejmé, že splajn $s_{k-1}(x)$ je totožný s již dříve zavedeným splajnem $S_{md}(x)$, dle Definice 1.1. Jediný rozdíl mezi nimi je v tom, že nyní nebudeme explicitně zmiňovat defekt splajnu a pro jednoduchost jej budeme v této chvíli uvažovat roven jedné.

Vzhledem k tomu, že uvažujeme stále stejný prostor splajnů, zůstane stejná také jeho dimenze. Mějme například síť uzlů $(\Delta x) = \{0, 1, 2\}$ a řád B-splajnu $k = 2$. Dimenze prostoru splajnů bude totožná, jako by byla v případě pp-reprezentace, tedy $\dim S_{md}(\Delta x) = 4 - 1 = 3$. Jednalo by se totiž o splajn určený dvěma polynomy:

$$P_0(x) = a_0 x + b_0 \quad \text{pro} \quad x \in \langle x_0, x_1 \rangle ,$$

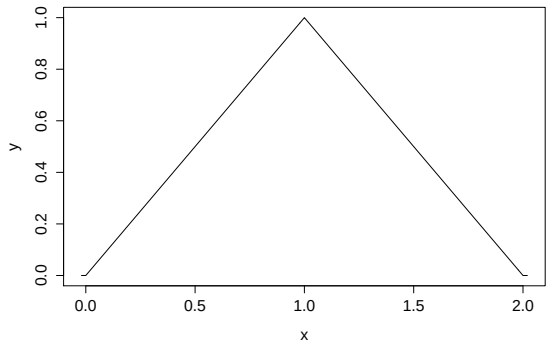
$$P_1(x) = a_1 x + b_1 \quad \text{pro} \quad x \in \langle x_1, x_2 \rangle ,$$

tj. máme 4 neznámé koeficienty a_i, b_i, pro $i = 0, 1$ a v případě defektu $d = 1$ by platila tato podmínka spojitosti:

$$P_0(x_1) = P_1(x_1).$$

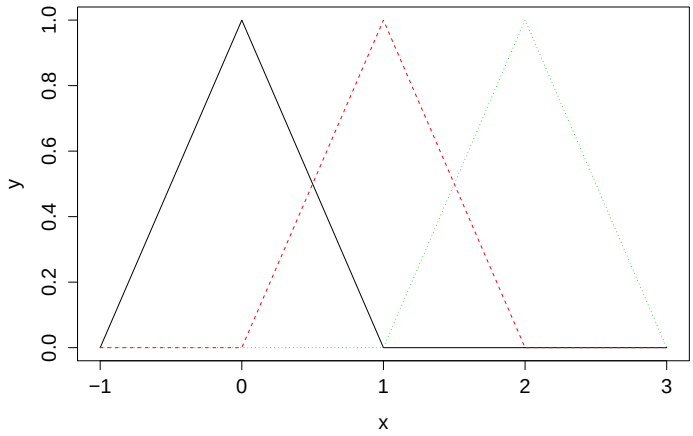
Naším cílem je tedy na (Δx) sestavit 3 bázové funkce odpovídající B-splajnům $B_1^2(x), B_2^2(x)$ a $B_3^2(x)$. Vykreslíme-li si ale graf bázových funkcí odpovídající síti

uzlů (Δx), je z následujícího obrázku patrné, že na uzlech x_0, x_1 a x_2 lze zkonstruovat jediný B-splajn $B_1^2(x)$. Na síti uzlů (Δx) tedy nejsme schopni sestavit tolik B-splajnů, aby tvořily celou bázi tohoto prostoru.



Obrázek 6: Graf bázové funkce odpovídající síti (Δx)

Z toho důvodu je nutné přejít k nové síti uzlů (Δy), na které již budeme schopni zavést kompletní bázi. Prostor splajnů $s_{k-1}(x)$, definovaných na této síti (Δy), si označíme $S^k(\Delta y)$ a je tedy zřejmé, že $S_{md}(\Delta x) = S^k(\Delta y)$. Z původní sítě tedy vytvoříme novou síť, např. $(\Delta y) = \{-2, -1, 0, 1, 2, 3, 4\}$. Jak ukazuje Obr. 7, na této síti již jsme schopni definovat celou bázi prostoru splajnů $S^k(\Delta y)$.



Obrázek 7: Grafy bázových funkcí prostoru $S^2(\Delta y)$

Následující Věta 1.1 nám dává návod, jak zvolit posloupnost uzlů (Δy), aby

odpovídala požadavkům na spojitost splajnu (viz Poznámka 1.1) a jednotlivým defektům v uzlech x_i. Při tvorbě této posloupnosti vycházíme z původní rostoucí posloupnosti uzlů $(\Delta x) = \{x_i, i = 0, \ldots, n+1\}$.

Věta 1.1. ***(Curry, Schoenberg)*** *Nechť $(\Delta x) = \{x_i; i = 0, \ldots, n+1\}$ je monotónně rostoucí posloupnost uzlů splajnu a složky vektoru $\boldsymbol{d} = (d_1, \ldots, d_n)^T$ udávají defekt splajnu v uzlech x_i, kde d_i jsou celá čísla, pro $i = 1, \ldots, n$. Symbol $P_{k-1}(x, \boldsymbol{d})$ značí lineární prostor po částech polynomických funkcí stupně nejvýše $k-1$, definovaných na síti uzlů (Δx) s defektem d_i v uzlu x_i, pro $i = 1, \ldots, n$. Dále nechť je síť uzlů $(\Delta y) = \{y_1, \ldots, y_{N+k}\}$ vytvořena ze sítě (Δx) následujícím způsobem:*

1. $y_1 \le y_2 \le \ldots \le y_k \le x_0$;
   $x_{n+1} \le y_{N+1} \le \ldots \le y_{N+k}$,
   $N = k + \sum_{i=1}^{n} d_i$;

2. *uzel $x_i, i = 1, \ldots, n$ se vyskytuje v síti (Δy) právě d_i-krát.*

Pak platí, že dimenze prostoru $P_{k-1}(x, \boldsymbol{d})$ je rovna N.
Posloupnost $\left\{B_i^k(x)\right\}$ bázových splajnů na síti (Δy) tvoří bázi prostoru $P_{k-1}(x, \boldsymbol{d})$ na intervalu $\langle y_k, y_{N+1}\rangle$, tj. $S^k(\Delta y) = P_{k-1}(x, \boldsymbol{d})$.

Věta 1.1 je převzata z [2], strana 111.

Poznámka 1.7. Nadále budeme předpokládat, že interval $\langle y_k, y_{N+1}\rangle$ je totožný s intervalem $\langle x_0, x_{n+1}\rangle$. Zjednodušeně jej budeme označovat $\langle a, b\rangle$. Platí tedy, že

$$a = x_0 = y_k, \qquad b = x_{n+1} = y_{N+1.}$$

Ze znění Věty 1.1 je také patrné, že prostor splajnů $S^k(\Delta y)$, který jsme si dříve zavedli, je na intervalu $\langle a, b\rangle$ totožný s prostorem splajnů $P_{k-1}(x, \boldsymbol{d})$.

Poznámka 1.8. Síť (Δy), vytvořená dle Věty 1.1, se nazývá **rozšířená síť uzlů** a uzly

$$y_1, y_2, \ldots, y_{k-1}, \qquad y_{N+2}, y_{N+3}, \ldots, y_{N+k}$$

nazýváme **rozšiřující uzly**.

Pro větší názornost si ukážeme tvorbu této rozšířené sítě uzlů pomocí softwaru R. Rozšiřující uzly (Δy) je možné volit libovolně, my se ovšem zaměříme na dva nejpoužívanější způsoby:

- Má-li smysl uvažovat uzly nalevo od a a napravo od b, můžeme je zvolit např. ekvidistantně. To znamená, že původní posloupnost (Δx) rozšíříme z obou stran o $k-1$ uzlů. Uvažujme tedy $(\Delta x) = \{-2, 0, 8\}$, řád splajnu $k = 3$ a počet prvků v původní posloupnosti si označíme symbolem r, v našem případě tedy $r = 3$. Délku kroku budeme uvažovat rovnu jedné. K vytvoření ekvidistantní sítě (Δy) použijeme následující for cyklus, který jsem k tomuto účelu v R naprogramovala:

```
Celkova_Delka = 2*(k-1) + r
y = c()
for (i in 1:Celkova_Delka) {
if (i <= (k-1)) { y[i] = x[1]-(k-1)-1+i }
if ((i > (k-1))&(i <= (r+(k-1)))) { y[i] = x[i-(k-1)] }
if (i > (r+(k-1))) { y[i] = x[r]+i-r-(k-1) } }
```

  Výsledkem je posloupnost ve tvaru $(\Delta y) = \{-4, -3, -2, 0, 8, 9, 10\}$. Chceme-li, aby přidané rozšiřující uzly měly mezi sebou konstantní vzdálenost, která je rovna vzdálenosti mezi uzly x_{n-1} a x_n, použijeme k vytvoření (Δy) for cyklus:

```
Celkova_Delka = 2*(k-1)+r
krok = x[r]-x[r-1]
y = c()
for (i in 1:Celkova_Delka){
if (i < (k-1)){y[i] = x[1]-krok*((k-i))}
if (i == (k-1)){y[i] = x[1]-krok}
if ((i > (k-1))&(i <= (r+(k-1)))){y[i] = x[i-(k-1)]}
if (i > (r+(k-1))){ y[i] = x[r]+krok*(i-r-(k-1))}}
```

  Tedy pro $(\Delta x) = \{-15, 15, 45\}$ a $k = 2$ dostaneme rozšířenou síť uzlů ve tvaru $(\Delta y) = \{-45, -15, 15, 45, 75\}$.

- Dalším častým způsobem, jak zvolit rozšiřující uzly, je pomocí uzlů násobných. V takovém případě se tyto rozšiřující uzly rovnají krajním bodům (Δx). Také zde budeme uvažovat $(\Delta x) = \{-2, 0, 8\}$, $k = 3$ a $r = 3$. Pro tento způsob tvorby (Δy) vypadá příkaz v R následovně:

```
y = c()
if (k==1){
for (i in 1:r){y[i] = x[i]}}
if (k != 1){
Celkova_Delka = 2*(k-1) + r
y[1:(k-1)] = x[1]
y[k:(k+r-1)] = x[1:r]
y[(k+r):Celkova_Delka] = x[r]}
```

  Výsledná posloupnost bude nyní ve tvaru $(\Delta y) = \{-2, -2, -2, 0, 8, 8, 8\}$.

Poznámka 1.9. Vytvoříme-li tedy rozšířenou síť uzlů (Δy) z dané rostoucí posloupnosti (Δx), pak dimenze prostoru $S^k(\Delta y)$ je rovna N, k je řád splajnu, $k-1$ je jeho stupeň a složky vektoru $\boldsymbol{d} = (d_1, \ldots, d_n)^T$ udávají defekt spajnu v uzlech x_i, pro $i = 1, \ldots, n$. Zvolíme-li si např. defekt $d_i = 1$, pro $i = 1, \ldots, n$, bude síť uzlů vypadat takto

$$y_1 \leq \ldots \leq y_k \leq x_0 < \ldots < x_{n+1} \leq y_{N+1} \leq \ldots \leq y_{N+k}.$$

Bázi prostoru $S^k(\Delta y)$ tvoří posloupnost bázových splajnů $\left\{B_i^k(x), i = 1, \ldots, N\right\}$ a proto lze splajn $s_{k-1}(x)$ vyjádřit jako lineární kombinaci bázových funkcí

$$s_{k-1}(x) = \sum_{i=1}^{N} b_i B_i^k(x),$$

kde b_i jsou koeficienty splajnu $s_{k-1}(x)$ vzhledem k bázi $\left\{B_i^k(x), i = 1, \ldots, N\right\}$. Každý splajn $s_{k-1}(x) \in S^k(\Delta y)$ je tedy jednoznačně určen vektorem B-splajnových koeficientů $\boldsymbol{b} = (b_1, \ldots, b_N)^T$. Vyjádříme-li si splajn maticově, potom

$$s_{k-1}(x) = \boldsymbol{K}_k(x)\boldsymbol{b},$$

kde

$$\boldsymbol{K}_k(x) = (B_1^k(x) \dots B_N^k(x))$$

je řádkový vektor funkčních hodnot jednotlivých B-splajnů v bodě x. Nyní budeme uvažovat vektor $\boldsymbol{t} \in \mathbb{R}^m$ se složkami $\boldsymbol{t} = (t_1, \dots, t_m)^T$. Funkční hodnoty B-splajnů v těchto bodech t_j, pro $j = 1, \dots, m$ lze zapsat do tzv. **kolokační matice** tvaru

$$\boldsymbol{K}_k(\boldsymbol{t}) = \begin{pmatrix} B_1^k(t_1) & B_2^k(t_1) & \dots & B_N^k(t_1) \\ B_1^k(t_2) & B_2^k(t_2) & \dots & B_N^k(t_2) \\ \vdots & \vdots & \ddots & \vdots \\ B_1^k(t_m) & B_2^k(t_m) & \dots & B_N^k(t_m) \end{pmatrix} = (B_i^k(t_j))_{j=1,i=1}^{m\ N}.$$

Na rozšířené síti uzlů $(\Delta y) = \{y_1, \dots, y_{N+k}\}$ je poměrně problematické pracovat s přesným zněním definice bázových splajnů (viz Definice 1.2) a to zejména kvůli výpočtům poměrných diferencí. Z toho důvodu se pro práci s nimi využívají rekurentní vztahy pro výpočet hodnot bázových splajnů.

Věta 1.2. *Pro hodnoty bázových splajnů se používají následující rekurentní vztahy*

$$B_i^1(x) = \begin{cases} 1 & pro \quad y_i \le x < y_{i+1} \\ 0 & jinak \end{cases}$$

$$B_i^k(x) = \frac{x - y_i}{y_{i+k-1} - y_i} B_i^{k-1}(x) + \frac{y_{i+k} - x}{y_{i+k} - y_{i+1}} B_{i+1}^{k-1}(x), \quad i = 1, \dots, N, \quad k = 2, 3, \dots$$

Důkaz: viz [2], strana 112.

V literatuře [2] najdeme i následující uváděné poznatky, týkající se **derivování splajnů**.

Pokud budeme splajn derivovat, snížíme tím jeho řád. Například první derivace splajnu $s_{k-1}(x) = \sum_{i=1}^{N} b_i B_i^k(x)$ je rovna výrazu

$$\frac{d}{dx}s_{k-1}(x) = \frac{d}{dx}\sum_{i=1}^{N} b_i B_i^k(x) = \sum_{i=2}^{N}(k-1)\frac{b_i - b_{i-1}}{y_{i+k-1} - y_i}B_i^{k-1}(x) =$$

$$= \sum_{i=2}^{N} b_i^{(1)} B_i^{k-1}(x) = s_{k-2}(x),$$

kde b_i jsou koeficienty původního splajnu $s_{k-1}(x)$ a $b_i^{(1)} = (k-1)\frac{b_i - b_{i-1}}{y_{i+k-1} - y_i}$ jsou koeficienty splajnu $s_{k-2}(x)$.

Je tedy vidět, že derivací splajnu řádu k získáme splajn řádu $k-1$. Opakujeme-li tento postup, dostaneme vztah pro výpočet j-té derivace splajnů

$$\frac{d^{(j)}}{dx^{(j)}}\sum_{i=1}^{N} b_i B_i^k(x) = \sum_{i=1+j}^{N} b_i^{(j)} B_i^{k-j}(x),$$

s koeficienty $b_i^{(0)} = b_i, \quad b_i^{(j)} = \frac{b_i^{(j-1)} - b_{i-1}^{(j-1)}}{y_{i+k-j} - y_i}, \quad$ pro $i = 1, \ldots, N$ a $j = 1, \ldots, k-1$. Maticově můžeme tuto j-tou derivaci splajnu zapsat ve tvaru

$$s_{k-1}^{(j)}(x) = \boldsymbol{K}_k(x)\boldsymbol{b}^{(j)},$$

kde

$$\begin{aligned}
\boldsymbol{K}_k(x) &= (B_1^k(x) \ldots B_N^k(x))^T \in \mathbb{R}^N \\
\boldsymbol{b}^{(0)} &= \boldsymbol{b} \in \mathbb{R}^N \\
\boldsymbol{b}^{(j)} &= \boldsymbol{D}_j \boldsymbol{L}_j \boldsymbol{b}^{(j-1)} \in \mathbb{R}^{N-j} \\
&= \boldsymbol{D}_j \boldsymbol{L}_j \ldots \boldsymbol{D}_1 \boldsymbol{L}_1 \boldsymbol{b} \\
&= \boldsymbol{S}_j \boldsymbol{b},
\end{aligned}$$

přičemž $\boldsymbol{S}_j$ je horní trojúhelníková matice plné řádkové hodnosti, která je definovaná vztahem

$$\begin{aligned}
\boldsymbol{S}_0 &= \boldsymbol{I}_N \\
\boldsymbol{S}_j &= \boldsymbol{D}_j \boldsymbol{L}_j \ldots \boldsymbol{D}_1 \boldsymbol{L}_1 \in \mathbb{R}^{N-j,N},
\end{aligned}$$

kde $\boldsymbol{I}_N \in \mathbb{R}^{N,N}$ je jednotková matice,

$$\boldsymbol{D}_l = (k-l)\,\mathrm{diag}\left\{\frac{1}{y_{i+k-l}-y_i}\right\}_{i=1+l}^{N} \in \mathbb{R}^{N-l,N-l},$$

a

$$\boldsymbol{L}_l = \begin{pmatrix} -1 & 1 & & \\ & \ddots & \ddots & \\ & & -1 & 1 \end{pmatrix} \in \mathbb{R}^{N-l,N+1-l}.$$

Předešlé vztahy jsou převzaty ze [4] na str. 33-34 a byly odvozeny na základě následující Věty 1.3, která nám říká, jaký je vztah mezi koeficienty původního a derivovaného splajnu.

Věta 1.3. *Splajn $\sum_i b_i B_i^k(x)$ je derivací splajnu $\sum_i c_i B_i^{k+1}(x)$, jestliže mezi koeficienty b_i, c_i platí vztahy*

$$\frac{k(c_i - c_{i-1})}{y_{i+k}-y_i} = b_i, \quad tj. \quad c_i = c_{i-1} + \frac{b_i(y_{i+k}-y_i)}{k},$$

pro $i = 1, \ldots, N$, kde jeden z parametrů c_i je volitený.

2. Interpolace

Nejprve si ujasníme, v čem spočívá základní myšlenka interpolace. Mějme dáno m navzájem různých bodů $t_i \in \mathbb{R}$, pro $i = 1, \ldots, m$. Dále mějme dané funkční hodnoty v těchto bodech, které budeme označovat $f(t_i)$. Úkolem interpolace je najít funkci $\psi(x, a_0, \ldots, a_n)$ proměnné x, která je závislá na parametrech $a_0, \ldots, a_n$. Tyto parametry musí být zvoleny tak, aby funkce ψ splňovala interpolační podmínky

$$\psi(t_i, a_0, \ldots, a_n) = f(t_i), \qquad i = 1, \ldots, m.$$

Existuje více možností, jak si zvolit interpolační funkci ψ. Nejpoužívanější je tzv. **lineární interpolace**, kde interpolační funkci hledáme ve tvaru

$$\psi(x, a_0, \ldots, a_n) = a_0\psi_0(x) + \ldots + a_n\psi_n(x).$$

Do této interpolace řadíme také hodně užívanou **polynomickou interpolaci**, kde hledaná funkce ψ je polynomem stupně nejvýše n. Hledaný polynom je potom ve tvaru

$$P_n(x) = \psi(x, a_0, \ldots, a_n) = a_0x^n + a_1x^{n-1} + \ldots + a_n.$$

Tento interpolační polynom je možné najít například pomocí formule Lagrangeova interpolačního polynomu.

Dalším typem interpolace, kterému se budeme v následujícím textu podrobněji věnovat, je **interpolace pomocí splajnů**. V tomto případě hledáme funkci ψ ve tvaru po částech polynomu. V praxi se ukázalo, že je výhodnější řešit interpolační úlohy použitím funkce, která je po částech polynom nízkého stupně a jejíž jednotlivé části na sebe dostatečně hladce navazují. Zejména pokud máme předepsaný větší počet dat, je praktičtější použít tento způsob interpolace, protože se tím vyhneme polynomům vysokých stupňů.

2.1. Interpolace B-splajny

Nechť je dána rozšířená síť uzlů splajnu (Δy) pro kterou platí $y_i < y_{i+k}$ pro $\forall i = 1, \ldots, N$ a kde k značí přirozené číslo udávající řád splajnu (jedná se tedy o

po částech polynom stupně nejvýše $k-1$). Dále nechť je dána rostoucí posloupnost bodů interpolace $(\Delta t) = \{t_1, \ldots, t_m\}$, ke kterým jsou dané odpovídající funkční hodnoty $f(t_j)$, jenž budeme označovat zkráceně f_j, pro $j = 1, \ldots, m$. Naším úkolem je najít na intervalu $\langle a, b\rangle$ **interpolační splajn** $s_{k-1}(x) \in S^k(\Delta y)$, který je ve tvaru

$$s_{k-1}(x) = \sum_{i=1}^{N} B_i^k(x) b_i,$$

a který splňuje v bodech t_j podmínky interpolace, tj.

$$s_{k-1}(t_j) = \sum_{i=1}^{N} B_i^k(t_j) b_i = f_j \qquad j = 1, \ldots, m.$$

Připomeneme si, že pro krajní body intervalu $\langle a, b\rangle$ uvažujeme rovnost

$$a = x_0 = y_k, \qquad b = x_{n+1} = y_{N+1.}$$

Splajn $s_{k-1}(x)$ je jednoznačně určený vektorem B-splajnových koeficientů $\boldsymbol{b} = (b_1, \ldots, b_N)^T$. Z interpolačních podmínek dostaneme pro tyto hledané koeficienty b_i soustavu m lineárních rovnic o N neznámých $b_1, \ldots, b_N$, kterou zapíšeme ve tvaru $\boldsymbol{K}_k \boldsymbol{b} = \boldsymbol{f}$. Kde

$$\boldsymbol{K}_k = (B_i^k(t_j))_{j=1,i=1}^{m\ N}$$

představuje kolokační matici funkčních hodnot jednotlivých B-splajnů v bodech interpolace $t_j, j = 1, \ldots, m$. Dále $\boldsymbol{b}$ je vektor hledaných koeficientů b_i a $\boldsymbol{f}$ je vektor funkčních hodnot f_j v bodech interpolace t_j, pro $j = 1, \ldots, m$. Rozepsaná soustava lineárních rovnic $\boldsymbol{K}_k \boldsymbol{b} = \boldsymbol{f}$ vypadá následovně:

$$\begin{pmatrix} B_1^k(t_1) & B_2^k(t_1) & \ldots & B_N^k(t_1) \\ B_1^k(t_2) & B_2^k(t_2) & \ldots & B_N^k(t_2) \\ \vdots & \vdots & \ddots & \vdots \\ B_1^k(t_m) & B_2^k(t_m) & \ldots & B_N^k(t_m) \end{pmatrix} \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_N \end{pmatrix} = \begin{pmatrix} f_1 \\ f_2 \\ \vdots \\ f_m \end{pmatrix}.$$

Jednoznačnost řešení této soustavy rovnic a tedy také jednoznačné určení hledaného interpolačního splajnu je závislé na tom, zda je matice $\boldsymbol{K}_k$ čtvercová regulární. V takovém případě existuje právě jedno řešení $\boldsymbol{b}^* = \boldsymbol{K}_k^{-1}\boldsymbol{f}$, kde $\boldsymbol{b}^* = (b_1^*, \ldots, b_N^*)^T$ je vektor B-splajnových koeficientů a existuje tedy právě jeden interpolační splajn

$$s_{k-1}^*(x) = \sum_{i=1}^{N} B_i^k(x) b_i^*.$$

Následující věta udává, za jakých podmínek je kolokační matice regulární a kdy má plnou sloupcovou hodnost. Jak již bylo řečeno, je-li matice $\boldsymbol{K}_k$ regulární, potom existuje právě jedno řešení soustavy $\boldsymbol{K}_k\boldsymbol{b} = \boldsymbol{f}$ a existuje právě jeden interpolační splajn $s_{k-1}^*(x)$. Větu 2.1 využijeme ale i v následujících kapitolách 3 a 4, tj. v metodě nejmenších čtverců a ve vyhlazujících splajnech. Také v rámci těchto metod budeme hledat aproximační splajn $s_{k-1}^*(x)$ na daném intervalu $\langle a, b\rangle$. Jednou z podmínek existence tohoto splajnu bude právě plná sloupcová hodnost matice $\boldsymbol{K}_k$.

Věta 2.1. ***(Schoenberg, Whitney)*** *Nechť je dána neklesající posloupnost uzlů splajnu* $(\Delta y) = \{y_1, \ldots, y_{N+k}\}, k \in \mathbb{N}$ *a posloupnost B-splajnů* $\left\{B_i^k(x)\right\}_{i=1}^N$ *řádu* k*. Dále nechť je dána rostoucí posloupnost bodů* $(\Delta t) = \{t_1, \ldots, t_m\}$ *a posloupnost příslušných funkčních hodnot* f_j *v bodech* t_j*. Pak pro matici* $\boldsymbol{K}_k = (B_i^k(t_j))_{j=1,i=1}^{m\ N}$ *platí následující tvrzení:*

1. *Matice* $\boldsymbol{K}_k$ *je regulární právě tehdy, když* $m = N$ *a zároveň*

   $$B_i^k(t_i) \neq 0 \qquad i = 1, \ldots, N,$$

   tedy jestliže platí $y_i < t_i < y_{i+k}$ *pro* $i = 1, \ldots, N$*.*

2. *Matice* $\boldsymbol{K}_k$ *je plné sloupcové hodnosti právě tehdy, když* $m > N$ *a existuje podmnožina* $\{u_1, \ldots, u_N\} \subset \{t_1, \ldots, t_m\}$*, kde* $u_i < u_{i+1}$ *pro* $i = 1, \ldots, N-1$*, taková, že*

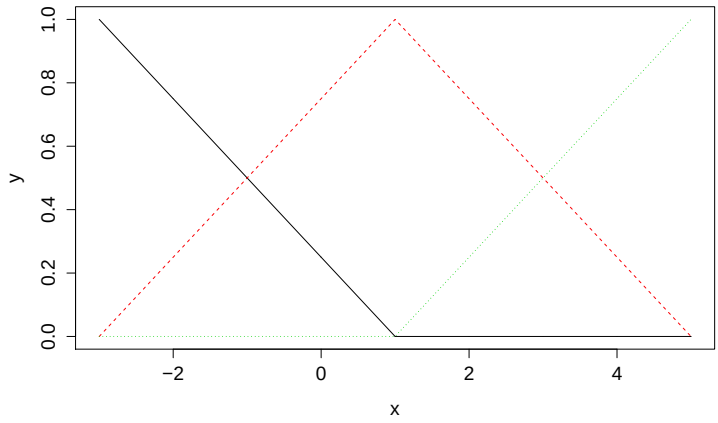
   $$y_i < u_i < y_{i+k} \qquad i = 1, \ldots, N.$$

Důkaz:

1. viz [2], strana 116,

2. viz [4], strana 32.

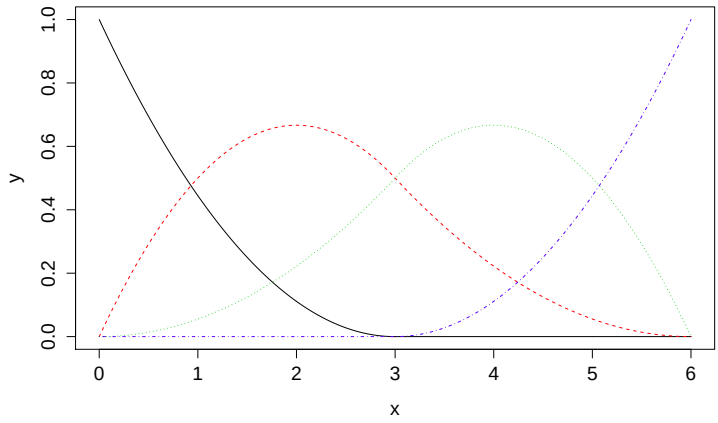
Poznámka 2.1. První část Věty 2.1 vlastně říká, že kolokační matice $\boldsymbol{K}_k$ je regulární právě tehdy, když i-tý bod posloupnosti $(\Delta t) = \{t_1, \ldots, t_m\}$ leží v nosiči B-splajnu $B_i^k(x)$. Tedy například, uvažujme lineární B-splajn definovaný na síti uzlů $(\Delta x) = \{-3, 1, 5\}$ a posloupnost bodů interpolace $(\Delta t) = \{-1, 0, 4\}$. Abychom byli schopni sestavit celou bázi prostoru splajnů, musíme od sítě (Δx) přejít k rozšířené síti uzlů, kterou budeme uvažovat např. ve tvaru $(\Delta y) = \{-3, -3, 1, 5, 5\}$.



Obrázek 8: B-splajny $B_1^2(x), \ldots, B_3^2(x)$

Nejprve požadujeme, aby $m = N$, kde m je počet bodů interpolace a N je dimenze prostoru splajnů $S^2(\Delta y)$. V našem případě $m = N = 3$. Dále požadujeme, aby pro každé $i = 1, 2, 3$ platila nerovnost $y_i < t_i < y_{i+k}$. Z grafu je patrné, že bod $t_1 = -1$ leží v nosiči B-splajnu $B_1^2(x)$ černé barvy, bod $t_2 = 0$ leží v nosiči B-splajnu $B_2^2(x)$ červené barvy a konečně bod $t_3 = 4$ leží v nosiči B-splajnu $B_3^2(x)$ zelené barvy. Můžeme tedy říct, že zvolíme-li uzly splajnu a body interpolace tímto způsobem, matice $\boldsymbol{K}_k$ bude regulární.

Druhá část Věty 2.1. říká, že pro posloupnost bodů aproximace $(\Delta t) = \{t_1, \ldots, t_m\}$ je kolokační matice $\boldsymbol{K}_k$ plné sloupcové hodnosti právě tehdy, když lze z (Δt) vybrat rostoucí podposloupnost $(\Delta u) = \{u_1, \ldots, u_N\}$ takovou, že i-tý bod této podposloupnosti leží právě v nosiči B-splajnu $B_i^k(x)$, pro $i = 1, \ldots, N$. Pro jednoduchost mějme síť uzlů $(\Delta x) = \{0, 3, 6\}$, řád $k = 3$ a body aproximace $(\Delta t) = \{1, 2, 3, 4, 5\}$. Z toho důvodu, že na síti uzlů (Δx) nejsme schopni sestavit kompletní bázi prostoru splajnů, je nutné přejít k rozšířené síti uzlů, kterou budeme uvažovat např. ve tvaru $(\Delta y) = \{0, 0, 0, 3, 6, 6, 6\}$.



Obrázek 9: B-splajny $B_1^3(x), \ldots, B_4^3(x)$

Nyní požadujeme, aby $m > N$, kde m je opět počet bodů aproximace a N udává dimenzi prostoru splajnů $S^3(\Delta y)$, tj. $5 > 4$. Zároveň uvažujme například rostoucí podposloupnost $(\Delta u) = \{1, 2, 3, 5\}$, tj. $(\Delta u) \subset (\Delta t)$. Pro body (Δu) platí nerovnost $y_i < u_i < y_{i+k}$, pro $\forall i = 1, \ldots, 4$. Z grafu bázových funkcí je patrné, že bod $u_1 = 1$ leží v nosiči B-splajnu $B_1^3(x)$ černé barvy, bod $u_2 = 2$ leží v nosiči B-splajnu $B_2^3(x)$ červené barvy, bod $u_3 = 3$ leží v nosiči B-splajnu $B_3^3(x)$ zelené barvy a konečně bod $u_4 = 5$ leží v nosiči B-splajnu $B_4^3(x)$ modré barvy. Můžeme tedy říct, že zvolíme-li uzly splajnu a body aproximace tímto způsobem, matice $\boldsymbol{K}_k$ bude plné sloupcové hodnosti.

Pro vykreslení grafu hledaného interpolačního splajnu jsem v R naprogramovala funkce `IS1` a `IS2`, do kterých uživatel na vstupu zadá síť uzlů (Δx), posloup-

nost bodů interpolace (Δt), funkční hodnoty v těchto bodech a řád splajnu k. Kromě samotného grafu interpolačního splajnu $s^*_{k-1}(x)$ dostane uživatel na výstupu funkce také vektor B-splajnových koeficientů, jímž je výsledný interpolační splajn jednoznačně určen.

Rozdíl mezi funkcemi `IS1` a `IS2` je v tom, jakým způsobem se ověřuje regularita kolokační matice $\boldsymbol{K}_k$ a jak se vytváří rozšířená síť uzlů (Δy) z původní posloupnosti (Δx). V případě funkce `IS1`, kde (Δy) je tvořena přidáním $k-1$ ekvidistantních uzlů z obou stran (Δx), stačí pro ověření regularity matice $\boldsymbol{K}_k$ splnění nerovnosti $y_i < t_i < y_{i+k}$, pro každé $i = 1, \ldots, N$. Nicméně, jak uvidíme na Příkladu 2.2, u funkce `IS2` nám takové ověření regularity nemusí vždy stačit. Zde se totiž (Δy) vytváří přidáním násobných uzlů a je tedy zřejmé, že může nastat situace, kdy

$$t_1 = y_1 \qquad \text{a} \qquad t_i = y_{i+k},$$

nebude tak splněna nerovnost $y_i < t_i < y_{i+k}$ pro každé $i = 1, \ldots, N$. V případě funkce `IS2` budeme tedy pro ověření regularity kolokační matice $\boldsymbol{K}_k$ požadovat, aby funkční hodnota i-tého B-splajnu byla v bodě interpolace t_i nenulová, tj. $B^k_i(t_i) \neq 0$, pro $\forall i = 1, \ldots, N$, viz Věta 2.1. Před samotným spuštěním funkce je nutné načíst příslušnou knihovnu příkazem `library(splines)`. Algoritmus pro vykreslení interpolačního splajnu $s^*_{k-1}(x)$ je následující:

```
IS1 = function(x,t,f,k){
# VSTUP: x = sit uzlu,
# t = posloupnost bodu interpolace,
# f = funkcni hodnoty v bodech interpolace,
# k = rad splajnu.
# VYSTUP: vektor B-splajnovych koeficientu,
# graf interpolacniho splajnu.
r = length(x)
F = matrix(f)
# Overeni podminek
```

```
for (i in 1:length(t)){
if (t[i] < x[1] | t[i] > x[r]) stop ('body aproximace musi byt z
intervalu <x[1],x[r]>')}
for (i in 1:(length(t)-1)){
if (t[i] >= t[i+1]) stop ('t musi byt rostouci posloupnost')}
if (k != round(k)) stop ('rad k neni cele cislo')
if (k <= 0) stop ('rad splajnu musi byt kladne cislo')
for (i in 1:(r-1)){
if (x[i] >= x[i+1]) stop ('x musi byt rostouci posloupnost')}
# Rozsirena sit uzlu
Celkova_Delka = 2*(k-1) + r
krok = x[r]-x[r-1]
y = c()
for (i in 1:Celkova_Delka){
if (i < (k-1)){y[i] = x[1]-krok*(k-i)}
if (i == (k-1)){y[i] = x[1]-krok}
if ((i > (k-1)) & (i <= (r+(k-1)))){y[i] = x[i-(k-1)]}
if (i > (r+(k-1))){y[i] = x[r]+krok*(i-r-(k-1))}}
# Kolokacni matice K
K = splineDesign(y, t, k, outer.ok = TRUE)
# Kolokacni matice C - pro rozdeleny interval
deleni = seq(min(y), max(y), by = 0.001)
lambda = c(0:(r-1))
g = lambda[length(lambda) - 1]
# Dimenze prostoru
N = g+(k-1)+1
C = array(0, c(length(deleni),N))
l = c()
for(i in (1:N)){
for (j in 1:(k+1)){
```

```
l[j] = y[i+j-1]}
C[ ,i] = splineDesign(l, deleni, k, outer.ok = TRUE)}
# Overeni regularity kolokacni matice K
if (length(t) != N) stop ('pocet bodu interpolace se musi
rovnat dimenzi prostoru splajnu')
podminka = 0
for (i in 1:N){
podminka[i] = 0
if (y[i]<t[i] & t[i]<y[i+k]){
podminka[i] = 1}}
for (j in 1:N){
if (podminka[j] == 0) stop ('kolokacni matice neni regularni')}
# Vypocet B-splajnovych koeficientu
B = (solve(K,F))
# Interpolacni splajn
S = C%*%B
matplot(deleni, S, type="l", xlab="x", ylab = "f")
points(t,f)
B}
```

Příklad 2.1. Najděte kvadratický splajn s uzly $(\Delta x) = \{0, 2, 7, 9\}$, který interpoluje body $(\Delta t) = \{1, 3, 4, 7, 8\}$. Funkční hodnoty v bodech interpolace jsou $f = \{0.2, 1, 2, 2.5, 1\}$.

Řešení: Na tomto příkladu si ukážeme použití obou funkcí, které jsem pro vykreslení interpolačního splajnu naprogramovala. Použijeme tedy obě funkce `IS` a příkazy v R budou vypadat následovně:

```
IS1(c(0,2,7,9),c(1,3,4,7,8),c(0.2,1,2,2.5,1),3),
IS2(c(0,2,7,9),c(1,3,4,7,8),c(0.2,1,2,2.5,1),3).
```

Kromě výsledného grafu je výstupem těchto funkcí také vektor B-splajnových koeficientů, který jednoznačně určuje interpolační splajn.

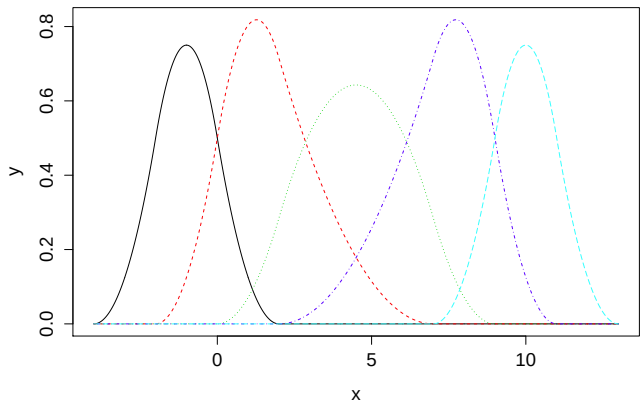
Nejprve se zaměříme na funkci `IS1`. Z algoritmu je patrné, že rozšířená síť uzlů (Δy) je tvořena přidáním $k-1$ uzlů z obou stran sítě (Δx). Výsledná posloupnost v našem případě je ve tvaru:

```
> y
[1] -4 -2 0 2 7 9 11 13
```

Pro zajímavost se můžeme podívat, jak vypadá kolokační matice $\boldsymbol{K}_k$ v bodech interpolace a také grafy příslušných B-splajnů $B_1^3(x), \ldots, B_5^3(x)$, viz Obr. 10.

```
> K
```

	[,1]	[,2]	[,3]	[,4]	[,5]
[1,]	0.125	0.8035714	0.07142857	0.00000000	0.000
[2,]	0.000	0.4571429	0.51428571	0.02857143	0.000
[3,]	0.000	0.2571429	0.62857143	0.11428571	0.000
[4,]	0.000	0.0000000	0.28571429	0.71428571	0.000
[5,]	0.000	0.0000000	0.07142857	0.80357143	0.125

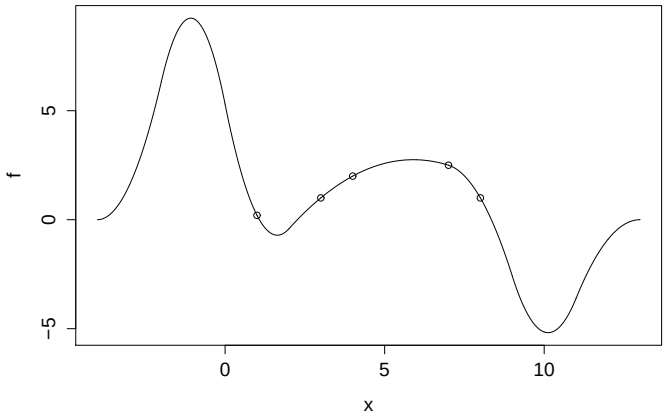


Obrázek 10: B-splajny $B_1^3(x), \ldots, B_5^3(x)$

Platí, že počet bodů interpolace je roven N, tj. pěti a zároveň je splněna nerovnost $y_i < t_i < y_{i+k}$, pro každé $i = 1, \ldots, 5$. Z grafu bázových funkcí je patrné, že i-tý bod interpolace leží v nosiči B-splajnu $B_i^3(x)$, pro každé $i = 1, \ldots, 5$. Pro takto zvolené uzly a body interpolace je tedy kolokační matice $\boldsymbol{K}_k$ regulární a funkce `IS1` vygeneruje příslušný graf interpolačního splajnu a vypíše vektor B-splajnových koeficientů.

```
> B
                [,1]
[1,]  12.641667
[2,]  -2.041667
[3,]   3.645833
[4,]   2.041667
[5,]  -7.208333
```

Výsledný graf interpolačního splajnu vidíme na následujícím obrázku.



Obrázek 11: Výstup `IS1` - interpolační splajn $s_2^*(x)$

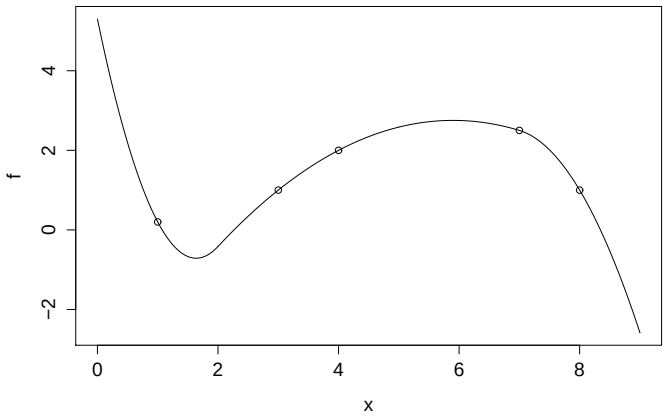
Ve funkci `IS1` je tedy nastavena volba rozšiřujících uzlů (Δy) ekvidistantně. Nabízí se samozřejmě také možnost, vytvořit tuto síť za pomoci násobných uzlů s použitím funkce `IS2`. V takovém případě by výsledná posloupnost (Δy) byla ve tvaru:

```
> y
[1] 0 0 0 2 7 9 9 9
```

Také v tomto případě jsou splněny podmínky pro regularitu kolokační matice. Víme, že sloupce matice $\boldsymbol{K}_k$ představují B-splajny $B_1^3(x), \ldots, B_5^3(x)$ a její řádky odpovídají bodům interpolace $t_1, \ldots, t_5$. Z kolokační matice je tedy zřejmé, že funkční hodnota i-tého B-splajnu je v bodě t_i nenulová, pro každé $i = 1, \ldots, N$. Můžeme tedy říct, že pro zadané uzly splajnu a body interpolace je matice $\boldsymbol{K}_k$ regulární.

```
> K
```

```
      [,1]      [,2]       [,3]       [,4]     [,5]
[1,]  0.25 0.6785714 0.07142857 0.00000000 0.00
[2,]  0.00 0.4571429 0.51428571 0.02857143 0.00
[3,]  0.00 0.2571429 0.62857143 0.11428571 0.00
[4,]  0.00 0.0000000 0.28571429 0.71428571 0.00
[5,]  0.00 0.0000000 0.07142857 0.67857143 0.25
```



Obrázek 12: Výstup `IS2` - interpolační splajn $s_2^*(x)$

Graf takto vytvořeného interpolačního splajnu je zobrazen na Obr. 12 a vektor B-splajnových koeficientů bude nyní ve tvaru:

```
> B
```

```
           [,1]
[1,]   5.300000
[2,]  -2.041667
[3,]   3.645833
[4,]   2.041667
[5,]  -2.583333
```

Srovnáme-li Obr. 11 a 12 všimneme si, že kdybychom se na Obr. 11 omezili pouze na interval $\langle 0, 9\rangle$, tj. nechali bychom tento graf vykreslit pouze na síti uzlů (Δx), oba výsledné grafy by byly totožné. Je to dáno tím, že za předpokladu splnění podmínek regularity má soustava rovnic právě jedno řešení $\boldsymbol{b}^*$ a tedy existuje také právě jeden interpolační splajn $s_2^*(x)$.

Nyní si ukážeme, jak se graf interpolačního splajnu změní, budeme-li uvažovat krajní uzly sítě (Δx) totožné s krajními body interpolace (Δt), tj.

$$x_0 = t_1, \qquad x_{n+1} = t_m.$$

Příklad 2.2. Najděte kvadratický splajn s uzly $(\Delta x) = \{0, 2, 7, 9\}$, který interpoluje body $(\Delta t) = \{0, 3, 4, 7, 9\}$. Funkční hodnoty v bodech interpolace jsou $f = \{0.2, 1, 2, 2.5, 1\}$.

Řešení: Pro vykreslení grafu interpolačního splajnu opět použijeme obě funkce `IS1` i `IS2`. Příkazy v R budou vypadat následovně:

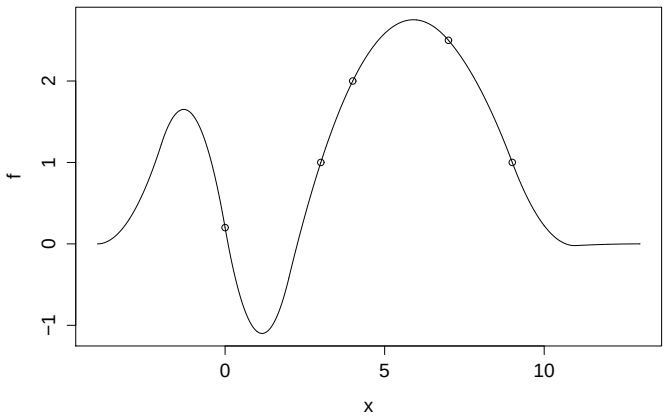
```
IS1(c(0,2,7,9),c(0,3,4,7,9),c(0.2,1,2,2.5,1),3),
IS2(c(0,2,7,9),c(0,3,4,7,9),c(0.2,1,2,2.5,1),3).
```

Nejprve ověříme, zda zadané uzly a body interpolace splňují podmínky pro regularitu kolokační matice $\boldsymbol{K}_k$. Platí, že počet bodů interpolace je roven N, tj. pěti. Dále pro každé $i = 1, \ldots, 5$ platí, že funkční hodnota B-splajnu $B_i^3(x)$ je v bodě interpolace t_i nenulová, tedy $B_i^3(t_i) \neq 0$ pro $\forall i = 1, \ldots, 5$. Můžeme tak tvrdit, že pro takto zadané uzly splajnu a body interpolace je kolokační matice $\boldsymbol{K}_k$ regulární.

Výstup funkce `IS1`:

```
> B
```

```
            [,1]
[1,]  2.44166667
[2,] -2.04166667
[3,]  3.64583333
[4,]  2.04166667
[5,] -0.04166667
```



Obrázek 13: Výstup `IS1` - interpolační splajn $s_2^*(x)$

Jak již bylo naznačeno, u tohoto příkladu, kdy uvažujeme krajní uzly (Δx) totožné s krajními body interpolace (Δt), si u funkce `IS2` nevystačíme při ověřování regularity kolokační matice pouze se splněním nerovnosti $y_i < t_i < y_{i+k}$, pro $i = 1, \ldots, 5$. Rozšířenou síť uzlů totiž nyní uvažujeme ve tvaru $(\Delta y) = \{0, 0, 0, 2, 7, 9, 9, 9\}$ a je tedy zřejmé, že dostaneme pro

$$\begin{aligned} \mathrm{i} = 1 \quad &: \qquad \mathrm{y}_1 = t_1 = 0 < 2 = y_2, \\ \mathrm{i} = 5 \quad &: \qquad \mathrm{y}_5 = 7 < 9 = t_5 = y_8. \end{aligned}$$

To je dáno tím, že volíme násobné rozšiřující uzly a zároveň, že

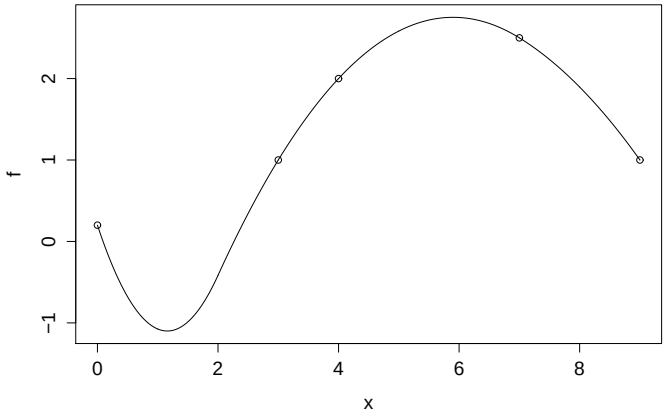
$$x_0 = t_1 \quad \text{a} \quad x_3 = t_5.$$

Pro $i = 1, 5$ tedy není splněna nerovnost $y_i < t_i < y_{i+k}$, ale i přesto je kolokační matice $\boldsymbol{K}_k$ regulární. V tomto případě je pro nás stěžejní to, že $B_i^3(t_i) \neq 0$, pro každé $i = 1, \ldots, 5$.

Výstup funkce `IS2`:

```
> B
```

```
             [,1]
[1,]   0.200000
[2,]  -2.041667
[3,]   3.645833
[4,]   2.041667
[5,]   1.000000
```



Obrázek 14: Výstup `IS2` - interpolační splajn $s_2^*(x)$

3. Metoda nejmenších čtverců

Nechť je dána rozšířená síť uzlů splajnu $(\Delta y) = \{y_1, \ldots, y_{N+k}\}$, pro kterou platí nerovnost $y_i < y_{i+k}$, pro každé $i = 1, \ldots, N$ a nechť k je přirozené číslo udávající řád splajnu. Tato síť uzlů je odvozena od původní sítě úzlů $(\Delta x) = \{x_0, \ldots, x_{n+1}\}$ jedním ze způsobů, které jsou popsané na str. 19 a 20. Dále nechť $(\Delta t) = \{t_1, \ldots, t_m\}$ je rostoucí posloupnost bodů a f_j označuje funkční hodnoty v těchto bodech, pro $j = 1 \ldots, m$. Pro tuto aproximační metodu je typické, že počet bodů aproximace (t_j, f_j) výrazně převyšuje počet uzlů splajnu.

Metoda nejmenších čtverců je jednou z nejznámějších aproximačních metod. Jejím cílem je nalézt takový splajn $s_{k-1}(x) \in S^k(\Delta y)$ pro který bude platit, že součet druhých mocnin chyb aproximace v daných bodech je minimální. Jinými slovy, že součet čtverců odchylek je co nejmenší. Tento splajn budeme hledat na intervalu $\langle a, b \rangle$, pro jehož krajní body platí, že

$$\begin{aligned} a &= x_0 = y_k, \\ b &= x_{n+1} = y_{N+1}. \end{aligned}$$

Cílem metody nejmenších čtverců je tedy najít splajn $s_{k-1}(x)$, který aproximuje body $(t_1, f_1), \ldots, (t_m, f_m)$ tím způsobem, že minimalizuje funkci

$$L(s_{k-1}) = \sum_{j=1}^{m} w_j \left[f_j - s_{k-1}(t_j) \right]^2,$$

kde $w_j \geq 0$ jsou váhové koeficienty, které umožňují „ocenit“ přesnost hodnoty f_j, pro $j = 1, \ldots, m$. Nemáme-li o hodnotách f_j žádné bližší informace, volíme váhy $w_j = 1$, pro $j = 1, \ldots, m$.

Nejprve si splajn $s_{k-1}(x)$ vyjádříme pomocí B-splajnových koeficientů

$$s_{k-1}(x) = \sum_{i=1}^{N} B_i^k(x) b_i = \boldsymbol{K}_k(x) \boldsymbol{b}.$$

Je tedy zřejmé, že splajn vyjádřený v bodech aproximace t_j, pro $j = 1, \ldots, m$, je ve tvaru

$$s_{k-1}(\boldsymbol{t}) = \boldsymbol{K}_k \boldsymbol{b},$$

kde $\boldsymbol{K}_k = (B_i^k(t_j))_{j=1,i=1}^{m\ N} \in \mathbb{R}^{m,N}$ je kolokační matice vyjádřená v bodech t_j. Dostaneme tak vektor funkčních hodnot splajnu v m bodech aproximace

$$s_{k-1}(\boldsymbol{t}) = (s_{k-1}(t_1), \ldots, s_{k-1}(t_m)).$$

Funkci $L(s_{k-1})$ si přepíšeme jako funkci proměnné $\boldsymbol{b}$, protože hledaný splajn je jednoznačně určený vektorem svých B-splajnových koeficientů. Funkce $\boldsymbol{L}(\boldsymbol{b})$ bude tedy ve tvaru

$$\boldsymbol{L}(\boldsymbol{b}) = (\boldsymbol{K}_k \boldsymbol{b} - \boldsymbol{f})^T \boldsymbol{W} (\boldsymbol{K}_k \boldsymbol{b} - \boldsymbol{f}),$$

tj. jedná se o kvadratickou funkci, kde $\boldsymbol{b}$ je vektor B-splajnových koeficientů, $\boldsymbol{f}$ je vektor zadaných funkčních hodnot a $\boldsymbol{W}$ představuje diagonální matici vah, tj. $\boldsymbol{W} = \operatorname{diag}\{w_j\}_{j=1}^m$. Roznásobením získáme následující podobu funkce

$$\boldsymbol{L}(\boldsymbol{b}) = \boldsymbol{b}^T \boldsymbol{K}_k^T \boldsymbol{W} \boldsymbol{K}_k \boldsymbol{b} - \boldsymbol{b}^T \boldsymbol{K}_k^T \boldsymbol{W} \boldsymbol{f} - \boldsymbol{f}^T \boldsymbol{W} \boldsymbol{K}_k \boldsymbol{b} + \boldsymbol{f}^T \boldsymbol{W} \boldsymbol{f}.$$

Máme tedy funkci proměnné $\boldsymbol{b}$ a naším cílem je nalézt takový vektor $\boldsymbol{b} = (b_1, \ldots, b_N)^T$ v němž funkce $\boldsymbol{L}(\boldsymbol{b})$ nabývá svého minima. Pro získání tohoto minima je nutné vypočítat následující parciální derivaci a položit ji rovnu nule

$$\frac{\partial \boldsymbol{L}(\boldsymbol{b})}{\partial \boldsymbol{b}^T} = 0.$$

Musíme si uvědomit, že řešením této soustavy je stacionární bod, který je bodem pouze podezřelým z extrému. Aby tento stacionární bod byl zároveň bodem globálního minima je nutné, aby funkce $\boldsymbol{L}(\boldsymbol{b})$ byla ryze konvexní. Kvadratická funkce $\boldsymbol{L}(\boldsymbol{b})$ je ryze konvexní právě tehdy, když je matice $\boldsymbol{K}_k^T \boldsymbol{W} \boldsymbol{K}_k$ pozitivně definitní, viz [5] str. 36. Vzhledem k rozměrům matic $\boldsymbol{W}$ a $\boldsymbol{K}_k$ víme, že matice $\boldsymbol{K}_k^T \boldsymbol{W} \boldsymbol{K}_k$ je vždy čtvercová. Navíc B-splajny jsou na svém nosiči kladné funkce

a mimo něj nulové. Je tedy zřejmé, že v matici $\boldsymbol{K}_k^T \boldsymbol{W}\boldsymbol{K}_k$ nebudou žádné záporné hodnoty a prvky na diagonále budou vždy kladné. Můžeme tedy o matici $\boldsymbol{K}_k^T \boldsymbol{W}\boldsymbol{K}_k$ říct, že je pozitivně definitní. V takovém případě je dle [5] zaručeno, že stacionární bod je zároveň hledaným bodem globálního minima a že toto minimum je jediné.

Vypočítáním a úpravou předešlého vztahu získáme následující rovnici

$$\boldsymbol{K}_k^T \boldsymbol{W}\boldsymbol{K}_k \boldsymbol{b} = \boldsymbol{K}_k^T \boldsymbol{W}\boldsymbol{f},$$

ve které matice $\boldsymbol{K}_k^T \boldsymbol{W}\boldsymbol{K}_k \in \mathbb{R}^{N,N}$ je vždy čtvercová.

Jednoznačnost řešení této soustavy rovnic a tedy také jednoznačné určení hledaného aproximačního splajnu je závislé na tom, zda je matice $\boldsymbol{K}_k^T \boldsymbol{W}\boldsymbol{K}_k$ regulární. Regularitu této matice ovlivňuje matice vah $\boldsymbol{W}$. Pokud bude existovat alespoň jedna váha $w_j = 0$, pro $j = 1, \ldots, m$, tak v takovém případě bude matice $\boldsymbol{W}$ singulární, což poruší regularitu celé matice $\boldsymbol{K}_k^T \boldsymbol{W}\boldsymbol{K}_k$. Kromě matice vah ovlivňuje regularitu matice $\boldsymbol{K}_k^T \boldsymbol{W}\boldsymbol{K}_k$ také kolokační matice $\boldsymbol{K}_k$, která musí být plné sloupcové hodnosti. To je zajištěno v případě, kdy existuje rostoucí podposloupnost bodů aproximace (Δu), tj. $(\Delta u) \subset (\Delta t)$, pro kterou platí nerovnost $y_i < u_i < y_{i+k}$, pro každé $i = 1, \ldots, N$ a zároveň počet bodů aproximace je větší než N. Při splnění těchto podmínek má matice $\boldsymbol{K}_k$ plnou sloupcovou hodnost, která je rovna právě N.

Za předpokladu, že funkce $\boldsymbol{L}(\boldsymbol{b})$ je ryze konvexní a matice $\boldsymbol{K}_k^T \boldsymbol{W}\boldsymbol{K}_k$ regulární, existuje právě jedno řešení ve tvaru

$$\boldsymbol{b}^* = (\boldsymbol{K}_k^T \boldsymbol{W}\boldsymbol{K}_k)^{-1} \boldsymbol{K}_k^T \boldsymbol{W}\boldsymbol{f},$$

kde $\boldsymbol{b}^* = (b_1^*, \ldots, b_N^*)^T$ je vektor B-splajnových koeficientů a existuje právě jeden splajn

$$s_{k-1}^*(x) = \sum_{i=1}^{N} B_i^k(x) b_i^*,$$

který je nejlepší aproximací hodnot (t_j, f_j), pro $j = 1, \ldots, m$, ve smyslu metody nejmenších čtverců.

Pro vykreslení splajnu $s^*_{k-1}(x)$ jsem v R naprogramovala dvě funkce - `MNC1` a `MNC2`, které se liší tím, jakým způsobem se vytváří rozšířená síť uzlů (Δy) z původní posloupnosti (Δx). Uživatel na vstupu obou funkcí zadá síť uzlů (Δx), posloupnost bodů aproximace (Δt), funkční hodnoty v těchto bodech, váhové koeficienty w_j a řád splajnu k. Výstupem je kromě grafu samotného splajnu $s^*_{k-1}(x)$ také vektor B-splajnových koeficientů, kterým je tento splajn jednoznačně určen. Podmínkou pro fungování těchto funkcí je načtení v R knihovny `splines`. Funkce pro vykreslení splajnu $s^*_{k-1}(x)$ vypadá následovně:

```
MNC2 = function(x,t,f,w,k){
# VSTUP: x = sit uzlu splajnu,
# t = body aproximace,
# f = funkcni hodnoty v bodech aproximace,
# w = vahove koeficienty,
# k = rad splajnu.
# VYSTUP: graf splajnu aproximujici zadane body ve smyslu MNC,
# vektor B-splajnovych koeficientu.
r = length(x)
# Overeni podminek
for (i in 1:length(t)){
if (t[i] < x[1] | t[i] > x[r]) stop ('body aproximace musi byt
z intervalu <x[1],x[r]>')}
for (i in 1:(length(t)-1)){
if (t[i] >= t[i+1]) stop ('t musi byt rostouci posloupnost')}
if (k != round(k)) stop ('rad splajnu k neni cele cislo')
if (k <= 0) stop ('rad splajnu musi byt kladne cislo')
for (i in 1:(r-1)){
```

```
if (x[i] >= x[i+1]) stop ('x musi byt rostouci posloupnost')}
if (length(t) != length(f)) stop ('t,f musi byt stejne delky')
if (length(t) != length(w)) stop ('t,w musi byt stejne delky')
if (length(x) >= length(t)) stop ('pocet bodu aproximace neni
vyssi nez pocet uzlu splajnu')
for (i in 1:length(w)){
if (w[i] <= 0) stop ('vahy w musi byt kladne')}
# Rozsirena sit uzlu
y = c()
if (k==1){
for (i in 1:r){y[i] = x[i]}}
if (k != 1){
Celkova_Delka = 2*(k-1) + r
y[1:(k-1)] = x[1]
y[k:(k+r-1)] = x[1:r]
y[(k+r):Celkova_Delka] = x[r]}
# Kolokacni matice K
K = splineDesign(y, t, k, outer.ok = TRUE)
# Diagonalni matice vah
W = diag(w)
# Kolokacni matice C - pro rozdeleny interval
deleni = seq(min(y), max(y), by = 0.001)
lambda = c(0:(r-1))
g = lambda[length(lambda) - 1]
# Dimenze prostoru splajnu
N = g+(k-1)+1
C = array(0, c(length(deleni),N))
l = c()
for(i in (1:N)){
for (j in 1:(k+1)){
```

```
l[j] = y[i+j-1]}
C[ ,i] = splineDesign(l, deleni, k, outer.ok = TRUE)}
C[1,1] = 1
C[length(deleni),N] = 1
# Overeni plne sloupcove hodnosti kolokacni matice
if (length(t) <= N) stop ('pocet bodu aproximace musi byt
vetsi nez dimenze prostoru splajnu')
if (qr(K)$rank != N) stop ('kolokacni matice nema plnou
sloupcovou hodnost')
# Vypocet B-splajnovych koeficientu
L = t(K)%*%W%*%K
P = t(K)%*%W%*%f
B = (solve(L,P))
# Vysledny splajn
S = C%*%B
matplot(deleni, S, type="l", xlab="x", ylab = "f",
ylim=c(min(B),max(B))
points(t,f)
B}
```

Poznámka 3.1. V kódu ověřujeme plnou sloupcovou hodnost matice $\boldsymbol{K}_k$ tak, že porovnáme hodnost této kolokační matice s dimenzí prostoru splajnů N. Pokud jsou si tyto dvě hodnoty rovny, pak má kolokační matice plnou sloupcovou hodnost. Pro účely kódu nám takové zjednodušené ověření podmínky 2, Věty 2.1 stačí. Kdybychom ale chtěli najít konkrétní podposloupnost $(\Delta u) = \{u_1, \ldots, u_N\} \subset (\Delta t)$, pro kterou platí nerovnost $y_i < u_i < y_{i+k}$, pro $i = 1, \ldots, N$, byla by naše práce o dost komplikovanější. V takovém případě bychom museli načíst knihovnu `combinat`, jejíž příkaz `combn` vytvoří z (Δt) všechny možné podposloupnosti o N prvcích, ze kterých potom vybíráme tu, která splňuje danou podmínku. Pro nalezení této podposloupnosti (Δu) jsem v R vytvořila následující cyklus. Pro jeho použití musíme znát počet bodů aproximace m, dimenzi pro-

storu splajnů N, rozšířenou síť uzlů (Δy) a posloupnost bodů (Δt). Uživateli R následně vrátí buď prvky podposloupnosti (Δu) nebo v případě, že taková podposloupnost neexistuje, se objeví hláška, že kolokační matice nemá plnou sloupcovou hodnost. Cyklus pro nalezení podposloupnost (Δu) vypadá následovně:

```
podminka=0
pocet=0
if (m>N){
# vytvori vsechny mozne kombinace
kombinace=combn(m,N)
# pocet kombinaci
pocet_sloupcu=ncol(kombinace)
j=1
while (podminka==0 & j<=pocet_sloupcu){
# cyklus, ktery najde N prvku podposloupnosti
for (i in 1:N){
if (y[i]<t[kombinace[i,j]] & t[kombinace[i,j]]<y[i+k]){
pocet=pocet+1 }}
if (pocet==N){
podminka=1
# podposloupnost u
u=c()
for (i in 1:N){
u[i]=t[kombinace[i,j]] }}
pocet=0
j=j+1 }}
if (podminka == 0) stop ('kolokacni matice nema plnou
sloupcovou hodnost')
if (podminka == 1) { print(u) }
```

Příklad 3.1. Najděte kvadratický splajn s uzly $(\Delta x) = \{0, 5, 10\}$, který ve

smyslu metody nejmenších čtverců aproximuje body $(\Delta t) = \{1, 2, 3, 6, 6.5, 7, 7.6, 8.1, 9\}$. Funkční hodnoty v bodech aproximace jsou $f = \{3.8, 1.5, 1.9, 1.6, 1, 1.9, 2.2, 1, 2.7\}$ a váhové koeficienty w_j, pro $j = 1, \ldots, 9$, uvažujme rovny jedné.

Řešení: Na zadání tohoto příkladu si ukážeme použití obou funkcí `MNC1` i `MNC2`, které jsem k tomuto účelu v R vytvořila. Vzhledem k nepřiměřené délce příkazů si je zde uvedeme pouze v symbolické podobě:

`MNC1(x,t,f,w,k)`,
`MNC2(x,t,f,w,k)`.

Výstupem je kromě výsledného splajnu také vektor B-splajnových koeficientů, kterým je tento splajn jednoznačně určený.

Nejprve se zaměříme na funkci `MNC1`, ve které se rozšířená síť uzlů (Δy) vytváří z původní sítě (Δx) přidáním $k-1$ ekvidistantních uzlů z obou stran. Je zřejmé, že rozšířená síť uzlů bude v následujícím tvaru:

```
> y
[1] -10 -5 0 5 10 15 20
```

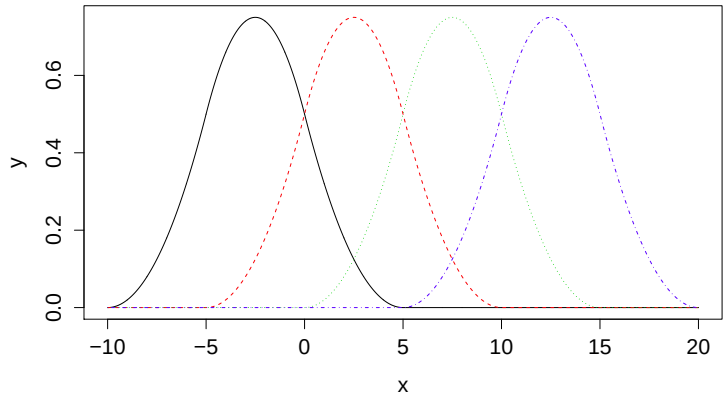
Nyní se podíváme na to, jak vypadá kolokační matice $\boldsymbol{K}_k$. Tento údaje sice není přímo výstupem funkce `MNC1`, nicméně je zajímavé vidět, z čeho při sestavování hledaného splajnu vycházíme.

```
> K
```

```
      [,1]   [,2]   [,3]   [,4]
[1,]  0.32  0.6600  0.0200  0.0000
[2,]  0.18  0.7400  0.0800  0.0000
[3,]  0.08  0.7400  0.1800  0.0000
[4,]  0.00  0.3200  0.6600  0.0200
[5,]  0.00  0.2450  0.7100  0.0450
[6,]  0.00  0.1800  0.7400  0.0800
[7,]  0.00  0.1152  0.7496  0.1352
[8,]  0.00  0.0722  0.7356  0.1922
[9,]  0.00  0.0200  0.6600  0.3200
```

Je vidět, že narozdíl od úlohy interpolace již $\boldsymbol{K}_k$ není čtvercová regulární. Požadujeme po ní ale, aby měla plnou sloupcovou hodnost. Pro zadané uzly a body aproximace platí, že $m = 9 > N = 4$. Zároveň je možné z (Δt) vybrat takovou podposloupnost (Δu), pro jejíž prvky bude platit nerovnost $y_i < u_i < y_{i+k}$, pro $i = 1, \ldots, 4$. Tato podposloupnost může být například ve tvaru $(\Delta u) = \{1, 3, 7, 8.1\} \subset (\Delta t)$. Pro tuto podposloupnost bodů aproximace platí, že i-tý bod (Δu) leží v nosiči B-splajnu $B_i^k(x)$. Můžeme tedy říct, že pro takto zvolené uzly a body aproximace je kolokační matice $\boldsymbol{K}_k$ plné sloupcové hodnosti, která je rovna N, tj. čtyřem.

Pro úplnost si ještě ukážeme, jak vypadají grafy B-splajnů $B_1^3(x), \ldots, B_4^3(x)$, jenž jsou definované na rozšířené síti uzlů (Δy):



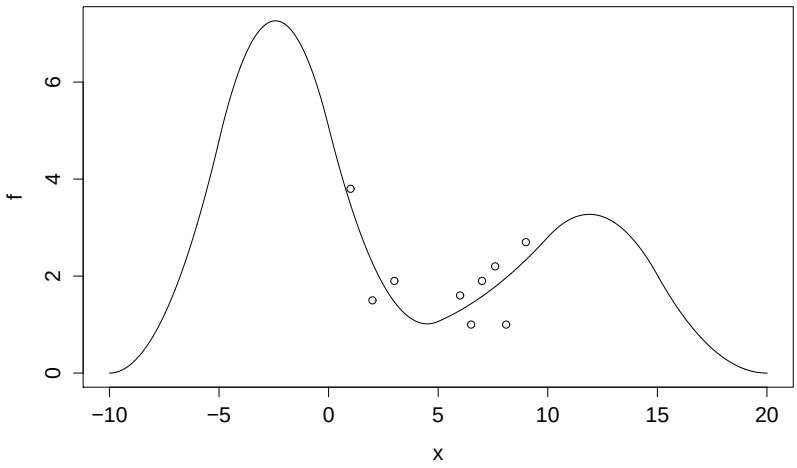
Obrázek 15: Grafy B-splajnů $B_1^3(x), \ldots, B_4^3(x)$

Vzhledem k tomu, že kolokační matice $\boldsymbol{K}_k$ má plnou sloupcovou hodnost a všechny váhy $w_j > 0$, pro $j = 1, \ldots, m$, je matice $\boldsymbol{K}_k^T \boldsymbol{W} \boldsymbol{K}_k$ regulární a existuje právě jedno řešení soustavy rovnic $\boldsymbol{b}^*$. Vektor B-splajnových koeficientů je ve tvaru:

```
> B
```

```
          [,1]
[1,]  9.5874270
[2,]  0.5681434
[3,]  1.5611537
[4,]  4.0411679
```

Tento vektor $\boldsymbol{b}^* = (b_1^*, \ldots, b_4^*)^T$ jednoznačně určuje výsledný splajn $s_2^*(x)$, viz Obr. 16, který je nejlepší aproximací zadaných hodnot ve smyslu metody nejmenších čtverců.



Obrázek 16: Výstup `MNC1` - splajn $s_2^*(x)$

Jak již bylo řečeno, ve funkci `MNC1` se rozšiřující uzly (Δy) volí ekvidistantně. My však v této práci uvažujeme ještě jeden způsob tvorby (Δy) a to přidáním násobných uzlů. V takovém případě použijeme pro vygenerování grafu $s_2^*(x)$ druhou z funkcí, kterou je `MNC2`. Rozšířená síť uzlů bude nyní ve tvaru:

```
> y
[1] 0 0 0 5 10 10 10
```

Také v tomto případě má kolokační matice $\boldsymbol{K}_k$ plnou sloupcovou hodnost, protože existuje podposloupnost (Δu) pro kterou platí nerovnost $y_i < u_i < y_{i+k}$, pro $\forall i = 1, \ldots, 4$. Tuto podmínku splňuje opět podposloupnost $(\Delta u) = \{1, 3, 7, 8.1\} \subset$

(Δt) a zároveň počet bodů aproximace je vyšší než N. Můžeme tedy říct, že pro zadané uzly a body aproximace je $\boldsymbol{K}_k$ plné sloupcové hodnosti.

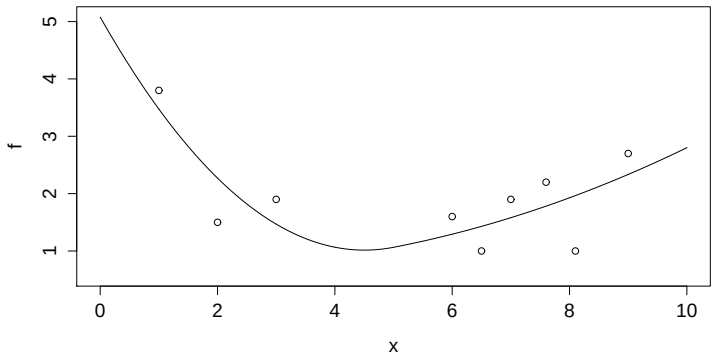
```
> K
```

```
      [,1]   [,2]   [,3]   [,4]
[1,]  0.64 0.3400 0.0200 0.0000
[2,]  0.36 0.5600 0.0800 0.0000
[3,]  0.16 0.6600 0.1800 0.0000
[4,]  0.00 0.3200 0.6400 0.0400
[5,]  0.00 0.2450 0.6650 0.0900
[6,]  0.00 0.1800 0.6600 0.1600
[7,]  0.00 0.1152 0.6144 0.2704
[8,]  0.00 0.0722 0.5434 0.3844
[9,]  0.00 0.0200 0.3400 0.6400
```

Vzhledem k tomu, že matice $\boldsymbol{K}_k$ je plné sloupcové hodnosti a všechny váhové koeficienty jsou kladné, je matice $\boldsymbol{K}_k^T \boldsymbol{W} \boldsymbol{K}_k$ regulární a funkce `MNC2` vygeneruje požadované výstupy. Výstupem funkce je graf splajnu $s_2^*(x)$ zobrazený na Obr. 17 a vektor B-splajnových koeficientů $\boldsymbol{b}^*$, který tento splajn jednoznačně určuje.

```
> B
```

```
          [,1]
[1,] 5.0777852
[2,] 0.5681434
[3,] 1.5611537
[4,] 2.8011608
```



Obrázek 17: Výstup `MNC2` - splajn $s_2^*(x)$

V obou případech, u funkce `MNC1` i `MNC2`, nám vyšla matice $\boldsymbol{K}_k^T \boldsymbol{W} \boldsymbol{K}_k$ regulární. Soustava rovnic má tedy právě jedno řešení, kterým je vektor B-splajnových koeficientů $\boldsymbol{b}^*$ a existuje právě jeden splajn, jenž aproximuje zadané hodnoty ve smyslu metody nejmenších čtverců. Z toho vyplývá, že pokud bychom se u výsledných grafů $s_2^*(x)$, Obr. 16 a 17, omezili pouze na interval $\langle 0, 10 \rangle$, tj. na síť uzlů (Δx), byly by oba grafy totožné.

4. Vyhlazující splajny

Vyhlazující (vyhlazovací) splajny jsou jakýmsi kompromisem mezi interpolačním splajnem a aproximací ve smyslu metody nejmenších čtverců.

Opět budeme uvažovat síť uzlů splajnu $(\Delta x) = \{x_0, \ldots, x_n, x_{n+1}\}$, od které je odvozena rozšířená síť uzlů $(\Delta y) = \{y_1, \ldots, y_{N+k}\}$. Přirozené číslo k udává řád splajnu. Dále nechť $(\Delta t) = \{t_1, \ldots, t_m\}$ je rostoucí posloupnost bodů a f_j označuje funkční hodnoty v těchto bodech, pro $j = 1, \ldots, m$. Pro tuto metodu je typické, že počet bodů aproximace (t_j, f_j) výrazně převyšuje dimenzi prostoru splajnů, tj. $m > N$. I nadále budeme splajn $s_{k-1}(x)$ uvažovat na intervalu $\langle a, b\rangle$, pro jehož krajní body platí, že

$$\begin{aligned} a &= x_0 = y_k, \\ b &= x_{n+1} = y_{N+1}. \end{aligned}$$

Naším cílem je najít **vyhlazující splajn** $s_{k-1}(x) \in S^k(\Delta y)$, který bude na tomto intervalu $\langle a, b\rangle$ minimalizovat funkci

$$I_l(s_{k-1}) = J_l(s_{k-1}) + L(s_{k-1}),$$

kde

$$J_l(s_{k-1}) = \int_a^b \left[s_{k-1}^{(l)}(x)\right]^2 dx, \qquad l \in \{1, \ldots, k-2\}$$

a $s_{k-1}^{(l)}(x)$ představuje derivace l-tého řádu splajnu $s_{k-1}(x)$ v bodě x. Pro zopakování si zde připomeňme, jak vypadá vztah pro výpočet l-té derivace splajnů:

$$\frac{d^{(l)}}{dx^{(l)}} \sum_{i=1}^{N} b_i B_i^k(x) = \sum_{i=1+l}^{N} b_i^{(l)} B_i^{k-l}(x).$$

Jak již víme

$$L(s_{k-1}) = \sum_{j=0}^{m} w_j \left[f_j - s_{k-1}(t_j)\right]^2.$$

Tento člen $L(s_{k-1})$ jsme uvažovali také v předchozí kapitole, věnované metodě nejmenších čtverců a všechny jeho vlastnosti zůstávají stejné i pro účely vyhlazování.

Nyní si vyjádříme $I_l(s_{k-1})$ maticově, jako funkci B-splajnových koeficientů. Z předchozí kapitoly již víme, že přepsáním $L(s_{k-1})$ získáme funkci ve tvaru

$$\boldsymbol{L}(\boldsymbol{b}) = (\boldsymbol{K}_k\boldsymbol{b} - \boldsymbol{f})^T \boldsymbol{W}(\boldsymbol{K}_k\boldsymbol{b} - \boldsymbol{f})$$

a $J_l(s_{k-1})$ si dle [3] upravíme tímto způsobem

$$\boldsymbol{J}_l(\boldsymbol{b}) = (\boldsymbol{b}^{(l)})^T \boldsymbol{N}\boldsymbol{b}^{(l)} = \boldsymbol{b}^T \boldsymbol{S}_l^T \boldsymbol{M}\boldsymbol{S}_l\boldsymbol{b},$$

kde $\boldsymbol{b}$ je vektor B-splajnových koeficientů jednoznačně určující splajn

$$s_{k-1}(x) = \boldsymbol{K}_k(x)\boldsymbol{b}.$$

Ještě nám zbývá vyjádřit si matici $\boldsymbol{N} = \boldsymbol{S}_l^T \boldsymbol{M}\boldsymbol{S}_l$, přičemž

$$\boldsymbol{M} = \begin{pmatrix} (\mathrm{B}_{1+l}^{k-l}, B_{1+l}^{k-l}) & \dots & (\mathrm{B}_N^{k-l}, B_{1+l}^{k-l}) \\ \vdots & & \vdots \\ (\mathrm{B}_{1+l}^{k-l}, B_N^{k-l}) & \dots & (\mathrm{B}_N^{k-l}, B_N^{k-l}) \end{pmatrix},$$

kde $l \in \{1, \dots, k-2\}$ označuje derivaci l-tého řádu. Matice $\boldsymbol{M} \in \mathbb{R}^{N,N}$ je tedy vždy čtvercová. Jednotlivé prvky této matice jsou definovány jako integrály ze součinu B-splajnů na intervalu $\langle a, b\rangle$, tj.

$$(B_i^{k-l}, B_j^{k-l}) = \int_a^b B_i^{k-l}(x) B_j^{k-l}(x)\, dx.$$

Dle [3] je $\boldsymbol{M}$ pozitivně definitní, neboť bázové funkce jsou na svém nosiči kladné a mimo něj nulové. Navíc z vlastností B-splajnů vyplývá, že diagonální prvky této matice budou vždy větší než prvky nediagonální. Je zřejmé, že diagonální prvky (B_i^{k-l}, B_j^{k-l}), kde $i = j$, se počítají jako skalární součin B-splajnu s tím samým B-splajnem. Dostaneme tedy vždy vyšší hodnotu, než kdybychom skalární součin

počítali mezi dvěma různými B-splajny. Nediagonální prvky matice $\boldsymbol{M}$ jsou tedy nižší, než prvky diagonální. Zároveň pro takto definovanou matici $\boldsymbol{M}$ platí, že

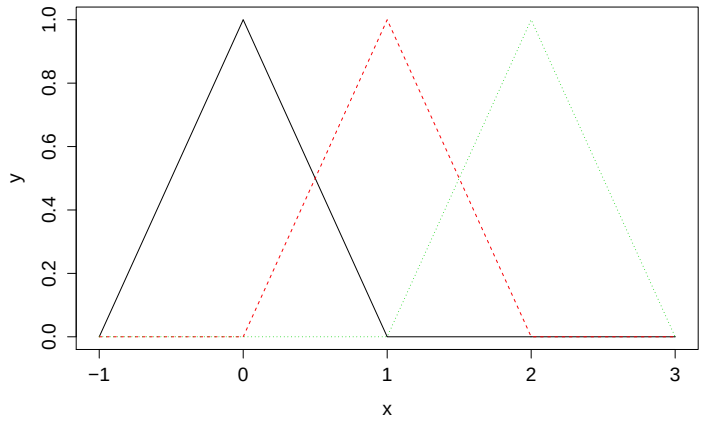
$$(B_i^{k-l}, B_j^{k-l}) = 0, \quad \text{pro} \quad (i-j) \geq k,$$

což je dáno opět tím, že nosič B-splajnu je omezený a mimo svůj nosič je daný B-splajn nulový. Takto tedy získáme tzv. pásovou matici $\boldsymbol{M}$. S rostoucím řádem derivace se šířka tohoto pásu snižuje. Pro nejvyšší možný řád derivace $l = k-2$, dostaneme pás o šířce $(k-l)+1$. Následně se snižujícím l se šířka pásu zvyšuje, tedy např. pro řád derivace $l = k-2-1 = k-3$ je šírka pásu rovna hodnotě $(k-l)+1+1 = (k-l)+2$, apod. Je zřejmé, že maximální šířka pásu je rovna N, tj. počtu řádků v matici $\boldsymbol{M}$. Zároveň platí, že tato matice nebude nikdy nulová. Prvky této matice totiž počítáme jako integrál ze dvou funkcí, které jsou na daném intervalu kladné a mimo něj nulové.

Poznámka 4.1. Z důvodu zjednodušení výpočtu prvků matice $\boldsymbol{M}$, použijeme pro jejich výpočet numerické integrování. Jednotlivé integrály vyčíslíme prostřednictvím jedné z nejužívanějších kvadraturních formulí, pomocí tzv. **složeného lichoběžníkového pravidla**. K tomuto účelu jsem v R vytvořila následující funkci:

```
SLP=function(krok,c) {
integral=krok*(0.5*c[1]+sum(c[2:(length(c)-1)])+0.5*c[length(c)])
return(integral) }
```

Parametr `krok` udává posun po ose x. Druhý parametr `c` představuje vektor funkčních hodnot B-splajnu $B_i^k(x)$ v námi zvolených bodech. Výstupem `SLP` je výsledná hodnota integrálu. Uvažujme pro jednoduchost lineární B-splajn, definovaný na síti uzlů $(\Delta x) = \{0, 1, 2\}$. Aby bylo možné sestavit kompletní bázi prostoru splajnů, musíme od sítě (Δx) přejít k rozšířené síti uzlů, kterou budeme uvažovat například ve tvaru $(\Delta y) = \{-1, 0, 1, 2, 3\}$.



Obrázek 18: Bázové funkce $B_1^2(x), \ldots, B_3^2(x)$

Již ze samotného grafu bázových funkcí je patrné, že obsah plochy pod křivkou každého B-splajnu je roven jedné. Pro kontrolu použijeme funkci `SLP`, pomocí které spočítáme obsah plochy pod křivkou prvního B-splajnu $B_1^2(x)$. Vstup v R bude ve tvaru

```
> SLP(0.2,c),
```

kde `0.2` udává, o jak velký krok se posunujeme po ose x a `c` je sloupec funkčních hodnot z kolokační matice, který odpovídá funkčním hodnotám B-splajnu $B_1^2(x)$.

```
> x = c(-1,0,1)
> t = seq(min(x),max(x),by=0.2)
> c = splineDesign(x,t,2,outer.ok=TRUE)
> c
```

```
      [,1]
[1,]   0.0
[2,]   0.2
[3,]   0.4
[4,]   0.6
[5,]   0.8
[6,]   1.0
[7,]   0.8
[8,]   0.6
[9,]   0.4
[10,]  0.2
[11,]  0.0
```

Jako výstup funkce `SLP` obdržíme hodnotu

```
> [1] 1,
```

která odpovídá našemu předpokladu, že obsah plochy pod křivkou B-splajnu $B_1^2(x)$ je roven jedné.

Je vidět, že pro výpočet obsahu plochy jsme uvažovali pouze interval $\langle -1, 1\rangle$, tj. pouze tu část intervalu $\langle -1, 3\rangle$, na které je nosič B-splajnu $B_1^2(x)$ nenulový. Je to z toho důvodu, aby se při velkém množství dat zbytečně neprodlužoval výpočet hodnoty integrálu.

Ještě se podíváme na to, jak použijeme funkci `SLP` pro výpočet prvků matice $\boldsymbol{M}$:

```
> M[i,j] = SLP(0.1,soucin)
```

Do funkce `SLP` dosazujeme hodnotu `0.1`, která udává velikost posunu po ose x a `soucin`, který představuje vektor hodnot, z nichž se počítá hodnota integrálu. Tento vektor vznikne vynásobením i-tého a j-tého sloupce kolokační matice $\boldsymbol{K}_k$ a navíc z něj vybereme pouze ty prvky, které jsou nenulové. Jak již bylo řečeno, je to z toho důvodu, aby se výpočet hodnoty integrálu zbytečně neprodlužoval. Takto vypočítaná hodnota se následně dosadí do matice $\boldsymbol{M}$ na pozici prvku $\boldsymbol{m}_{ij}$, pro každé $i, j = 1, \ldots, N$.

Nyní již víme, jak vytvořit matici $\boldsymbol{M}$. Pro zopakování si ještě připomeňme, jak vypadá matice $\boldsymbol{S}_l$, kterou jsme si zavedli v části věnující se derivování splajnů.

Jedná se tedy o horní trojúhelníkovou matici plné řádkové hodnosti, která je dána vztahem

$$\boldsymbol{S}_0 = \boldsymbol{I}_N,$$

$$\boldsymbol{S}_j = \boldsymbol{D}_j \boldsymbol{L}_j \dots \boldsymbol{D}_1 \boldsymbol{L}_1 \in \mathbb{R}^{N-j,N},$$

kde $\boldsymbol{I}_N \in \mathbb{R}^{N,N}$ je jednotková matice,

$$\boldsymbol{D}_l = (k-l)\,\mathrm{diag}\left\{\frac{1}{y_{i+k-l} - y_i}\right\}_{i=1+l}^{N} \in \mathbb{R}^{N-l,N-l},$$

a

$$\boldsymbol{L}_l = \begin{pmatrix} -1 & 1 & & \\ & \ddots & \ddots & \\ & & -1 & 1 \end{pmatrix} \in \mathbb{R}^{N-l,N+1-l}.$$

Z vlastností matic $\boldsymbol{S}_l$ a $\boldsymbol{M}$ vyplývá, že výsledná matice $\boldsymbol{N} = \boldsymbol{S}_l^T \boldsymbol{M} \boldsymbol{S}_l \in \mathbb{R}^{N,N}$ je symetrická s hodností $N - l$. Jedná se tedy o singulární matici.

Nyní již známe vše potřebné k tomu, abychom si mohli maticově vyjádřit funkci $I_l(s_{k-1})$, jako funkci B-splajnových koeficientů. Platí tedy, že

$$\boldsymbol{I}_l(\boldsymbol{b}) = \boldsymbol{b}^T \boldsymbol{N} \boldsymbol{b} + \left[\boldsymbol{f} - \boldsymbol{K}_k \boldsymbol{b}\right]^T \boldsymbol{W} \left[\boldsymbol{f} - \boldsymbol{K}_k \boldsymbol{b}\right],$$

tj. jedná se o kvadratickou funkci, kde $\boldsymbol{b}$ je vektor B-splajnových koeficientů a $\boldsymbol{f}$ je vektor zadaných funkčních hodnot. Matice $\boldsymbol{K}_k$ představuje kolokační matici v bodech t_j, $\boldsymbol{N} = \boldsymbol{S}_l^T \boldsymbol{M} \boldsymbol{S}_l$ a $\boldsymbol{W}$ je diagonální matice vah $w_j \geq 0$, tj. $\boldsymbol{W} = \mathrm{diag}\,\{w_j\}_{j=1}^m$.

Roznásobením $\boldsymbol{I}_l(\boldsymbol{b})$ získáme následující podobu funkce

$$\boldsymbol{I}_l(\boldsymbol{b}) = \boldsymbol{b}^T \boldsymbol{N} \boldsymbol{b} + \boldsymbol{b}^T \boldsymbol{K}_k^T \boldsymbol{W} \boldsymbol{K}_k \boldsymbol{b} - \boldsymbol{b}^T \boldsymbol{K}_k^T \boldsymbol{W} \boldsymbol{f} - \boldsymbol{f}^T \boldsymbol{W} \boldsymbol{K}_k \boldsymbol{b} + \boldsymbol{f}^T \boldsymbol{W} \boldsymbol{f}.$$

Máme tedy funkci proměnné $\boldsymbol{b}$ a naším cílem je nalézt takový vektor $\boldsymbol{b} = (b_1, \dots, b_N)^T$, v němž funkce $\boldsymbol{I}_l(\boldsymbol{b})$ nabývá svého minima. Musíme tedy vypočítat parciální derivaci dle $\boldsymbol{b}^T$ a položit ji rovnu nule

$$\frac{\partial \boldsymbol{I}_l(\boldsymbol{b})}{\partial \boldsymbol{b}^T} = 0.$$

Je nutné si uvědomit, že položíme-li vypočítanou parciální derivaci rovnu nule, získáme tím rovnici, jejímž řešením je pouze stacionární bod. Stacionární bod obecně není bodem minima, je pouze bodem podezřelým z extrému. Aby tento stacionární bod byl zároveň bodem globálního minima je nutné, aby funkce $\boldsymbol{I}_l(\boldsymbol{b})$ byla ryze konvexní.

Vypočítáním a úpravou předchozího vztahu získáme následující rovnici

$$\left[\boldsymbol{N} + \boldsymbol{K}_k^T \boldsymbol{W} \boldsymbol{K}_k\right] \boldsymbol{b} = \boldsymbol{K}_k^T \boldsymbol{W} \boldsymbol{f}.$$

Dle [5] str. 36, je kvadratická funkce $\boldsymbol{I}_l(\boldsymbol{b})$ ryze konvexní právě tehdy, když je matice $\left[\boldsymbol{N} + \boldsymbol{K}_k^T \boldsymbol{W} \boldsymbol{K}_k\right]$ pozitivně definitní. Jak již z kapitoly věnované metodě nejmenších čtverců víme, matice $\boldsymbol{K}_k^T \boldsymbol{W} \boldsymbol{K}_k \in \mathbb{R}^{N,N}$. Matice $\boldsymbol{N}$ má shodné rozměry jako předchozí matice, tedy i zde platí, že $\boldsymbol{N} \in \mathbb{R}^{N,N}$. Sčítáme tak dvě čtvercové matice stejného rozměru. Proto i výsledná matice $\left[\boldsymbol{N} + \boldsymbol{K}_k^T \boldsymbol{W} \boldsymbol{K}_k\right] \in \mathbb{R}^{N,N}$ je vždy čtvercová. Zároveň platí, že rozměry této výsledné matice nejsou nijak ovlivněné tím, jakou hodnotu derivace si zvolíme pro výpočet matice $\boldsymbol{N}$. Navíc z vlastností B-splajnů plyne, že v matici $\left[\boldsymbol{N} + \boldsymbol{K}_k^T \boldsymbol{W} \boldsymbol{K}_k\right]$ nemohou být záporné hodnoty a že její diagonála bude vždy kladná. Můžeme o ní tedy říct, že je pozitivně definitní. V takovém případě je dle [5] zaručeno, že stacionární bod je zároveň bodem globálního minima a že toto minimum je jediné.

Jednoznačnost řešení této soustavy rovnic a tedy také jednoznačné určení hledaného vyhlazujícího splajnu je závislé na tom, zda je matice $\left[\boldsymbol{N} + \boldsymbol{K}_k^T \boldsymbol{W} \boldsymbol{K}_k\right]$ regulární. Dle [3] je tato matice regulární právě tehdy, když je alespoň jedna z matic $\boldsymbol{N}$ a $\boldsymbol{K}_k^T \boldsymbol{W} \boldsymbol{K}_k$ regulární. Matice $\boldsymbol{N}$ regulární není, protože $\boldsymbol{N} \in \mathbb{R}^{N,N}$, ale její hodnost je pouze $N - l$. Jedná se tedy o singulární matici. Matice $\boldsymbol{K}_k^T \boldsymbol{W} \boldsymbol{K}_k$ je regulární tehdy, splňuje-li podmínky regularity popsané v rámci předchozí kapitoly, viz str. 40. Platí, že pokud k regulární matici $\boldsymbol{K}_k^T \boldsymbol{W} \boldsymbol{K}_k$ přičteme singulární matici $\boldsymbol{N}$, zůstane ve výsledné matici $\left[\boldsymbol{N} + \boldsymbol{K}_k^T \boldsymbol{W} \boldsymbol{K}_k\right]$ regularita zachována.

Za předpokladu, že funkce $\boldsymbol{I}_l(\boldsymbol{b})$ je ryze konvexní a matice $\left[\boldsymbol{N} + \boldsymbol{K}_k^T \boldsymbol{W} \boldsymbol{K}_k\right]$

regulární, má soustava rovnic právě jedno řešení ve tvaru

$$\boldsymbol{b}^* = \left[\boldsymbol{N} + \boldsymbol{K}_k^T \boldsymbol{W} \boldsymbol{K}_k\right]^{-1} \boldsymbol{K}_k^T \boldsymbol{W} \boldsymbol{f},$$

kde $\boldsymbol{b}^* = (b_1^*, \ldots, b_N^*)^T$ je vektor B-splajnových koeficientů, který jednoznačně určuje hledaný vyhlazující splajn

$$s_{k-1}^*(x) = \sum_{i=1}^{N} B_i^k(x) b_i^*.$$

Pro vykreslení vyhlazujícího splajnu $s_{k-1}^*(x)$ jsem v R vytvořila dvě funkce `VS1` a `VS2`. Rozdíl mezi nimi je v tom, jakým způsobem se vytváří rozšířená síť uzlů (Δy) z původní sítě (Δx). Pro první funkci `VS1` platí, že (Δy) se vytvoří z původní sítě přidáním $k-1$ ekvidistantních uzlů z obou stran (Δx). V případě funkce `VS2` jsou tyto rozšiřující uzly zvoleny jako násobné. Na vstupu obou funkcí uživatel zadává síť uzlů splajnu (Δx), rostoucí posloupnost bodů aproximace (Δt), funkční hodnoty v těchto bodech a váhové koeficienty w_j, pro $j = 1, \ldots, m$, kde m je počet bodů aproximace. Dále uživatel zadává řád splajnu k a derivaci *der*. Stejně jako u ostatních funkcí, tak také zde je zapotřebí načíst knihovnu `splines`, jejíž příkaz `splineDesign` používáme pro vytvoření kolokační matice $\boldsymbol{K}_k$. Výstupem funkcí je graf hledaného vyhlazujícího splajnu $s_{k-1}^*(x)$ a také vektor B-splajnových koeficientů, kterým je tento splajn jednoznačně určen. Výsledná funkce vypadá následovně:

```
VS1 = function(x,t,f,w,k,der){
# VSTUP: x = sit uzlu splajnu,
# t = body aproximace,
# f = funkcni hodnoty v bodech aproximace,
# w = vahove koeficienty,
# k = rad splajnu,
# der = derivace.
```

```
# VYSTUP: graf splajnu aproximujici zadane body ve smyslu MNC,
# vektor B-splajnovych koeficientu.
r = length(x)
# Overeni podminek
for (i in 1:length(t)){
if (t[i] < x[1] | t[i] > x[r]) stop ('body aproximace musi byt
z intervalu <x[1],x[r]>')}
for (i in 1:(length(t)-1)){
if (t[i] >= t[i+1]) stop ('t musi byt rostouci posloupnost')}
if (k != round(k)) stop ('rad splajnu k neni cele cislo')
if (k <= 0) stop ('rad splajnu musi byt kladne cislo')
for (i in 1:(r-1)){
if (x[i] >= x[i+1]) stop ('x musi byt rostouci posloupnost')}
if (length(t) != length(f)) stop ('t,f musi byt stejne delky')
if (length(t) != length(w)) stop ('t,w musi byt stejne delky')
if (length(x) >= length(t)) stop ('pocet bodu aproximace neni
vyssi nez pocet uzlu splajnu')
for (i in 1:length(w)){
if (w[i] <= 0) stop ('vahy w musi byt kladne')}
if (der >= (k-1)) stop ('derivace musi byt mnoziny {1,...,k-2}')
# Rozsirena sit uzlu
Celkova_Delka = 2*(k-1) + r
krok = x[r]-x[r-1]
y = c()
for (i in 1:Celkova_Delka){
if (i < (k-1)){y[i] = x[1]-krok*(k-i)}
if (i == (k-1)){y[i] = x[1]-krok}
if ((i > (k-1)) & (i <= (r+(k-1)))){y[i] = x[i-(k-1)]}
if (i > (r+(k-1))){y[i] = x[r]+krok*(i-r-(k-1))}}
# Kolokacni matice K
```

```
K = splineDesign(y, t, k, outer.ok = TRUE)
# Diagonalni matice vah
W = diag(w)
# Kolokacni matice C - pro rozdeleny interval
deleni = seq(min(y), max(y), by = 0.001)
lambda = c(0:(r-1))
g = lambda[length(lambda) - 1]
# Dimenze prostoru splajnu
N = g+(k-1)+1
C = array(0, c(length(deleni),N))
l = c()
for(i in (1:N)){
for (j in 1:(k+1)){
l[j] = y[i+j-1]}
C[ ,i] = splineDesign(l, deleni, k, outer.ok = TRUE)}
# Overeni plne sloupcove hodnosti kolokační matice K
if (length(t) <= N) stop ('pocet bodu aproximace musi byt
vetsi nez dimenze prostoru splajnu')
if (qr(K)$rank != N) stop ('kolokacni matice nema plnou
sloupcovou hodnost')
# Matice S
S = array(0)
if (der == 0){
S = diag(1, c(N,N))}
if (der > 0){
i=der
while (i>0){
D = array(0)
rozdil = y[(1+k):(N+k-i)] - y[(1+i):(N)]
D = (k-i)*diag(1/rozdil)
```

```
L = array(0, c(N-i,N-i+1))
for (j in (1:(N-i))){
L[j,j] = (-1)
L[j,j+1] = 1}
if (i==der){
S = D%*%L
}else{
S = S%*%D%*%L}
i=i-1}}
# Matice M - snizeni radu splajnu na k-der
kk = k-der
# Matice M - rozsirena sit uzlu
celkova_delka = 2*(kk-1) + r
Y = c()
for (i in 1:celkova_delka){
if (i < (kk-1)){Y[i] = x[1]-krok*(kk-1)-1+i}
if (i == (kk-1)){Y[i] = x[1]-krok}
if ((i > (kk-1)) & (i <= (r+(kk-1)))){Y[i] = x[i-(kk-1)]}
if (i > (r+(kk-1))){Y[i] = x[r]+krok*(i-r-(kk-1))}}
Deleni = seq(min(Y), max(Y), by = 0.1)
Lambda = c(0:(r-1))
G = Lambda[length(Lambda) - 1]
# Matice M - dimenze prostoru splajnu
NN = G+(kk-1)+1
# Matice M - kolokacni matice KK
KK = splineDesign(Y, Deleni, kk, outer.ok=TRUE)
# Matice M - funkce pro Vypocet integralu
SLP=function(krok, c){
integral = krok*(0.5*c[1]+sum(c[2:(length(c)-1)])+0.5*c[length(c)])
return (integral)}
```

```
#Matice M
M=array(0, c(NN,NN))
for (i in 1:NN){
for (j in 1:NN){
nenulove = c()
soucin = KK[,i]*KK[,j]
for (m in 1:length(Deleni)){
if (soucin[m] != 0) {nenulove[m] = soucin[m]}}
M[i,j]=SLP(0.1, soucin)}}
# Matice N
N_matice = t(S)%*%M%*%S
# Vypocet B-splajnovych koeficientu
L = N_matice + t(K)%*%W%*%K
P = t(K)%*%W%*%f
B = (solve(L,P))
# Vysledny splajn
SS = C%*%B
matplot(deleni, SS, type="l", xlab="x", ylab = "f",xlim=c(min(x),
max(x)),ylim=c(min(f)-1,max(f)+1))
points(t,f)
B}
```

Z kódu je patrné, že narozdíl od předešlých dvou metod aproximace budeme nyní výsledný splajn $s^*_{k-1}(x)$ vykreslovat pouze na intervalu $\langle a, b\rangle$, kde $a = x_0 < \ldots < x_n < x_{n+1} = b$. Je to z toho důvodu, že hodnotu funkce $J_l(s_{k-1})$ minimalizujeme právě na tomto intervalu. Nemělo by tedy význam vykreslovat výsledný splajn na celé rozšířené síti uzlů (Δy).

Příklad 4.1. Najděte kubický vyhlazující splajn s uzly $(\Delta x) = \{-5, 15, 35\}$ a s body aproximace $(\Delta t) = \{-3, -1, 0, 5, 7, 12, 18, 20, 22, 26, 29, 30, 31, 33, 34\}$. Funkční hodnoty v těchto bodech jsou $f = \{3.2, 2.8, 4.7, 7, 5.3, 5.4, 3.4, 4, 2.3, 1.1, 1.9,$

$2, 4.5, 5, 4\}$, váhové koeficienty w_j, pro $j = 1, \ldots, 15$ uvažujte rovny jedné a derivaci zvolte druhého řádu.

Řešení: Vyhlazující splajn sestavíme s použitím obou funkcí `VS1` i `VS2`, které jsem k tomuto účelu v softwaru R vytvořila. Vzhledem k tomu, že zadávané příkazy jsou nepřiměřeně dlouhé, uvedeme zde pouze jejich symbolickou podobu:

```
VS1(x,t,f,w,k,der),
VS2(x,t,f,w,k,der).
```

Výstupem těchto funkcí je graf hledaného vyhlazujícího splajnu a vektor B-splajnových koeficientů, kterým je tento vyhlazující splajn jednoznačně určen.

Začneme první funkcí `VS1`, ve které se rozšířená sít vytváří z (Δx) přidáním $k-1$ ekvidistantních uzlů. Nová síť uzlů (Δy) bude tedy vypadat následovně:

```
> y
[1] -65 -45 -25 -5 15 35 55 75 95
```

Nejprve se podíváme na to, jestli je matice $[\boldsymbol{N} + \boldsymbol{K}_k^T \boldsymbol{W} \boldsymbol{K}_k]$ regulární. To bude splněno právě tehdy, když alespoň jedna z matic $\boldsymbol{N}$ a $\boldsymbol{K}_k^T \boldsymbol{W} \boldsymbol{K}_k$ bude regulární. Vzhledem k tomu, že matice $\boldsymbol{N}$ je vždy singulární, požadujeme splnění podmínky regularity po matici $\boldsymbol{K}_k^T \boldsymbol{W} \boldsymbol{K}_k$. Tato matice je regulární v případě, kdy kolokační matice má plnou sloupcovou hodnost. V praxi to znamená, že musí platit nerovnost $m > N$ a současně existovat taková podposloupnost $(\Delta u) \subset (\Delta t)$, pro kterou platí nerovnost $y_i < u_i < y_{i+k}$, pro $\forall i = 1, \ldots, N$.

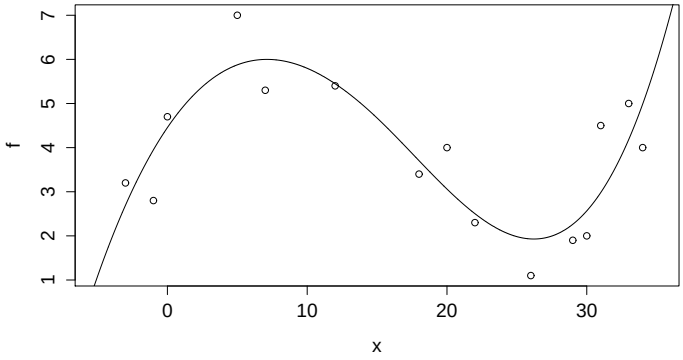
V našem případě je počet bodů aproximace $m = 15$ a dimenze prostoru splajnů $N = 1 + (4 - 1) + 1 = 5$. Je tedy evidentní, že platí nerovnost $m > N$. Co se podposloupnosti (Δu) týče, tak těch můžeme nalézt rovnou několik. Jednou z nich je např. podposloupnost $(\Delta u) = \{-3, -1, 0, 5, 18\}$, pro jejíž prvky platí nerovnost $y_i < u_i < y_{i+k}$, pro $\forall i = 1, \ldots, 5$. Můžeme tedy říct, že pro takto zvolené uzly splajnu a body aproximace je kolokační matice $\boldsymbol{K}_k$ regulární. Vzhledem k tomu, že nemáme zadané žádné $w_j = 0$, pro $j = 1, \ldots, 15$, je matice $\boldsymbol{K}_k^T \boldsymbol{W} \boldsymbol{K}_k$ regulární.

Z toho plyne, že je regulární také matice $[\boldsymbol{N} + \boldsymbol{K}_k^T \boldsymbol{W} \boldsymbol{K}_k]$ a funkce `VS1` vygeneruje příslušné výstupy.

```
> B
```

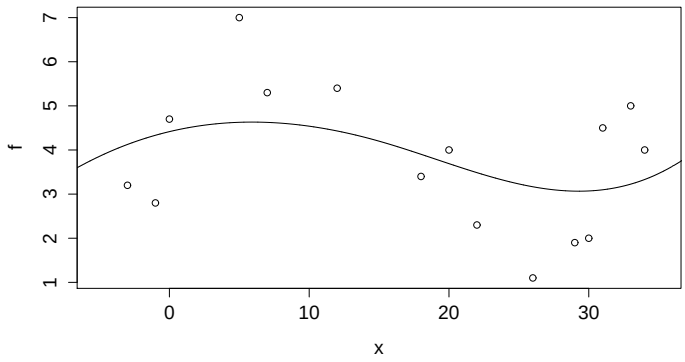
```
             [,1]
[1,] -29.730421
[2,]   7.488886
[3,]   6.262613
[4,]  -4.511588
[5,]  47.420308
```

Obr. 19 odpovídá grafu vyhlazujícího splajnu $s_3^*(x)$, který je vykreslený na síti uzlů (Δx).



Obrázek 19: Výstup `VS1` - vyhlazující splajn $s_3^*(x)$

Poznámka 4.2. Pro zajímavost se podíváme na to, jak se změní graf hledaného vyhlazujícího splajnu, budeme-li uvažovat derivaci pouze prvního řádu. Místo `der = 2` tedy zadáme na vstupu funkce hodnotu `der = 1`.



Obrázek 20: Výstup `VS1` - vyhlazující splajn $s_3^*(x)$

Porovnáme-li Obr. 19 a 20 je zřejmé, že změna řádu derivace ovlivnila graf vyhlazujícího splajnu. Matice $\boldsymbol{N}$, pro jejíž výpočet se hodnota derivace používá, totiž jasně stanovuje, jak bude graf výsledného splajnu vypadat. Volbou řádu derivace tedy ovlivňujeme to, jestli chceme minimalizovat například strmost splajnu či konvexitu konkavit.

Vraťme se zpět k našemu Příkladu 4.1., ve kterém uvažujeme derivaci druhého řádu. Nyní nás bude zajímat výstup funkce `VS2`. V ní je rozšířená síť uzlů (Δy) tvořena přidáním $k-1$ násobných uzlů z obou stran původní sítě (Δx). Rozšířenou síť tedy uvažujeme ve tvaru:

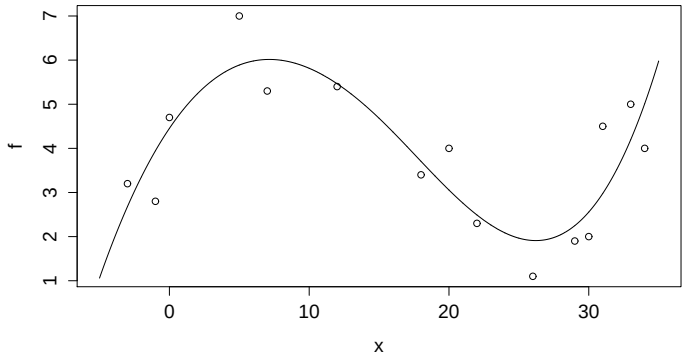
```
> y
[1] -5 -5 -5 -5 15 35 35 35 35
```

I nadále platí, že počet bodů aproximace je vyšší než dimenze prostoru splajnů, tj. $15 > 5$. Podposloupnost bodů aproximace můžeme nadále uvažovat ve stejném tvaru $(\Delta u) = \{-3, -1, 0, 5, 18\}$, protože i zde platí nerovnost $y_i < u_i < y_{i+k}$, pro každé $i = 1, \ldots, 5$. Pro takto zvolené uzly splajnu a body aproximace je tedy kolokační matice $\boldsymbol{K}_k$ plné sloupcové hodnosti. Vzhledem k tomu, že všechny váhové koeficienty jsou kladné, matice $\boldsymbol{K}_k^T \boldsymbol{W} \boldsymbol{K}_k$ je regulární. Regulární je tedy také matice $[\boldsymbol{N} + \boldsymbol{K}_k^T \boldsymbol{W} \boldsymbol{K}_k]$, protože přičteme-li k regulární matici $\boldsymbol{K}_k^T \boldsymbol{W} \boldsymbol{K}_k$

signulární matici $\boldsymbol{N}$, zůstane regularita ve výsledné matici zachována. Funkce `VS2` tedy vygeneruje příslušné výstupy. Vypíše vektor B-splajnových koeficientů a vykreslí graf vyhlazujícího splajnu.

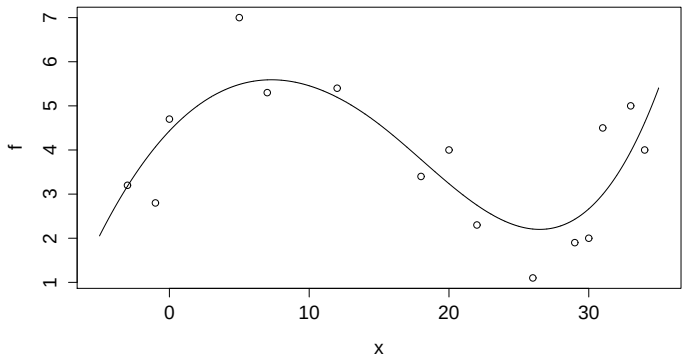
```
> B
```

```
             [,1]
[1,]   1.0598634
[2,]   7.0891654
[3,]   6.3088392
[4,]  -0.9785551
[5,]   5.9766922
```



Obrázek 21: Výstup `VS2` - vyhlazující splajn $s_3^*(x)$

Poznámka 4.3. Opět se podívejme na to, jak se změní graf vyhlazujícího splajnu $s_3^*(x)$, pokud snížíme hodnotu derivace. Na vstupu funkce tedy místo původní hodnoty `der = 2` zadáme derivaci pouze prvního řádu, tj. `der = 1`. Změnou řádu derivace se změní také hodnoty v matici $\boldsymbol{N}$, která ovlivňuje právě vzhled vyhlazujícího splajnu $s_3^*(x)$, viz Obr. 22.



Obrázek 22: Výstup `VS2` - vyhlazující splajn $s_3^*(x)$

5. Splajny v R

Software R se specializuje především na statistické výpočty. Jedná se o volně šířitelný software, který nám nabízí širokou škálu metod a nástrojů, a to od testování hypotéz až po analýzu časových řad. Mimo to je vhodný také pro zpracování velkého počtu dat a při tvorbě grafických výstupů. Software R můžeme také libovolně rozšiřovat o další funkce pomocí balíčků, tzv. `packages`. Tento software si můžeme stáhnout z webové stránky `http://www.R-project.org`, kde najdeme také manuály v angličtině a online nápovědu. Na této stránce jsou k dispozici také zmiňované `packages`, které do R doinstalujeme např. pomocí příkazu `install.packages("jméno_balíčku")`.

5.1. Package splines

Pro práci se splajny R nabízí balíček příkazů s názvem `splines`. Tento balíček obsahuje příkazy, které nám poskytnou jak základní informace o splajnu, tak pomocí nich můžeme určit například jeho řád či uzly. Abychom tyto příkazy mohli použít, je nejprve nutné balíček načíst příkazem `library(splines)`. Samotný balíček obsahuje tyto následující příkazy:

- `splineDesign`
- `bs`
- `ns`
- `polySpline`
- `backSpline`
- `interpSpline`
- `periodicSpline`
- `splineOrder`
- `splineKnots`
- `xyVector`

Ve funkcích, které jsem v rámci této práce naprogramovala, se používá zejména příkaz `splineDesign`, který slouží k vytvoření kolokační matice splajnu. Moji snahou ovšem bylo přiblížit čtenáři celý tento balíček `splines`. Bohužel musím říct, že se mi to ne zcela povedlo. Tento balíček byl vytvořen statistiky, primárně

pro jejich potřeby. Nápověda, která je k balíčku nabízena, je nedostačující a velmi často bylo takřka nemožné výstupy jednotlivých příkazů rozklíčovat. Také z toho důvodu jsem vytvořila své vlastní funkce, které čtenář může využít k interpolaci, metodě nejmenších čtverců a také k vyhlazování splajnů. Navíc, narozdíl od balíčku `splines`, mé funkce neobsahují žádná výrazná omezení a jsou v nich odděleny uzly splajnu od bodů aproximace. Pro úplnost se tedy nyní blíže podívejme na jednotlivé příkazy z tohoto balíčku.

5.1.1. `splineDesign`

Příkaz `splineDesign` slouží k vytvoření kolokační matice splajnu. Jak již bylo řečeno, kolokační matice

$$\boldsymbol{K}_k = (B_i^k(t_j))_{j=1,i=1}^{m\ \ N}$$

je matice funkčních hodnot B-splajnů $B_i^k(x)$ v daných bodech t_j, pro $i = 1, \ldots, N$ a $j = 1, \ldots, m$.

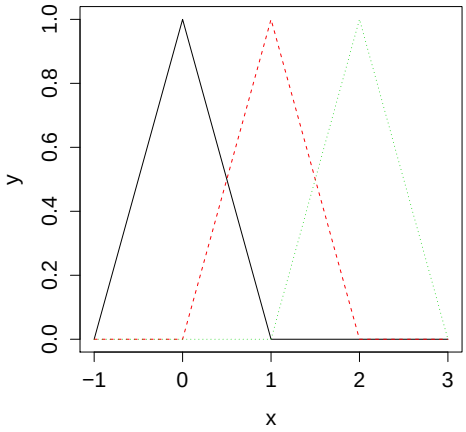
Syntaxe tohoto příkazu je `splineDesign(knots,t,ord)`. Za `knots` dosazujeme rozšířenou síť uzlů (Δy). Způsob vytvoření této posloupnosti (Δy) je popsaný na str. 19 a 20. Dále `t` je posloupnost bodů, v nichž počítáme funkční hodnoty jednotlivých B-splajnů a `ord` udává řád B-splajnu $B_i^k(x)$, pro $i = 1, \ldots, N$.

```
> x=c(0,1,2)
> t=c(0.5,1.5,2)
> K=splineDesign(knots=c(-1,0,1,2,3),t,2)
> K
```

```
     [,1]  [,2]  [,3]
[1,]  0.5   0.5   0.0
[2,]  0.0   0.5   0.5
[3,]  0.0   0.0   1.0
```

Řádky výsledné matice odpovídají bodům posloupnosti $(\Delta t) = \{0.5, 1.5, 2\}$ a sloupce B-splajnům $B_1^2(x), B_2^2(x)$ a $B_3^2(x)$. Prvky uvnitř matice jsou rovny příslušným funkčním hodnotám jednotlivých B-splajnů v bodech posloupnosti

(Δt). Vidíme tedy, že například B-splajn $B_1^2(x)$ má v bodě $t_1 = 0.5$ funkční hodnotu rovnu 0.5, $B_3^2(x)$ má funkční hodnotu v bodě $t_3 = 2$ rovnu jedné apod. Vykreslíme-li si grafy bázových funkcí, můžeme se přesvědčit o tom, že funkční hodnoty uváděné v kolokační matici jsou opravdu správné.



Obrázek 23: Grafy bázových funkcí $B_1^2(x), B_2^2(x)$ a $B_3^2(x)$

5.1.2. `interpSpline`

Příkaz `interpSpline` slouží k sestavení kubického interpolačního splajnu. Syntaxe příkazu je `interpSpline(t,f,bSpline=TRUE)`. Za `t` dosazujeme posloupnost bodů interpolace $(\Delta t) = \{t_1, \ldots, t_m\}$ a `f` jsou příslušné funkční hodnoty f_j, pro $j = 1, \ldots, m$. Za uzly splajnu příkaz bere zadané body interpolace, tj. $(\Delta x) = (\Delta t)$. V případě, že by parametr `bSpline` byl nastaven na hodnotu `FALSE`, příkaz by vrátil polynomickou reprezentaci interpolačního splajnu namísto B-splajnové reprezentace.

Uvažujme posloupnost bodů interpolace $(\Delta t) = \{-3, -1, 0, 2, 4\}$. Příkaz na sestavení interpolačního splajnu bude následující:

```
> interpSpline(c(-3,-1,0,2,4),c(4,6.8,5,7,6.3),bSpline=TRUE)
```

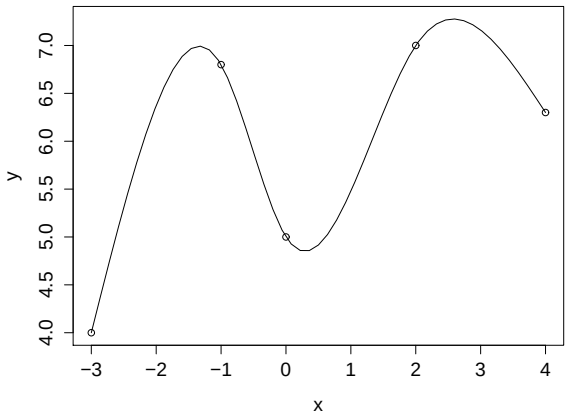
Výstup dostaneme v podobě tabulky:

```
bSpline representation of spline for c(4,6.8,5,7,6.3) ~
                    c(-3,-1,0,2,4)

 -8  -6  -5  -3      -1          0          2
 NA  NA  NA  NA -0.4934896  4.0000000  8.4934896

       4          5          7          9
3.2662760  8.3632812  6.6438802  4.9244792
```

Horní řádek výstupu představuje síť uzlů (Δy), která vznikla rozšířením původní sítě (Δx) z obou stran o $k-1$ uzlů.

Výsledný interpolační splajn, včetně zadaných bodů interpolace, vykreslíme pomocí následující dvojice příkazů:

```
> plot(interpSpline(c(-3,-1,0,2,4),c(4,6.8,5,7,6.3),bSpline=TRUE))
> points(c(-3,-1,0,2,4),c(4,6.8,5,7,6.3))
```



Obrázek 24: Interpolační splajn

Poznámka 5.1. Vidíme tedy, že v R sice je možné sestavit interpolační splajn, ale uživatel musí přijmou určitá omezení. Příkaz `interpSpline` volí za uzly splajnu přímo zadané body interpolace. Navíc je evidentní, že tento příkaz uvažuje okrajové podmínky splajnu. V opačném případě by totiž kolokační matice $\boldsymbol{K}_k$ nemohla vyjít čtvercová a nebylo by tedy možné vypočítat její inverzi a následně vektor

B-splajnových koeficientů. Z dostupné nápovědy také není zcela zřejmé, jak interpretovat druhý řádek výstupu. Z toho důvodu jsem v R vytvořila své vlastní funkce pro vykreslení grafu interpolačního splajnu `IS1` a `IS2`, ve kterých žádná omezení nejsou. Stačí pouze, aby byly splněny podmínky na regularitu kolokační matice a funkce vygeneruje graf interpolačního splajnu a vypíše vektor jeho B-splajnových koeficientů. Mé funkce tedy představují univerzálnější nástroj, jak nalézt požadovaný interpolační splajn.

5.1.3. `polySpline`

Příkaz vytvoří po částech polynomickou reprezentaci zadaného splajnu. Syntaxe příkazu je ve tvaru `polySpline(object)`.

Za `object` dosadíme kubický interpolační splajn, který jsme si zavedli v podkapitole 5.1.2. Příkaz pro vytvoření pp-reprezentace bude vypadat následovně:

```
> object=interpSpline(c(-3,-1,0,2,4),c(4,6.8,5,7,6.3))
> polySpline(object)
```

Výstup je opět v podobě tabulky:

```
polynomial representation of spline for c(4,6.8,5,7,6.3) ~
                    c(-3,-1,0,2,4)

     constant     linear  quadratic       cubic
-3        4.0   2.696094   0.000000  -0.3240234
-1        6.8  -1.192187  -1.944141   1.3363281
 0        5.0  -1.071484   2.064844  -0.5145508
 2        7.0   1.013281  -1.022461   0.1704102
 4        6.3  -1.031641   0.000000   0.0000000
```

Jak již bylo řečeno, výstupem příkazu by měla být po částech polynomická reprezentace splajnu. Nicméně, jak víme z úvodu této práce o pp-reprezentaci,

splajn je na síti uzlů $(\Delta x) = \{-3, -1, 0, 2, 4\}$ určený čtyřmi polynomy:

$$\begin{aligned} P_0(x) &= a_0x^3 + b_0x^2 + c_0x + d_0, \quad x \in (-3, -1), \\ P_1(x) &= a_1x^3 + b_1x^2 + c_1x + d_1, \quad x \in (-1, 0), \\ P_2(x) &= a_2x^3 + b_2x^2 + c_2x + d_2, \quad x \in (0, 2), \\ P_3(x) &= a_3x^3 + b_3x^2 + c_3x + d_3, \quad x \in (2, 4). \end{aligned}$$

Vidíme ale, že příkaz v R nám vrátil polynomů pět. Vzhledem k nedostatečné nápovědě se mi nepodařilo zjistit, jak výstupy tohoto příkazu správně interpretovat.

Poznámka 5.2. Pokud nedefinujeme jinak, standardně je v příkazu `interpSpline` uvažován atribut `bSpline` roven hodnotě `FALSE`. Pro zajímavost se ještě podíváme na to, jestli se nějakým způsobem ovlivní výstup příkazu, změníme-li atribut `bSpline` na hodnotu `TRUE`. Použijeme tedy syntaxi ve tvaru:

```
> object=interpSpline(c(-3,-1,0,2,4),c(4,6.8,5,7,6.3),bSpline=TRUE))
> polySpline(object)
```

Výstup bude mít následující podobu:

```
polynomial representation of spline for c(4,6.8,5,7,6.3) ~
                    c(-3,-1,0,2,4)

   constant     linear      quadratic       cubic
-3      4.0   2.696094   2.220446e-16  -0.3240234
-1      6.8  -1.192187  -1.944141e+00   1.3363281
 0      5.0  -1.071484   2.064844e+00  -0.5145508
 2      7.0   1.013281  -1.022461e+00   0.1704102
 4      6.3  -1.031641   5.181041e-16   0.0000000
```

Je tedy vidět, že výstupy jsou pro parametry `bSpline=TRUE` a `bSpline=FALSE` totožné. Pouze ve sloupci `quadratic` došlo ke změně hodnoty prvního a posledního prvku. To je ovšem způsobeno pouze jiným zaokrouhlením příliš malých hodnot.

5.1.4. backSpline

Příkaz `backSpline` slouží k vytvoření inverze monotónního splajnu. Syntaxe příkazu je ve tvaru `backSpline(object)`, kde `object` definuje splajn, k němuž chceme tuto inverzi vytvořit.

Vstup může být ve tvaru:

```
> t=c(1,2,3,4,5)
> f=c(3,7,10,12,13)
> Int_splajn=interpSpline(t,f,bSpline=FALSE)
> Back_splajn=backSpline(Int_splajn)
```

Jedná se tedy o inverzi kubického interpolačního splajnu. Výstup dostaneme v podobě tabulky:

```
> Back_splajn

          polynomial representation of spline for t ~ f

    constant  linear      quadratic        cubic
3          1    0.25   0.000000e+00  0.000000000
7          2    0.28   1.333333e-02  0.001481481
10         3    0.40  -4.440892e-16  0.025000000
12         4    0.70   3.000000e-01  0.000000000
13         5      NA             NA           NA
```

Porovnáme-li tuto inverzi s výstupem `Int_splajn` původního interpolačního splajnu je patrné, že došlo k záměně souřadnicových os x a y. To znamená, že v inverzi interpolačního splajnu budeme uvažovat uzly splajnu $(\Delta x) = \{3, 7, 10, 12, 13\}$ a k nim funkční hodnoty $f = \{1, 2, 3, 4, 5\}$. V praxi se to projeví tak, že u výstupu příkazu `backSpline` dojde oproti příkazu `interpSpline` k záměně prvních dvou sloupců. Výstup pro původní interpolační splajn by byl tedy ve tvaru:

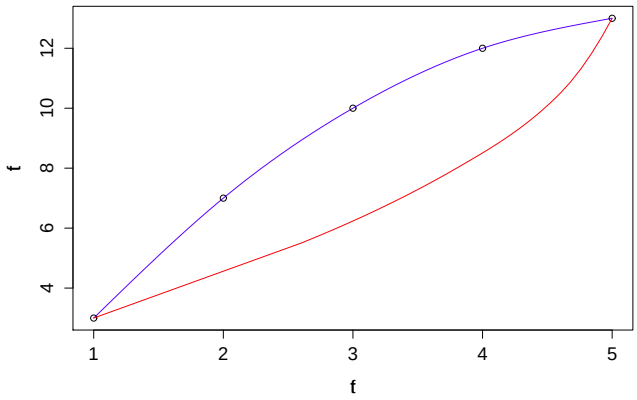
```
> Int_splajn

          polynomial representation of spline for t ~ f
```

```
  constant     linear   quadratic        cubic
1        3  4.2142857   0.0000000 -0.21428571
2        7  3.5714286  -0.6428571  0.07142857
3       10  2.5000000  -0.4285714 -0.07142857
4       12  1.4285714  -0.6428571  0.21428571
5       13  0.7857143   0.0000000  0.00000000
```

Pro vykreslení grafu použijeme v R následující syntaxi:

```
> plot(Int_splajn,col='blue')
> points(t,f)
> par(new=TRUE)
> plot(Back_splajn,col='red',axes=FALSE)
```

Modrá barva na Obr. 25 představuje interpolační splajn, ze kterého jsme vycházeli a červenou je zobrazena výsledná inverze daného interpolačního splajnu.



Obrázek 25: Inverzní splajn

5.1.5. `periodicSpline`

Příkaz `periodicSpline` vytvoří periodický interpolační splajn. Syntaxe příkazu je `periodicSpline(t,f,period)`. Kde `t` je číselný vektor bodů, které chceme interpolovat, `f` udává funkční hodnoty v těchto bodech a `period` definuje periodu výsledného splajnu, kterou si můžeme libovolně zvolit. Také u tohoto příkazu platí, že uzly splajnu jsou totožné se zadanými body interpolace, tj. $(\Delta x) = (\Delta t)$.

Vstup může mít následující podobu:

```
> t = seq(-pi,pi)
> f = cos(t)
> periodicSpline(t,f,period=2*pi)
```

Máme tedy zadané body interpolace:

```
> t
[1] -3.1415927 -2.1415927 -1.1415927 -0.1415927 0.8584073 1.8584073
2.8584073
```

A v nich tyto funkční hodnoty:

```
> f
[1] -1.0000000 -0.5403023 0.4161468 0.9899925 0.6536436 -0.2836622
0.9601703
```

Výstup dostaneme ve tvaru:

```
          bSpline representation of spline for f ~ t

      -5.42478   -4.42478   -3.42478    -3.14159    -2.14159
            NA         NA         NA          NA  -0.9395573
       0.85841    1.85841    2.85841     3.14159     4.14159
     0.4917803  1.1689668  0.7723074  -0.3363348  -0.9395573


                      -1.14159    -0.14159
                    -1.0483230  -0.6392069
                       5.14159     6.14159
                    -1.0483230  -0.6392069
```

První a třetí řádek výstupu představuje uzly výsledného splajnu. To si můžeme ověřit pomocí příkazu `splineKnots`, který vrátí uzly splajnu (tento příkaz si blíže popíšeme v podkapitole 5.1.7.):

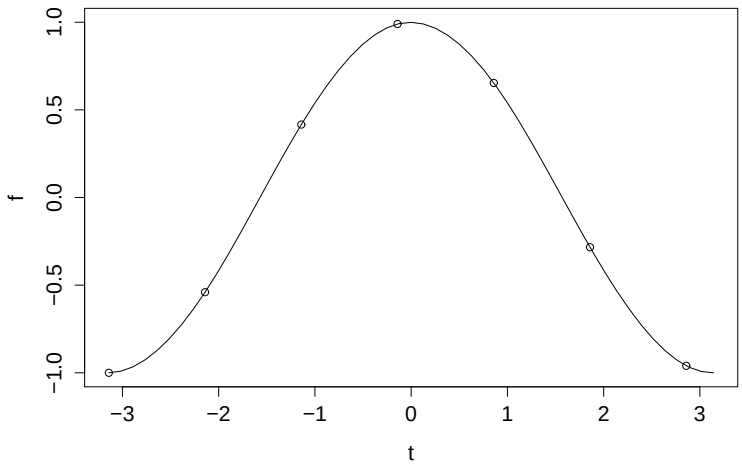
```
> splineKnots(periodicSpline(t,f,period = 2*pi))
```

```
-5.4247780  -4.4247780  -3.4247780  -3.1415927  -2.1415927
-1.1415927  -0.1415927   0.8584073   1.8584073   2.8584073
 3.1415927   4.1415927   5.1415927   6.1415927
```

Význam hodnot ve druhém a čtvrtém řádku výstupu, se mi kvůli špatně popsané nápovědě bohužel nepodařilo rozklíčovat.

Graf periodického splajnu, včetně aproximovaných bodů, získáme v R pomocí dvojice příkazů:

```
> plot(periodicSpline(t,f, period = 2*pi))
> points(t,f)
```



Obrázek 26: Periodický splajn

Nyní již tedy víme, jak v R vytvořit kolokační matici a nadefinovat interpolační splajn, periodický splajn, inverzní splajn aj. Následující příkazy uváděné v podkapitolách 5.1.6. - 5.1.10. slouží pro získání informací právě o těchto splajnech. Tyto příkazy je tedy možné použít pouze ve spojitosti s příkazy z balíčku `splines`.

5.1.6. `splineOrder`

Příkaz `splineOrder` udává, kolika neznámými koeficienty je polynom, v případě pp-reprezentace splajnů, definovaný. Syntaxe tohoto příkazu je ve tvaru

`splineOrder(object)`, kde `object` udává splajn, u jehož polynomů chceme počet neznámých koeficientů zjistit. Za `object` následně dosazujeme některý z příkazů `interpSpline`, `polySpline`, `periodicSpline` nebo `backSpline`.

My budeme uvažovat interpolační splajn. Posloupnost bodů interpolace mějme ve tvaru $(\Delta t) = \{0, 2, 4, 6\}$ a funkční hodnoty v těchto bodech jsou $f = \{4, 6.8, 5, 2\}$. Vstup bude vypadat následovně:

```
> object=interpSpline(c(0,2,4,6),c(4,6.8,5,2))
> splineOrder(object)
```

V tomto případě nám R vrátí hodnotu

```
[1] 4,
```

jedná se tedy o kubický interpolační splajn, jak jsme si uvedli v podkapitole 5.1.2. Tento splajn by byl v případě pp-reprezentace vyjádřený polynomy se čtyřmi neznámými koeficienty a, b, c a d, tj.

$$P(x) = ax^3 + bx^2 + cx + d.$$

5.1.7. `splineKnots`

Příkaz `splineKnots` vrátí uzly zadaného splajnu. Syntaxe příkazu je opět ve tvaru `splineKnots(object)`, kde `object` definuje splajn, jehož uzly chceme zjistit.

Vyjdeme z B-splajnové reprezentace interpolačního splajnu, tj. syntaxe bude ve tvaru:

```
> object=interpSpline(c(0,2,4,6),c(4,6.8,5,2),bSpline=TRUE)
> splineKnots(object)
```

Jako výstup získáme vektor představující rozšířenou síť uzlů splajnu (Δy), která je vytvořená z původní sítě (Δx). Nezadáme-li jinak, bude R uvažovat kubický splajn, tj. splajn řádu $k = 4$ a přidávat tak $k - 1$ uzlů z obou stran původní posloupnosti (Δx).

```
[1] -6 -4 -2 0 2 4 6 8 10 12
```

Poznámka 5.3. V případě, že nebudeme uvažovat B-splajnovou reprezentaci, nám příkaz vrátí přímo uzly splajnu. Mějme tedy syntaxi v následujícím tvaru, ve kterém jsme parametr `bSpline` změnili na hodnotu `FALSE`:

```
> object=interpSpline(c(0,2,4,6),c(4,6.8,5,2),bSpline=FALSE)
> splineKnots(object)
```

Vzhledem k tomu, že u po částech polynomické reprezentace se žádná rozšířená síť (Δy) nevytváří, obdržíme na výstupu přímo uzly splajnu, tj. původní síť uzlů (Δx):

```
[1] 0 2 4 6
```

5.1.8. `bs`

Příkaz `bs` vrátí základní informace o polynomickém splajnu. Obecná syntaxe příkazu je `bs(t,df)`, kde `t` je číselný vektor bodů, v nichž chceme určit funkční hodnoty a `df` udává stupně volnosti. Tyto stupně volnosti jsou dle nápovědy v R doporučeny volit tímto způsobem

```
df = length(t) + (k-1),
```

neboli jako počet uzlů splajnu + jeho stupeň.

Uvažujme kubický splajn definovaný na síti uzlů $(\Delta x) = \{0, 1, 2\}$, kde body aproximace jsou totožné s uzly splajnu. Počet stupňů volnosti určíme ze vztahu $3 + (4 - 1) = 6$. Výsledná syntaxe bude tedy mít následující podobu:

```
> bs(c(0,1,2),6)
```

Výstupem je výčet vlastností:

```
      1          2          3          4  5  6
[1,]  0  0.0000000  0.0000000  0.0000000  0  0
[2,]  0  0.1666667  0.6666667  0.1666667  0  0
[3,]  0  0.0000000  0.0000000  0.0000000  0  1
```

```
attr(,"degree")
[1] 3
attr(,"knots")
25% 50% 75%
0.5 1.0 1.5
attr(,"Boundary.knots")
[1] 0 2
attr(,"intercept")
[1] FALSE
attr(,"class")
[1] "bs"    "basis"    "matrix"
```

Z výstupu příkazu `bs` vidíme, že `degree` je rovno třem, což odpovídá stupni kubického splajnu s vnitřními uzly `knots` 0.5, 1.0 a 1.5. Krajní uzly splajnu `Boundary.knots` jsou potom 0 a 2. Bohužel, ostatní hodnoty výstupu jsou v dostupné nápovědě špatně popsané a z toho důvodu se mi nepodařilo zjistit, jak je správně interpretovat.

Poznámka 5.4. Příkaz `bs` je velmi podobný příkazu `ns`. Syntaxe i výstup těchto dvou příkazů jsou totožné. Rozdíl mezi nimi je v tom, že `ns` vrací informace o přirozeném splajnu. Pro přirozený splajn je typické, že druhé derivace v krajních bodech intervalu, na kterém je definovaný, jsou nulové.

5.1.9. `xyVector`

Příkaz `xyVector` vytváří objekt, který je reprezentovaný dvojicemi hodnot `x` a `y`, přičemž `x`, `y` jsou číselné vektory stejné délky. S výsledným objektem může být zacházeno jako s maticí, datovou tabulkou nebo jako s vektorem.

Na vstupu si nadefinujeme dva vektory a zadáme je do syntaxe příkazu:

```
> x = c(2,7,4,-1)
> y = c(1,3,5,9)
> xyVector(x,y)
```

Výstup dostaneme v následující podobě:

```
$ x
[1] 2 7 4 -1
$ y
[1] 1 3 5 9
attr(,"class")
[1] "xyVector"
```

5.2. Vykreslení bázových funkcí v R

Jak jsme si mohli všimnout, v balíčku `splines` úplně chybí jakýkoliv příkaz pro vykreslení grafu B-splajnu $B_i^k(x)$. Z toho důvodu jsem v R vytvořila svoji vlastní funkci, která pro vykreslení používá příkaz `splineDesign`. Uživatel do funkce zadá řád B-splajnu k a síť uzlů (Δx), na které má být daný B-splajn definovaný. Z těchto vstupních údajů funkce vytvoří kolokační matici, tj. matici funkčních hodnot B-splajnů v daných bodech, která se následně použije pro vykreslení grafu splajnu. Celá funkce vypadá následovně:

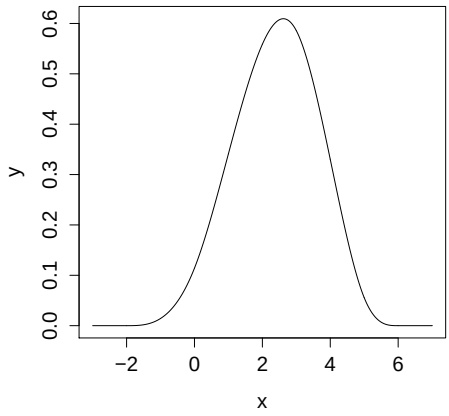
```
BSpline = function(k,x) {
# VSTUP:
# x = sit uzlu,
# k = rad B-splajnu.
# VYSTUP: graf prislusneho B-splajnu.
r = length(x)
# Overeni podminek
if (k != round(k)) stop ('rad B-splajnu musi byt cele cislo')
if (k <= 0) stop ('rad B-splajnu musi byt kladne cislo')
for (i in 1:(r-1)){
if (x[i] >= x[i+1]) stop ('x musi byt rostouci posloupnost')}
if (r != (k+1)) stop ('pocet uzlu musi byt k+1')
# Kolokacni matice
deleni = seq(min(x)-1, max(x)+1, by = 0.001)
```

```
K = splineDesign(x,deleni,k, outer.ok = TRUE)
plot(deleni, K, type = "l", xlab = "x", ylab = "y")}
```

Z kódu je patrné, že graf B-splajnu $B_i^k(x)$ není vykreslován pouze na síti uzlů (Δx), ale že tato síť je z obou stran rozšířená. Je to z toho důvodu, aby bylo na první pohled zřejmé, že B-splajn je kladný pouze na svém nosiči a mimo něj je nulový.

Pro vykreslení grafu kubického B-splajnu, který je definovaný na síti uzlů $(\Delta x) = \{-2, 0, 3, 5, 6\}$, stačí v R zadat následující příkaz:

```
> BSpline(4,c(-2,0,3,5,6))
```



Obrázek 27: Výstup funkce `BSpline`

Z kapitoly 1.2 víme, že posloupnost B-splajnů $\left\{B_i^k(x)\right\}$ tvoří bázi. Pro vykreslení této báze jsem vytvořila následující dvě funkce - `Baze1` a `Baze2`. Rozdíl mezi nimi je v tom, jak se vytváří rozšířená síť uzlů splajnu (Δy) z původní sítě (Δx). U funkce `Baze1` se rozšířená síť uzlů vytváří přidáním $k-1$ ekvidistantních uzlů z obou stran sítě (Δx). V druhém případě, u funkce `Baze2`, se (Δy) vytváří přidáním $k-1$ násobných uzlů. Na vstupu obou funkcí uživatel zadává řád splajnu k a síť uzlů (Δx). Výstupem funkcí jsou grafy příslušných bázových funkcí $B_1^k(x), \ldots, B_N^k(x)$. Tyto bázové funkce jsou vykresleny na základě hodnot

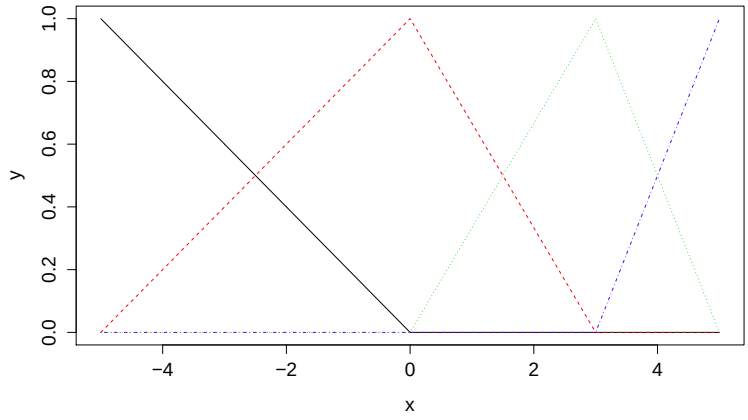
z kolokační matice, která je vytvořena příkazem `splineDesign`. Na ukázku si zde představíme pouze jednu z funkcí, kterou je `Baze2`.

```
Baze2 = function(k,x){
# VSTUP:
# k = rad B-splajnu,
# x = sit uzlu.
# VYSTUP: graf bazovych funkci definovanych na x.
r = length(x)
# Overeni podminek:
if (k != round(k)) stop ('rad k neni cele cislo')
if (k <= 0) stop ('rad splajnu musi byt kladne cislo')
for (i in 1:(r-1)){
if (x[i] >= x[i+1]) stop ('x musi byt rostouci posloupnost')}
# Rozsirena sit uzlu
y = c()
if (k==1){
for (i in 1:r){y[i] = x[i]}}
if (k != 1){
Celkova_Delka = 2*(k-1) + r
y[1:(k-1)] = x[1]
y[k:(k+r-1)] = x[1:r]
y[(k+r):Celkova_Delka] = x[r]}
deleni = seq(min(y), max(y), by = 0.001)
lambda = c(0:(r-1))
g = lambda[lambda_delka - 1]
# Dimenze prostoru
N = g+(k-1)+1
# Kolokacni matice
C = array(0, c(length(deleni),N))
l = c()
```

```
for(i in (1:N)){
for (j in 1:(k+1)){
l[j] = y[i+j-1]}
C[ ,i] = splineDesign(l, deleni, k, outer.ok = TRUE)}
C[1,1] = 1
C[length(deleni),N] = 1
matplot(deleni, C, type = "l", xlab = "x", ylab = "y")}
```

Uvažujme například řád splajnu $k = 2$ a síť uzlů $(\Delta x) = \{-5, 0, 3, 5\}$. U funkce `Baze2` se rozšířená síť uzlů tvoří z (Δx) přidáním násobných uzlů. V našem případě tedy bude ve tvaru $(\Delta y) = \{-5, -5, 0, 3, 5, 5\}$. Pro vykreslení grafu bázových funkcí $B_1^2(x), \ldots, B_4^2(x)$ stačí v R zadat následující příkaz:

```
> Baze2(2,c(-5,0,3,5))
```



Obrázek 28: Výstup funkce `Baze2`

Závěr

Ještě před rokem byl pro mě pojem „bázový splajn“ velkou neznámou. Nevěděla jsem, že se jedná pouze o jednu část rozsáhlé teorie splajnů a už vůbec jsem nevěděla, že lze bázové splajny využít také k aproximaci, čímž si můžeme značně ulehčit náš „matematický život“.

Vzhledem k tomu, že na počátku mého snažení byla má znalost bázových splajnů nulová, nejprve jsem si musela nastudovat potřebnou teorii. Tato teorie byla obsahem první kapitoly, ve které jsme si zavedli pojem bázový splajn, ukázali si, jak tento B-splajn vypadá a jaké jsou jeho vlastnosti. Jednou z těchto vlastností byl také fakt, že B-splajny tvoří bázi, s čímž souvisely pojmy jako rozšířená síť uzlů či kolokační matice. Veškerou tuto teorii jsem se snažila čtenáři podat srozumitelnou formou. Čímž se dostávám k prvnímu problému, na který jsem při psaní mé diplomové práce narazila a tím byla literatura. Skripta „Splajny“ od J. Kobzy [2] sice veškerou potřebnou teorii obsahují, nicméně abych byla upřímná, často mě tato skripta v dané problematice více dezorientovala, než aby mi ji pomohla objasnit. I proto jsem se snažila o to, aby kapitola věnovaná teorii byla snadno pochopitelná a pomohla tak případným budoucím studentům v jejich snaze o nastudování B-splajnů.

Cílem této diplomové práce bylo vytvořit vyhlazující splajny v R. K tomuto cíli jsme začali pomalu směřovat od druhé a třetí kapitoly, ve kterých jsme se seznámili s interpolací a metodou nejmenších čtverců. Teorie těchto aproximačních metod byla doplněna o mé vlastní funkce vytvořené v softwaru R, které čtenáři přiblížily použití těchto metod na konkrétních hodnotách. Následující pátá kapitola se již věnovala samotným vyhlazujícím splajnům. V teorii jsme využili znalostí, které jsme získali v předchozích dvou kapitolách věnovaných interpolaci a metodě nejmenších čtverců. Tato teorie byla opět vhodně doplněna o funkci, na které si čtenář mohl vyzkoušet vyhlazování splajnů v praxi.

Většina funkcí, které jsem v rámci této práce v softwaru R vytvořila, vycházela z balíčku příkazů `splines`. Pro moji práci byl stěžejní zejména příkaz `splineDesign`, který jsem používala k vytvoření kolokační matice splajnu. Jak

nám ale kapitola 5.1 ukázala, využití tohoto balíčku je dosti omezené. I z toho důvodu jsem se snažila o vytvoření univerzálnějšího nástroje, který by mohl čtenář pro práci se splajny použít.

Obecně si samozřejmě můžeme pro práci se splajny vybrat libovolný matematický software. Jak již bylo naznačeno, v této práci padla volba na R, ve kterém není problematika splajnů až tak rozšířená. Také pro mě samotnou byla práce s tímto softwareme určitou výzvou, nikdy dřív jsem jej totiž aktivně nevyužívala. Bohužel musím říct, že ani po roce vzájemného seznamování mi tento matematický software k srdci nepřirostl. Byl pro mě uživatelsky nepřívětivý a také vizuální podoba některých jeho výstupů nebyla z mého pohledu vždy optimální. Například z jakého důvodu nejsou všechny sloupce matice zaokrouhleny na stejný počet desetinných míst, bylo pro mě velkou záhadou. Nicméně si myslím, že nakonec jsem nad tímto softwarem zvítězila, což dokládá také fakt, že v opačném případě byste právě nečetli závěr mé diplomové práce.

Literatura

[1] Dierckx, P: *Curve and Surface Fitting with Splines*. Clarendon Press, 1993.

[2] Kobza, J.: *Splajny*. Vyd. 1, Olomouc: Vydavatelství Univerzity Palackého v Olomouci, 1993.

[3] Machalová, J.: *Optimal interpolating and optimal smoothing spline*. Journal of eletrical engineering, vol. 53, no. 12/s, 2002, 79 - 82.

[4] Machalová, J.: *Optimální interpolující a vyhlazující splajny*. Dizertační práce. Univerzita Palackého v Olomouci, 2002.

[5] Machalová, J., Netuka, H.: *Numerické metody nepodmíněné optimalizace*. Vyd. 1, Olomouc: Vydavatelství Univerzity palackého v Olomouci, 2013.